UNIVERSIDAD VERACRUZANA

INSTITUTO DE INVESTIGACIONES EN INTELIGENCIA ARTIFICIAL

CLASIFICACIÓN DE ACTIVOS FINANCIEROS CON ALGORITMOS DE APRENDIZAJE SUPERVISADO

T E S I S

Para obtener el título de:

Maestro en Inteligencia Artificial

Presenta:

Lic. Carlos Alexis Barrios Bello

Director:

Dr. Horacio Tapia-McClung

22 de septiembre de 2025

Agradecimientos

Quiero expresar mi más sincero agradecimiento al **Dr. Horacio Tapia-McClung**, director de esta tesis, por su guía, paciencia y compromiso durante todo el desarrollo de este trabajo. Su experiencia y consejos fueron fundamentales para mi primer acercamiento a la investigación.

Agradezco también a mis **profesores** del *Instituto de Investigaciones en Inteligencia Artificial* de la *Universidad Veracruzana*, por brindarme un entorno propicio para el aprendizaje y la investigación. En específico, agradezco a aquellos docentes que, con sus clases, me llenaron de curiosidad y conocimientos para continuar con la maestría.

Agradezco profundamente a mis sinodales: al **Dr. Alejandro Guerra**, por sus valiosas observaciones y el rigor de su revisión; a la **Dra. Rosa Madrid**, por sus sugerencias metodológicas y su guía crítica en la aplicación de esta investigación; y al **Dr. Nicandro Cruz Ramírez**, por su apoyo constante en las implementaciones y su orientación a lo largo de la maestría. Sus comentarios y acompañamiento fueron esenciales para fortalecer este trabajo.

A mis **compañeros de maestría**, gracias por las conversaciones, el apoyo mutuo y por compartir este camino lleno de retos y aprendizajes. En especial, agradezco a mi amigo **Fís. Ramiro Villegas Vega** por esas pláticas después de clases y su apoyo para resolver dudas.

A mis amigos de la licenciatura: **Hugo Rivera**, **José Castillo**, **Gustavo Espejo**, **Carlos Bernabe**, **Isaías Silíceo**, **Aarón Ortíz** y otros, por su apoyo y amistad durante la carrera. Sin ellos, habría sido mucho más complicado conseguir mi título de licenciatura.

A **mi familia**, especialmente a **mi madre**, por su amor incondicional, por enseñarme a nunca rendirme y por apoyarme en cada etapa de mi vida.

En especial, agradezco a **mi mascota, Nano**, sea donde sea que estés, siempre apreciaré que me hayas acompañado toda tu vida y hayas sido un gran compañero de clases todo el tiempo que estuve en casa tomando clases o estudiando, ambos nos titulamos, felicidades amigo mío.

Y finalmente, gracias a **ti**, que estás leyendo este trabajo.

Índice general

Resumen

Tradicionalmente, para entender cómo se comportan las inversiones (por ejemplo, acciones y bonos) y estimar cuánto podrían ganar o perder, se han utilizado métodos estadísticos como las regresiones y el análisis de múltiples factores. Estos métodos funcionan bien para ciertas situaciones, pero no explican por completo todas las variaciones en los precios y, además, se vuelven complicados cuando se manejan grandes volúmnenes de datos o patrones más complejos.

Para mejorar esto, **la Inteligencia Artificial (IA) y sus técnicas aplicadas a distintas áreas** ofrecen nuevas maneras de predecir y clasificar cómo se comportarán las inversiones. En lugar de usar únicamente fórmulas lineales, se pueden usar algoritmos avanzados, que son más flexibles para capturar patrones complejos. También existen maneras de organizar y elegir los datos (llamadas "discretización" y "selección de características"), que ayudan a que las computadoras trabajen más rápido y con mayor precisión al momento de hacer predicciones.

En este trabajo, se quiere **clasificar** las acciones según su posible comportamiento en lugar de sólo usar la tradicional "predicción de cuánto van a subir o bajar". Para ello:

1. Se **convertirán** datos continuos (como precio y tamaño de la empresa) en grupos o categorías más manejables.

2. Se **usarán** distintos algoritmos de IA (como redes neuronales, Naive Bayes, entre otros) para ver cuál se desempeña mejor.

3. Se **compararán** los resultados de usar estos métodos con los datos continuos sin tratar, buscando quién ofrece más exactitud y eficiencia.

4. Se **evaluarán** diferentes formas de elegir las características más importantes (por ejemplo, usando la teoría financiera o técnicas de IA).

La información que se utilizará viene de las **bases de datos de CRSP**, que reúnen registros de cerca de **30 mil acciones (en mercados como NYSE, AMEX y NASDAQ)** entre los años 2001 y 2020. Se tienen 94 características distintas de cada acción (por ejemplo, tamaño de la empresa, precio, volatilidad, entre otros.) incluyendo variables especulativas y fundamentales, lo que permite analizar muchísima información sobre cómo se comportan los activos financieros.

Este trabajo propone cambiar el enfoque basado en ecuaciones lineales por métodos más avanzados de IA que clasifiquen las acciones. Así, se podrá **descubrir patrones** ocultos y mejorar la forma en que se entiende y se calcula el valor de las inversiones, ayudando a crear estrategias de inversión más sólidas en el futuro.

Conceptos Financieros Clave

Activos reales: aquellos que ayudan a producir bienes y servicios para aportar a la economía de alguna entidad [10].

Activos financieros: derechos sobre los ingresos generados por estos activos reales, es decir, son un instrumentos que representan un derecho sobre futuros flujos de dinero, bienes, entre otros, como lo pueden ser acciones de una empresa, bonos emitidos por gobiernos, corporaciones u otras identidades, fondos de inversión (que son un conjunto de activos financieros), derivados financieros (cuyo valor depende de otro activo), entre otros [10].

Retorno: se define como la proporción de la diferencia del valor de inversión final y su valor inicial entre su valor inicial [6]; de tal manera que el retorno puede ser positivo, si se obtuvo una diferencia favorable al inversionista, o negativo en el caso contrario.

Tamaño: se refiere a la capitalización de mercado de la empresa, el precio de la acción multiplicado por el número de acciones en circulación [24].

Momentum: es la persistencia de las tendencias en los precios [42].

Book-to-market: es la proporción entre el valor contable de la empresa y su valor de mercado [24].

Liquidez: facilidad y velocidad de comprar/vender una acción.

Volatilidad: variabilidad de los precios de un activo.

Volatilidad idiosincrática: volatilidad que no depende del mercado.

Cartera o Portafolio: conjunto de distintos activos financieros.

Fondo de inversión: conjunto de distintos activos financieros de diferentes inversionistas.

Tasa libre de riesgo: rentabilidad esperada de una inversión que se considera segura y sin riesgo.

Dividendos: forma de remunerar a los accionistas [46].

Riesgo sistemático: peligro de obtener una pérdida significativa [73].

Riesgo no sistemático: es la volatilidad de una inversión [52].

Coeficiente Beta (β): es una medida del riesgo sistemático de un activo financiero en relación con el mercado en su conjunto. Cuantifica la sensibilidad del rendimiento de un activo respecto a los movimientos del rendimiento del mercado [73].

Retorno idiosincrático: es la variación porcentual entre el precio final y el precio inicial de un activo financiero en términos de la variación del mercado.

Variables especulativas: son aquellos que se asocian con el comportamiento del mercado, como precios, retornos o volatilidades.

Variables fundamentales: se refiere a las variables de los fundamentos sólidos, como indicadores de rentabilidad, razones de valoración o proporciones contables.

Capítulo 1

Introducción

Los mercados de capitales constituyen el punto de encuentro entre **activos reales**, aquellos que generan bienes y servicos, y los **activos financieros**, que confieren a sus tenedores derechos sobre los flujos de caja futuros [10]. Desde los trabajos pioneros de Markowitz y Sharpe, la relación entre *riesgo sistemático* y *retorno* ha sido formalizada a través de la varianza, la **beta** del mercado y otros indicadores derivados de la **volatilidad** [55, 73]. Sin embargo, la literatura muestra que factores adicionales, como el **tamaño**, **momentum**, **book-to-market**, **liquidez** e incluso la **volatilidad idiosincrática**, explican parte importante de la prima de riesgo [24]. Estos hallazgos destacan la complejidad en la tarea de pronosticar el **retorno**, definido como la variación porcentual entre los valores inicial y final de una inversión.

La predicción del retorno de activos financieros se ha abordado históricamente mediante la prueba de diferentes modelos que, además de los propios retornos pasados, añaden características que pueden explicar la dinámica de este fenómeno [23, 26, 35]. La evolución de esta problemática puede dividirse en dos etapas principales que propongo:

Primera etapa, caracterizada por el desarrollo de modelos económicos tradicionales para la predicción de activos utilizando métodos estadísticos (como las regresiones lineales).

Segunda etapa, marcada por la incorporación de técnicas avanzadas y métodos de aprendizaje automático para mejorar dichas predicciones.

Primera etapa: Modelos económicos tradicionales y métodos estadísticos

El punto de partida de la teoría moderna de carteras se remonta al trabajo pionero de Harry Markowitz en 1952. En su artículo " Portfolio Selection " [55], Markowitz introduce el concepto de diversificación eficiente para maximizar el retorno ajustado al riesgo. Su enfoque se basa en la media y la varianza de los retornos, y utiliza métodos estadísticos clásicos, como la regresión lineal y el análisis de varianza, para modelar y optimizar las carteras de inversión.

La década de 1960 vio avances significativos con el desarrollo del Modelo de Valoración de Activos de Capital (CAPM) [52, 73]. Sharpe, en su artículo " Capital Asset Prices: A Theory of Market Equilibrium under Conditions of Risk " [73], presenta el CAPM como un marco para entender el equilibrio en los mercados financieros bajo condiciones de riesgo. El modelo

establece una relación lineal entre el riesgo sistemático de un activo y su retorno esperado, utilizando regresiones lineales para estimar el coeficiente beta como medida del riesgo.
Simultáneamente, Lintner complementa el modelo CAPM [52] enfatizando la importancia del riesgo sistemático y distingue entre riesgo sistemático y no sistemático, argumentando que los inversores sólo deben ser compensados por asumir riesgo que no puede diversificarse. Se usó la regresión lineal para estimar las relaciones entre riesgo y retorno.

En los 70s, se llevaron a cabo diversas pruebas empíricas del CAPM utilizando regresiones y análisis estadísticos. Black, Jensen y Scholes, en "*The Capital Asset Pricing Model: Some Empirical Tests*" [9], y Fama y MacBeth, en "*Risk, Return, and Equilibrium: Empirical Tests*" [26], evaluaron la validez del modelo, encontrando inconsistencias y sugiriendo que la beta de mercado, que es el coeficiente que acompaña cada factor en la regresión lineal, no siempre es suficiente para explicar los retornos de los activos.
Merton [59], introduce un modelo intertemporal que extiende el CAPM al considerar múltiples periodos y oportunidades de inversión cambiantes. Aunque su enfoque es más complejo, sigue basándose en métodos estadísticos y en modelos de regresión para estimar las relaciones entre las variables financieras. Paralelamente, Ross propone la Teoría de Arbitraje de Valoración de Activos (APT) en "*The Arbitrage Theory of Capital Asset Pricing*" [70], como una alternativa al CAPM. La APT sostiene que los retornos de los activos están influenciados por múltiples factores de riesgo (como el riesgo del mercado, laboral, crédito, entre otros), y utiliza modelos de regresión multifactoriales para estimar las sensibilidades de los activos a estos factores.

Durante la década de 1980, estudios empíricos continuaron desafiando al CAPM. Banz [5], descubre el "efecto tamaño", observando que las empresas más pequeñas tienden a ofrecer mayores retornos ajustados por riesgo.
Por otro lado, aparece Merton de nueva cuenta [60], él hace una crítica a los métodos tradicionales de estimación del retorno esperado del mercado, proponiendo modelos que consideran cambios en el nivel de riesgo. Aunque introduce nuevas consideraciones teóricas, sigue utilizando métodos estadísticos convencionales para la estimación.
Por último, Rosenberg et al. en "*Persuasive Evidence of Market Inefficiency*" [69], proporciona evidencia de ineficiencias en el mercado al mostrar que las estrategias basadas en la proporción de valor contable a precio y en la reversión de retornos pueden generar retornos anormales.

En los años 90s, Fama y French, en "*The Cross-Section of Expected Stock Returns*" [25], y posteriormente en "*Common Risk Factors in the Returns on Stocks and Bonds*" [24], proponen modelos multifactoriales que incorporan el tamaño de la empresa y el valor contable sobre el valor de mercado como factores determinantes de los retornos esperados. Estos modelos utilizan regresiones multifactoriales para estimar las relaciones entre los factores y los retornos, ampliando el marco del CAPM, pero manteniendo el uso de métodos estadísticos.

Lo anterior queda resumido en las Tablas (1.1) y (1.2) que muestran los principales antecedentes de la primera etapa de la investigación para predecir activos financieros.

Tabla 1.1: Principales antecedentes de la primera etapa: Modelos económicos tradicionales y métodos estadísticos (1).

Autor(es)	**Año**	**Título**	**Hipótesis de trabajo**
Friedman, Milton and Savage, L. J.	1948	*The Utility Analysis of Choices Involving Risk [29]*	El análisis de la utilidad y las elecciones bajo riesgo ayuda a explicar cómo los inversores toman decisiones de cartera.
Markowitz, Harry	1952	*Portfolio Selection [55]*	La selección de carteras basada en media-varianza mejora la eficiencia en el rendimiento ajustado al riesgo.
Sharpe, William F.	1964	*Capital Asset Prices: A Theory of Market Equilibrium under Conditions of Risk [73]*	El riesgo sistemático (beta) es el factor clave que determina el retorno esperado de un activo.
Lintner, John	1965	*The Valuation of Risk Assets and the Selection of Risky Investments in Stock Portfolios and Capital Budgets [52]*	Los inversores deben ser compensados únicamente por el riesgo no diversificable; se extiende de la lógica CAPM.
Black, Fisher and Jensen, Michael C. and Scholes, Myron	1972	*The Capital Asset Pricing Model: Some Empirical Tests [9]*	Verifican empíricamente el CAPM y señalan posibles anomalías en la relación beta-retorno.
Fama, Eugene F. and French, Kenneth R.	1973	*Risk, Return, and Equilibrium: Empirical Tests [26]*	La relación entre beta y rendimiento no siempre es estable, sugiriendo limitaciones del CAPM.
Merton, Robert C.	1973	*An Intertemporal Capital Asset Pricing Model [59]*	El rendimiento de los activos depende de consideraciones intertemporales y cambios en las oportunidades de inversión.
Ross, Stephen A.	1976	*The Arbitrage Theory of Capital Asset Pricing [70]*	Los rendimientos responden a múltiples factores de riesgo (enfoque multifactorial APT).

Tabla 1.2: Principales antecedentes de la primera etapa: Modelos económicos tradicionales y métodos estadísticos (2).

Autor(es)	**Año**	**Título**	**Hipótesis de trabajo**
Merton, Robert C.	1973	*An Intertemporal Capital Asset Pricing Model [59]*	El rendimiento de los activos depende de consideraciones intertemporales y cambios en las oportunidades de inversión.
Ross, Stephen A.	1976	*The Arbitrage Theory of Capital Asset Pricing [70]*	Los rendimientos responden a múltiples factores de riesgo (enfoque multifactorial APT).
Merton, Robert C.	1980	*On estimating the expected return on the market: An exploratory investigation [60]*	Critica métodos estáticos de estimación y propone modelos dinámicos para el retorno esperado del mercado.
Banz, Rolf W.	1981	*The relationship between return and market value of common stocks [5]*	El "efecto tamaño" sugiere que las empresas pequeñas pueden ofrecer mayores retornos ajustados por riesgo.
Rosenberg, Ban and Reid, Kenneth and Lanstein, Ronald	1998	*Persuasive Evidence of Market Inefficiency [69]*	Hallan ineficiencias de mercado y muestran que factores como el valor contable/precio o la reversión de retornos generan rendimientos anormales.
Fama, Eugene F. and French, Kenneth R.	1992	*The Cross-Section of Expected Stock Returns [25]*	La inclusión de factores de tamaño y valor (size y book-to-market) mejora la explicación de los retornos respecto del CAPM.
Fama, Eugene F. and French, Kenneth R.	1993	*Common Risk Factors in the Returns on Stocks and Bonds [24]*	Modelo multifactorial que incorpora tamaño y valor; mantiene el método estadístico tradicional (regresión lineal).

Segunda etapa: Incorporación de técnicas avanzadas y métodos de aprendizaje automático en la valoración de activos

Con el avance de la tecnología y el incremento en la capacidad de procesamiento de datos, a partir de la década de 2000 comienza a gestarse una segunda etapa en la valoración de activos financieros. Esta nueva fase se caracteriza por la incorporación de técnicas avanzadas, incluyendo métodos de aprendizaje automático y modelos no lineales, para capturar la complejidad y las relaciones en los mercados financieros.

En la primera década del 2000, Pástor, en "*Portfolio Selection and Asset Pricing Models*" [64], introduce un enfoque bayesiano para la selección de carteras que permite a los inversores incorporar su grado de confianza en un modelo de valoración de activos. Este enfoque combina modelos tradicionales con evidencia empírica, permitiendo ajustar las creencias previas del inversor en función de los datos observados.

Durante la década de 2010, el uso de técnicas de aprendizaje automático se intensifica con trabajos como el de Kelly y Pruitt, "*The Three-Pass Regression Filter: A New Approach to Forecasting Using Many Predictors*" [46], donde presentan el método conocido como 3PRF. Este combina técnicas de reducción de dimensionalidad con regresiones, acercándose al ámbito del aprendizaje automático, mismo que permite manejar una gran cantidad de predictores para mejorar la precisión de los pronósticos financieros. Los mismo autores, un año después, emplearían técnicas de *Partial Least Squares* (PLS) para predecir retornos del mercado y crecimiento de dividendos utilizando proporncciones contables como el *book-to-market*. PLS es una técnica de aprendizaje estadístico que permite manejar múltiples predictores y capturar relaciones entre variables [45].

Luego, Fama y French, reconociendo las limitaciones de sus modelos anteriores, proponen un modelo de cinco factores en "*A Five-Factor Asset Pricing Model*" [23], incorporando factores de rentabilidad y de inversión para mejorar la explicación de las variaciones en los retornos promedios de las acciones.

Todo estos hallazgos fueron de inspiración para Kelly, Pruitt y Su, y su artículo "*Characteristics Are Covariances: A Unified Model of Risk and Return*" [47] donde usaron técnicas de aprendizaje automático y modelos de factores. Estos autores introducen el método *Instrumented Principal Component Analysis* (IPCA), que permite identificar factores latentes y cargas dinámicas mediante características observables de los activos, unificando modelos tradicionales de factores y basados en características. El IPCA es una técnica avanzada que combina análisis de componentes principales con regresiones instrumentadas [47].

Por último, Kozak, Nagel y Santosh, en "*Shrinking the Cross Section*" [51], abordan el desafío de trabajar con una gran cantidad de predictores mediante un enfoque bayesiano que impone una restricción de reducción (*shrinkage*) en los coeficientes del factor de descuento estocástico (SDF). Este enfoque utiliza técnicas de regularización, como el Lasso y Ridge, que penalizan la complejidad del modelo para mejorar la generalización [51].

En los últimos años, se ha explotado aún más los métodos de Inteligencia Artificial, por ejemplo, bosques de regresión, redes neuronales, PCA, PLS, autoencoders, entre otros. Lo anterior se ve reflejado con los autores Giglio, Kelly y Xiu, en "*Factor Models, Machine Learning, and Asset Pricing*" [32], revisan los avances recientes en la valoración de activos utilizando

modelos de factores y herramientas de aprendizaje automático. Destacan cómo estas técnicas permiten manejar mejor la alta dimensionalidad de los datos financieros y mejorar las predicciones, integrando modelos de factores estáticos y condicionales con métodos de aprendizaje automático. Además, Gu, Kelly y Xiu aportan significativamente al campo con dos trabajos clave. En "*Autoencoder Asset Pricing Models*" [35], proponen un modelo basado en *autoencoders*, redes neuronales no supervisadas utilizadas para la reducción de dimensionalidad, permitiendo identificar factores latentes de manera no lineal y mejorando la predicción de rendimientos de activos.

Otro trabajo con un uso exhaustivo de aprendizaje automático es en "*Empirical Asset Pricing via Machine Learning*" [36], donde superan las limitaciones de los enfoques tradicionales al aplicar distintas métodos de este aprendizaje. Demuestran que técnicas como árboles de regresión, bosques aleatorios y redes neuronales profundas superan a las estrategias basadas en regresiones lineales en el manejo de grandes volúmenes de datos.

Estos avances reflejan una tendencia creciente hacia la utilización de técnicas de aprendizaje automático, aprovechando su capacidad para manejar grandes volúmenes de datos y capturar relaciones complejas respecto a la variable retorno.

La segunda etapa puede resumir en las Tablas (1.3) y (1.4).

1.1. Planteamiento y justificación del problema de investigación.

Los modelos tradicionales de valoración de activos, como el de Sharpe [73], Markowitz [55], Fama y French [26], entre otros, han identificado factores comunes de riesgo en el retorno de acciones y bonos. Otro modelo de Fama y French [24] considera tres factores de mercado relacionados con el tamaño de la empresa y el *proporción de book-to-market*, así como dos factores adicionales relacionados con el riesgo de vencimiento y de incumplimiento para bonos. Aunque estos modelos han proporcionado una base sólida para la comprensión de los retornos de activos, sus resultados reportados expresan un sobreajuste en los modelos usados y las métricas tienen rendimientos bajos menores al 15 % en R^2 o inclusive valores negativos para la misma métrica [36].

La formación de múltiples características de los activos financieros se basa en conceptos ampliamente discutidos en la literatura, como: **el tamaño, momentum y book-to-market** [47].

Por otro lado, Gu, Kelly y Xiu [35] señalan un conjunto de características que consideran más relevantes para describir los activos financieros (A.1), las cuales pueden dividirse en los siguientes grupos:

- **Basadas en las tendencias de los precios recientes:** Son aquellas relacionadas con distintas definiciones de momentum en diversos plazos.
- **Basadas en las variables líquidas:** Son indicadores de liquidez, como la rotación de acciones, el valor en dólares transado, o el costo de las transacciones.

Tabla 1.3: Principales antecedentes de la segunda etapa: Incorporación de técnicas avanzadas y métodos de aprendizaje automático (1).

Autor(es)	**Año**	**Título**	**Hipótesis de trabajo**
Pastor, Lubos	2000	*Portfolio Selection and Asset Pricing Models [64]*	Enfoque bayesiano que combina modelos tradicionales con evidencia empírica, permitiendo ajustar creencias previas a los datos.
Kelly, Brian and Pruitt, Seth	2011	*The Three-Pass Regression Filter: A New Approach to Forecasting Using Many Predictors [46]*	Proponen el 3PRF que integra reducción de dimensionalidad y regresiones para aprovechar múltiples predictores financieros.
Kelly, Brian and Pruitt, Seth	2012	*Market Expectations in the Cross-Section of Present Values [45]*	Emplean *Partial Least Squares* (PLS) para pronosticar retornos y crecimiento de dividendos, manejando múltiples proporciones contables.
Fama, Eugene F. and French, Kenneth R.	2014	*A Five-Factor Asset Pricing Model [23]*	Añaden factores de rentabilidad e inversión para explicar mejor las variaciones de retornos promedio de las acciones.
Kelly, Bryan T. and Pruitt, Seth and Su, Yinan	2019	*Characteristics are covariances: A unified model of risk and return [47]*	Proponen *Instrumented PCA* (IPCA) para identificar factores latentes y cargas dinámicas, unificando modelos basados en características y factores.

Tabla 1.4: Principales antecedentes de la segunda etapa: Incorporación de técnicas avanzadas y métodos de aprendizaje automático (2).

Autor(es)	**Año**	**Título**	**Hipótesis de trabajo**
Kozak, Serhiy and Nagel, Stefan and Santosh, Shrihari	2020	*Shrinking the Cross Section [51]*	Método bayesiano con regularización (*shrinkage*) para controlar el sobreajuste en modelos con gran número de predictores.
Gu, Shihao and Kelly, Bryan and Xiu, Dacheng	2020	*Empirical Asset Pricing via Machine Learning [36]*	Demuestran que métodos de aprendizaje automático (árboles, bosques aleatorios, redes neuronales profundas) superan a la regresión lineal en la predicción de retornos.
Gu, Shihao and Kelly, Bryan and Xiu, Dacheng	2021	*Autoencoder Asset Pricing Models [35]*	Utilizan *autoencoders* para extraer factores latentes no lineales, mejorando la valoración de activos.
Giglio, Stefano and Kelly, Bryan and Xiu, Dacheng	2022	*Factor Models, Machine Learning, and Asset Pricing [32]*	Revisión de técnicas de ML y modelos de factores, mostrando cómo manejar alta dimensionalidad y mejorar predicciones en los mercados financieros.
Bagnara, Matteo	2024	*Asset Pricing and Machine Learning: A critical review [3]*	Análisis crítico de la intersección entre aprendizaje automático y valoración de activos, destacando riesgos y oportunidades.

- **Basadas en medidas de riesgo:** Miden distintos aspectos del riesgo de un activo, como la volatilidad total de los retornos, la volatilidad idiosincrática o la sensibilidad al mercado.
- **Tasas de evaluación y señales fundamentales:** Incluyen métricas como ganancias-precio, ventas-precio, crecimiento de activos o incrementos en las ganancias.

Estas características cubren una amplia gama de factores que pueden influir en la rentabilidad futura de los activos y permiten capturar relaciones en el mercado.

La aplicación de modelos de aprendizaje automático más recientes en economía [35, 36] han sido usados para predecir el retorno de activos financieros, usando características como las mencionadas anteriormente, sin embargo, no se han encontrado resultados que puedan explicar el mercado financiero y tampoco se ha explorado la metodología de aplicar clasificación a los retornos siendo una propuesta novedosa dentro de la economía e inteligencia artificial.

A partir de lo anterior, se propone un enfoque basado en **algoritmos supervisados de clasificación para el análisis de activos financieros**, con el objetivo de ofrecer una **perspectiva alternativa** a los modelos de predicción de retornos continuos. Este cambio metodológico permite replantear el problema desde una óptica categórica, donde la variable objetivo, el retorno del activo, es transformada en clases discretas, facilitando así la aplicación de técnicas de clasificación. Para ello, se emplea el criterio propuesto por Prado [67], quien sugiere un esquema de etiquetado específico para series de tiempo financieras, lo cual permite capturar patrones de comportamiento relevantes en los retornos de los activos sin depender de su valor exacto y se acomoda al mismo problema a resolver en esta investigación que es el etiquetado del retorno de los activos financieros.

No obstante, **categorizar únicamente la variable objetivo** no siempre es suficiente para mejorar el desempeño de los algoritmos de clasificación, especialmente cuando éstos se enfrentan a un gran número de variables continuas susceptibles a ruido y valores atípicos [21]. Al respecto, Dougherty et al. [21] señalan que **transformar las características de tipo continuo a categorías** (mediante discretización supervisada o no supervisada) puede mejorar la precisión de clasificadores. Esto se debe a que:

1. Se reduce el impacto del ruido, ya que los valores atípicos quedan absorbidos en los intervalos discretos.
2. Disminuye la complejidad del espacio de búsqueda, favoreciendo que el modelo generalice mejor.
3. Se simplifica la representación de los datos, haciendo más manejables los cálculos para cada clasificador.

Por estas razones, en la presente investigación se propone **aplicar distintas discretizaciones** a las características financieras, con el fin de obtener mejores métricas en la clasificación de activos, en comparación con la utilización de la base de datos sin transformar.
Además, utilizar distintos métodos de selección de características para mejorar el tiempo de ejecución y la descripción de las métricas.

Estas características financieras no se restringen únicamente a la información de mercado, como precios, retornos o volatilidades, que se asocian con **variables especulativas**, sino que

también incorporan **variables fundamentales**, en concordancia con la base de datos utilizada en los trabajos de Gu et al. [35, 36]. Dentro de estas últimas se incluyen indicadores de rentabilidad, como el *return on equity* y el *return on assets*; razones de valoración, como *earnings-to-price* y *sales-to-price*; además de otras proporciones contables. La integración de estas métricas permite que el modelo contemple tanto señales derivadas de fundamentos económicos sólidos como factores de mercado de naturaleza más volátil.

1.2. Preguntas e Hipótesis de investigación

En esta investigación, se busca explorar el uso de discretización de las características financieras para mejorar los resultados de las métricas de clasificación de los retornos de los activos financieros utilizando algoritmos supervisados de distintas familias, así como la mejora del tiempo de ejecución y resultados aplicando métodos de selección de características.
Las preguntas de investigación que guían este trabajo son:

Categorización y Discretización:

1. ¿Qué ventajas aporta la categorización de la variable objetivo "Retorno" en la descripción de activos financieros?

2. ¿Cuáles son las ventajas de utilizar métodos de discretización, como K-Means y MDLP, frente la alta dimensionalidad de los datos y sus relaciones con el retorno?

Modelos de Clasificación, Comparación y Selección de Características:

1. ¿Qué ventajas se obtienen al enfocar el problema como un modelo de clasificación en lugar de un modelo de regresión?

2. ¿Qué diferencias se observan entre la selección de características basada en la teoría, en Redes Bayesianas y en Análisis de Componentes Principales, en términos de las métricas de evaluación y tiempo de ejecución de los modelos?

3. ¿Qué tan rápidos en tiempo de ejecución y eficientes en la mejora de las métricas de evaluación son los algoritmos de clasificación con características discretizadas para la descripción de activos financieros en comparación con los que no tienen una discretización previa?

Hipótesis:

- Aplicar discretizaciones como K-Means y MDLP a las características de los activos financieros mejorará las métricas de *accuracy, precision, F1-score y recall* para clasificación en comparación con el uso de características sin transformar.

- La selección de características relevantes reportadas en los artículos de Gu et al. [36, 35] y Kelly et al. [47], redes bayesianas y análisis de componentes principales reducirá el tiempo de ejecución y mejorará las métricas de clasificación.

1.3. Objetivo General

Desarrollar un enfoque integral aplicando discretizaciones a las características de los activos financieros para clasificar los retornos mediante algoritmos supervisados de distintas familias. Asimismo, comparar los resultados con la clasificación de retornos sin la transformación de características y evaluar cómo la selección de características relevantes (basada en la teoría, redes bayesianas y análisis de componentes principales) influye en el tiempo de ejecución y la mejora en las métricas de clasificación.

1.4. Objetivos Específicos

- Revisar la bibliografía y el estado del arte sobre el uso de aprendizaje supervisado, categorización y discretización en la clasificación de variables continuas en distintas bases de datos.
- Revisar e investigar la bibliografía sobre las discretizaciones K-Means y MDLP frente la alta dimensionalidad de bases de datos, y sus relaciones con la variable objetivo.
- Realizar una exploración, análisis y preprocesamiento adecuados de la base de datos de activos financieros (31 de enero de 2001 al 31 de diciembre de 2020) utilizada en Gu et al. [36] y [35], asegurando su preparación para los algoritmos de discretización.
- Crear dos subconjuntos de la base de datos que expliquen las mejores y peores mil empresas con respecto a la característica que representa el tamaño (mvel1) siguiendo el enfoque de Gu et al. [36].
- Buscar diferentes configuraciones de los hiperparámetros de los métodos de discretización a emplear obteniendo una óptima calidad de datos y etiquetas.
- Revisar y analizar los datos discretizados confirmando su preparación para los algoritmos de clasificación.
- Configurar los hiperparámetros de los modelos de clasificación evitando el sobreajuste y bajo desempeño en las métricas.
- Evaluar la rapidez en tiempo de ejecución y eficiencia en la mejora de las métricas de los algoritmos supervisados de clasificación frente a diferentes configuraciones de datos y variaciones en su calidad.
- Analizar las ventajas de tratar la clasificación de activos financieros frente a un enfoque de modelos de regresión.
- Evaluar la efectividad de los métodos de selección de características, específicamente redes bayesianas y análisis de componentes principales, mediante la comparación teórica de sus fundamentos, el impacto en las métricas de clasificación y los tiempos de ejecución de cada modelo.

1.5. Alcances y Limitaciones

Alcances:

- **Innovación metodológica**: El tratar este problema como clasificación y no predicción, representa una nueva contribución para el análisis de datos financieros. Agregando que se aplica una categorización y discretización, lo cual abre nuevas vías en la representación de los datos.
- **Aplicación práctica**: Los resultados obtenidos son más claros que las métricas obtenidas con predicción, siendo más sencillo de interpretar y sin la necesidad de aplicar otro tipo de análisis a los resultados.
- **Cobertura de datos**: Se puede variar el intervalo de años para poder conseguir una interpretación más específica según un evento histórico, una firma en particular, entre otros.
- **Análisis de múltiples técnicas**: Se explora diversos métodos de discretización de características, diferentes familias de técnicas de Inteligencia Artificial para la clasificación y varias métricas para la evaluación de los resultados, lo que amplía la aplicabilidad de los resultados en diferentes escenarios y configuraciones.

Limitaciones:

- **Alta dimensionalidad de los datos**: El manejo de una gran cantidad de datos es muy costoso computacionalmente, el tiempo de ejecución y los recursos computacionales podrían ser un obstáculo en este trabajo.
- **Riesgo al sobreajuste**: Dada la complejidad de la base de datos y en adición con la complejidad de los modelos a usar, existe la posibilidad de que los algoritmos supervisados ajusten demasiado los datos de entrenamiento, reduciendo su capacidad de generalización.
- **Interpretación limitada por la metodología**: Al aplicar transformaciones a los datos, categorización y discretización, se puede perder información importante que influya en los patrones de los datos sin transformación.
- **Generalización a otros mercados o tipos de activos financieros**: La metodología propuesta estará sujeta únicamente a los activos que se están analizando, e inclusive las características que contenga la base de datos. Por lo tanto, si se quiere extender a otra base de datos, se tendría que ajustar la metodología.
- **Tratar la base de datos como estática**: Este metodología no necesariamente reflejará el comportamiento del mercado en tiempo real, lo cual limitaría su aplicabilidad a modelos dinámicos.

1.6. Metodología

La minería de datos es el proceso de descubrir patrones, tendencias y relaciones significativas en grandes conjuntos de datos. Una de las metodologías más reconocidas y ampliamente utilizadas en la minería de datos es CRISP-DM (Cross-Industry Standard Process for Data Mining) [74]. Se decidió seguir esta metodología para crear un ciclo personalizado y elaborar esta investigación complementándola con los puntos relevantes de Ciencia de Datos. La metodología se representa en la Tabla (1.5).

Tabla 1.5: Fases y actividades del proceso de minería de datos

Fase	**Actividades**
Definición del problema	Identificación de preguntas clave. Establecimiento de métricas de éxito.
Comprensión del negocio	Definición de los objetivos del negocio. Evaluación de la situación actual. Determinación de los objetivos de minería de datos. Planificación del proyecto.
Comprensión de los datos	Recopilación de datos iniciales. Descripción de los datos. Exploración de los datos. Verificación de la calidad de los datos.
Preparación de datos	Selección de los datos. Limpieza de los datos. Construcción de los datos. Integración de los datos. Formateo de los datos.
Ingeniería de características	Codificación de variables categóricas. Discretización de características.
Modelado predictivo	Selección de técnicas de modelado. Generación del diseño de pruebas. Construcción del modelo. Evaluación del modelo.
Evaluación	Evaluación de resultados. Revisión del proceso. Determinación de los próximos pasos.

Se puede apreciar el flujo de la metodología con el siguiente diagrama (1.1).

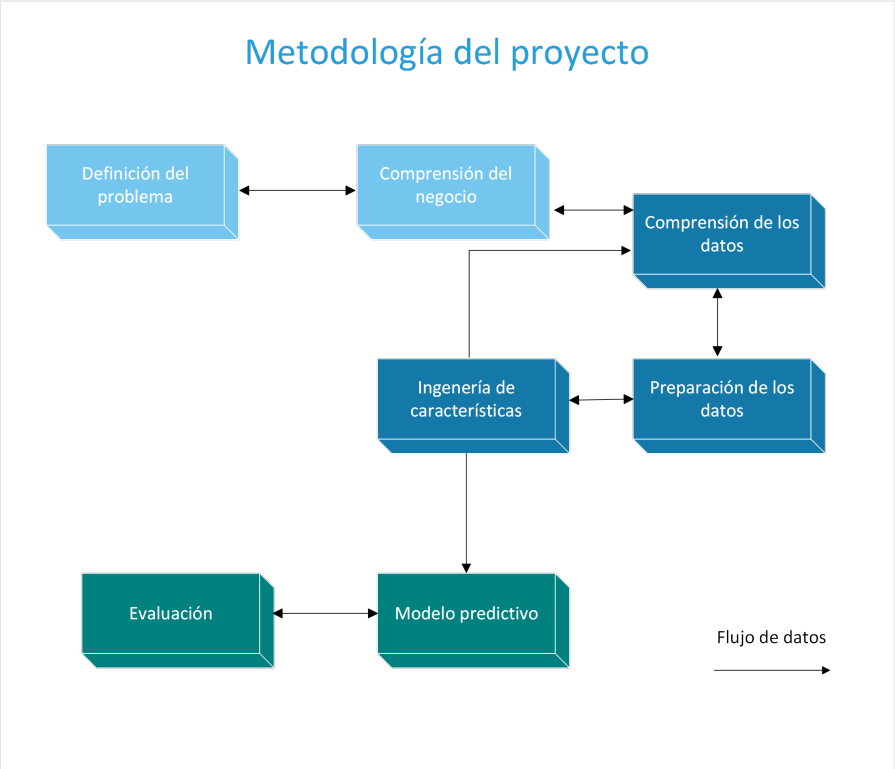


Figura 1.1: Pasos de la metodología personalizada. Fuente: Elaboración propia.

Técnicas y herramientas a emplear

El proyecto se desarrollará principalmente en el lenguaje de programación Python [77], a través del IDE de Jupyter Notebook. Para exploración, transformación y modelación de los datos, se utilizará modulos pertenecientes del lenguaje de programación Python, como: para visualización se usará Matplotlib [40] y Seaborn [79], para herramientas de análisis estadístico y herramientas para la base de datos se usará NumPy [37] y Pandas [58], para la discretización se implementará un código en Python y se usará Scikit-learn [65], para el modelado se usará Scikit-learn y TensorFlow [65, 1], y por último, para aplicar las pruebas estadísticas se usará Scipy [78].

Equipo de cómputo a utilizar

El equipo de cómputo utilizado es una laptop de la marca ASUS. Cuenta con un procesador Ryzen 7 de la serie 6000, compuesto por 8 núcleos que operan a una velocidad de 3.02 GHz cada uno. Dispone de una tarjeta gráfica Nvidia GeForce RTX 2050 con 4 GB de memoria dedicada y 7.6 GB de memoria compartida. Además, incorpora una memoria RAM DDR5 de 16 GB que funciona a una velocidad de 4800 MHz.

1.7. Contribuciones de la Investigación

Este trabajo propone un enfoque innovador en la valoración de activos financieros al tratar el problema como una tarea de clasificación supervisada en lugar de predicción directa de retornos.

Las principales contribuciones de esta investigación se pueden resumir en los siguientes puntos:

Cambio de metodología: Se plantea una reformulación del problema de predicción de retornos como un problema de clasificación, empleando categorización del retorno como variable objetivo. Este cambio permite explorar nuevas métricas de evaluación y enfoques metodológicos dentro de la inteligencia artificial aplicada a finanzas.

Evaluación del impacto de la discretización: Se comparan dos métodos de discretización, K-Means y MDLP, aplicados a las características predictoras, midiendo su efecto en el rendimiento de múltiples clasificadores. Esto permite identificar configuraciones que mejoran la generalización y estabilidad del modelo frente a datos ruidosos o con valores atípicos.

Análisis empírico del valor de la transformación categórica: A través de un conjunto de experimentos, se demuestra que la transformación de variables continuas a categóricas puede mejorar significativamente el desempeño de clasificadores supervisados, tanto en precisión como en tiempo de ejecución.

Comparación entre métodos de selección de características: Se evalúan tres estrategias para reducir la dimensionalidad: selección basada en conocimiento teórico, aprendizaje de estructuras con Redes Bayesianas y análisis mediante componentes principales. Se analiza su impacto no sólo en métricas de evaluación, sino también en eficiencia computacional en términos de tiempo de ejecución.

Desarrollo de una metodología replicable: La investigación adopta un enfoque basado en CRISP-DM adaptado al contexto financiero, integrando discretización, clasificación, selección de características y evaluación de modelos. Esto genera una propuesta metodológica que puede ser replicada y adaptada a otros dominios de datos económicos.

Uso de una base de datos real e histórica: Se trabaja con una base de datos de más de 1.6 millones de observaciones y 94 características provenientes del entorno financiero real (usada en [36, 35]), asegurando que los hallazgos tengan relevancia práctica y aplicabilidad a contextos reales de inversión.

Estas contribuciones buscan avanzar y desarrollar otro enfoque en la intersección entre finanzas e inteligencia artificial, proponiendo soluciones innovadoras y prácticas a los retos del análisis de activos financieros en contextos de alta dimensionalidad y complejidad

Capítulo 2

Marco Teórico

La comprensión del riesgo y retorno en los mercados financieros ha sido una de las preocupaciones centrales tanto de la teoría económica como de la práctica de inversión. A lo largo del tiempo, diversos enfoques han intentado modelar esta relación, partiendo desde formulaciones lineales que buscan explicar el retorno a través de un número reducido de factores sistemáticos. No obstante, las inconsistencias empíricas y la creciente complejidad de los datos financieros han evidenciado las limitaciones de dichos modelos. En respuesta, han emergido nuevas propuestas que incorporan estructuras multifactoriales, intertemporalidad y, más recientemente, técnicas de aprendizaje automático que permiten capturar relaciones no lineales y de alta dimensionalidad. Este recorrido teórico no solo refleja la evolución del pensamiento financiero, sino que sustenta el enfoque metodológico adoptado en esta investigación, el cual plantea una reconfiguración del problema de predicción de retornos bajo un paradigma de clasificación supervisada.

2.1. Modelos de Valoración de Activos Financieros

Teoría Moderna de Carteras

Las **Carteras o Portafolios** en el ámbito económico se refiere a un conjunto de distintos activos financieros, Markowitz (1952) [55] introduce el concepto de diversificación eficiente con el objetivo de maximizar el retorno ajustado al riesgo, sentando así las bases de la teoría moderna de carteras. Su enfoque se centra en la relación entre el retorno esperado y la varianza de los retornos, considerando esta última como una medida del riesgo. En contraposición al enfoque tradicional que busca maximizar el valor descontado de los retornos futuros —el cual, según Markowitz, no promueve la diversificación—, propone que los inversionistas deberían buscar maximizar el retorno esperado dado un nivel de riesgo o, alternativamente, minimizar el riesgo para un nivel dado de retorno esperado.

Estas ideas se formalizan mediante dos expresiones fundamentales. En primer lugar, el retorno esperado de una cartera se expresa como:

$$\mathrm{E}(R_p) = \sum_{i=1}^{N} X_i E(R_i), \tag{2.1}$$

donde $E(R_p)$ representa el retorno esperado de la cartera, $E(R_i)$ el retorno esperado del activo i, X_i la proporción del total de la cartera invertida en el activo i, y N el número total de activos considerados.

En segundo lugar, el riesgo de la cartera, medido como varianza de sus retornos, se calcula mediante:

$$\sigma_p^2 = \sum_{i=1}^{N} \sum_{j=1}^{N} X_i X_j \sigma_{ij}, \tag{2.2}$$

donde σ_p^2 es la varianza de los retornos de la cartera, σ_{ij} la covarianza entre los retornos de los activos i y j, y X_i, X_j las proporciones invertidas en dichos activos.

Además, Markowitz introduce el concepto de frontera eficiente, la cual representa el conjunto de carteras que, para cada nivel de riesgo, maximizan el retorno esperado o, inversamente, minimizan el riesgo para un nivel deseado de retorno. Esta frontera proporciona una base teórica sólida para la selección óptima de carteras diversificadas.

Modelo de Valoración de Activos de Capital (CAPM)

Sharpe (1964) [73] y Lintner (1965) [52] desarrollan el CAPM, que extiende la teoría de carteras de Markowitz hacia la valoración de activos en equilibrio de mercado. El CAPM establece una relación lineal entre el retorno esperado de un activo y su riesgo sistemático, medido por el coeficiente beta (β), que representa la sensibilidad del retorno del activo a los movimientos del mercado. La ecuación fundamental del CAPM es:

$$E(R_i) = R_f + \beta_i \left(E(R_m) - R_f \right), \tag{2.3}$$

donde $E(R_i)$ es el retorno esperado del activo i mismo de la ecuación (2.1), R_f es la tasa libre de riesgo, β_i es la beta del activo i, que mide la sensibilidad del retorno de activo i a los movimientos del mercado, $E(R_m)$ es el retorno esperado del mercado y $E(R_m) - R_f$ es la prima de riesgo del mercado

Los autores proponen que los inversores sean compensados únicamente por el riesgo sistemático, ya que el riesgo específico de cada activo puede eliminarse mediante diversificación.

Pruebas Empíricas del CAPM

Fama y MacBeth (1973) [26] evalúan el modelo de valoración de activos financieros (CAPM) mediante pruebas empíricas para determinar la relación entre el riesgo y el retorno esperado en los mercados financieros. Utilizan un modelo de dos parámetros (media y varianza), basado en la teoría de carteras de Markowitz, y plantean tres implicaciones fundamentales: en primer lugar, la relación entre el retorno esperado y el riesgo sistemático, representado por β, es lineal; en segundo lugar, β constituye una medida adecuada del riesgo en un portafolio eficiente; y finalmente, existe una relación positiva entre el nivel de riesgo asumido y el retorno esperado.

Los resultados empíricos respaldan estas proposiciones, ya que el modelo de dos parámetros mostró ser consistente con el comportamiento observado de los inversionistas en el mercado de

valores de Nueva York (NYSE), evidenciando una relación aproximadamente lineal entre los valores de β y los retornos esperados.

Por otro lado, Fisher, Jensen y Scholes (1972) [9] realizan, también, pruebas empíricas sobre el CAPM, asumen una relación lineal entre el riesgo sistemático de un activo, medido por su beta β, y su rendimiento esperado. Sin embargo, sus resultados sugieren inconsistencias: los activos con alta beta tienden a generar retornos más bajos de lo predicho, mientras que aquellos con beta baja generan retornos superiores a los esperados. Estas desviaciones indican que el CAPM no proporciona una descripción precisa del comportamiento de los retornos de los activos.

Para abordar estas deficiencias, los autores proponen un modelo de dos factores, que incluye un segundo componente además del mercado, para mejorar la explicación de los retornos:

$$E(R_i) = (1 - \beta_i)E(Z_r) + \beta_i E(R_m), \tag{2.4}$$

donde $E(Z_r)$ es el retorno esperado de un portafolio con beta cero, $E(R_m)$ es el retorno esperado del mercado.

Limitaciones de los Modelos Lineales en los Activos Financieros

A lo largo de las décadas, los modelos lineales han dominado la teoría de la valoración de activos, especialmente a través del CAPM. Sin embargo, las pruebas empíricas y los desarrollos teóricos han señalado limitaciones significativas en la capacidad de estos modelos para capturar la complejidad del comportamiento de los retornos de los activos.

Modelo Intertemporal de Valoración de Activos (ICAPM)

Merton (1973) [59] critica al CAPM clásico por su enfoque estático de un solo período, proponiendo en cambio el ICAPM, que incorpora un horizonte temporal múltiple. En el ICAPM, los inversionistas optimizan sus decisiones de inversión a lo largo del tiempo, considerando no sólo el riesgo actual, sino también las oportunidades futuras de inversión.

El modelo se diferencia del CAPM al introducir un "factor de cobertura" que refleja la capacidad de un activo para actuar como protección frente a cambios en las oportunidades de inversión. La ecuación de equilibrio para los retornos esperados en el ICAPM es:

$$E(R_i) - R_f = \beta_i(E(R_m) - R_f) + \text{Factor de cobertura} \tag{2.5}$$

Este enfoque permite una modelización más realista del comportamiento del mercado, ya que los inversionistas ajustan sus portafolios no sólo en función del riesgo sistemático, sino también en respuesta a cambios en el entorno de inversión.

Teoría de Arbitraje de Valoración de Activos (APT)

Ross (1976) [70] desarrolla la APT como alternativa al CAPM, argumentando que este último depende de suposiciones restrictivas difíciles de justificar empíricamente, como la norma-

lidad de los retornos. La APT propone que los retornos de los activos están determinados por múltiples factores comunes, no solo el riesgo de mercado. La ecuación general del modelo es:

$$R_i = E(R_i) + \beta_{i1}F_1 + \beta_{i2}F_2 + ... + +\beta_{ik}F_k + \varepsilon_i, \tag{2.6}$$

donde βs son las sensibilidades del activo a los factores de riesgo comunes de F_k y ε_i es el componente de retorno idiosincrático, que se espera que sea independiente de los factores comunes.

Efecto Tamaño

Banz (1980) [5] documenta una anomalía en la relación lineal postulada por el CAPM: el "efecto tamaño", muestra que las empresas pequeñas tienden a ofrecer mayores retornos ajustados por riesgo que las grandes. Al incluir el tamaño como factor adicional en una versión extendida del CAPM:

$$E(R_i) = E(Z_r) + \gamma_1\beta_i + \gamma_2\left(\frac{M_i - M_m}{M_m}\right), \tag{2.7}$$

donde $E(Z_r)$ es el retorno esperado de un portafolio de beta cero, γ_1 es la prima de riesgo del mercado, γ_2 es la constante que mide la contribución del tamaño M_i (valor del mercado) al retorno esperado, M_i es el valor del mercado de la empresa i y M_m Valor del mercado promedio de todas las empresas.

Modelo de Tres Factores

Fama y French (1993) [24] proponen un modelo de tres factores, extendiendo el CAPM para incluir, además del mercado, dos factores adicionales: el tamaño de la empresa y la proporción del valor contable sobre valor de mercado (BE/ME). Argumentan que estos factores capturan mejor la variación de los retornos que el CAPM tradicional. La ecuación del modelo es:

$$R_i - R_f = \alpha_i + \beta_m(R_m - R_f) + s_i \cdot SMB + h_i \cdot HML + \varepsilon_i, \tag{2.8}$$

donde R_i es el retorno de la cartera i, R_f es la tasa libre de riesgo, R_m es el retorno del mercado, α_i es el intercepto del modelo para el activo i, β_m es la sensibilidad de la cartera al factor mercado, *SMB* (Small Minus Big) es el factor de tamaño que captura la diferencia entre los retornos de empresas pequeñas y grandes, *HML* (High Minus Low) es el factor de valor que captura la diferencia entre las empresas con alto valor contable a valor del mercado y aquellos con bajo valor del mercado, s_i es la sensibilidad de *SMB*, h_i es la sensibilidad de *HML*, ε_i es el término de error específico para la cartera i. El modelo de Fama y French muestra que el riesgo sistemático no es la única fuente de variación en los retornos, lo que cuestiona los modelos lineales.

Los resultados de esta subsección sugieren que los modelos lineales tradicionales, como el CAPM, tienen limitaciones importantes. Estos modelos no logran capturar completamente la complejidad de los mercados financieros, donde múltiples factores interactúan de manera no

lineal. La evidencia muestra que incorporar más factores o adoptar enfoques intertemporales y multifactoriales mejora significativamente la precisión en la explicación de los retornos de los activos. Por lo tanto, se hace evidente la necesidad de desarrollar modelos más flexibles que vayan más allá de las relaciones lineales simples.

Evolución Hacia Técnicas de Aprendizaje Máquina

La evolución de los modelos de valoración de activos ha sido impulsada por la necesidad de superar las limitaciones de los enfoques lineales, como el CAPM. Recientes desarrollos han introducido técnicas más complejas que buscan capturar la naturaleza de los retornos de los activos.

Selección de Carteras Basada en un Enfoque Bayesiano

Pástor (2000) [64] propone un enfoque bayesiano para la selección de carteras que combina modelos de valoración de activos con evidencia empírica. Utiliza la teoría bayesiana para actualizar las creencias previas del inversionista sobre los retornos de los activos en función de la evidencia de los datos observados. La ecuación central del modelo describe la evolución de la riqueza del inversor:

$$W_{t+1} = W_t \left(1 + r_f(1-\delta)w^T r_{t+1}\right), \tag{2.9}$$

donde W_t es la riqueza en el tiempo t, r_f es el retorno del activo sin riesgo, w son los pesos de la cartera, r_{t+1} son los retornos de los activos en exceso sobre el activo sin riesgo, δ es la proporción de la riqueza invertida en el activo libre de riesgo. En esta ecuación, el inversor maximiza la utilidad esperada de su riqueza futura.
La introducción del enfoque bayesiano en la selección de carteras permite incorporar la incertidumbre en el modelo, lo que mejora la toma de decisiones bajo condiciones de riesgo.

Filtro de Regresión en Tres Pasos (3PRF)

El Three-Pass Regression Filter (3PRF) [46] aborda el problema de sobreajuste al manejar grandes cantidades de predictores en economía moderna. La técnica se basa en tres pasos de regresión de Mínimos Cuadrados Ordinarios (OLS) para extraer los factores relevantes para la predicción, ignorando aquellos que no aportan información significativa.

1. Regresión de predictores sobre indicadores (2.10) para estimar la relación de los predictores con los factores relevantes.

$$x_{i,t} = \tilde{\phi}_{0,i} + z_t^{'}\tilde{\phi}_i + \tilde{\varepsilon}_{i,t} \tag{2.10}$$

   donde $x_{i,t}$ es el valor del predictor i en el tiempo t (por ejemplo, una variable macroeconómica relevante), $\tilde{\phi}_{0,i}$ es el intercepto de la regresión para el predictor i, z_t' es el vector de características observables en el tiempo t (por ejemplo, tasas, índices, proporciones), $\tilde{\phi}_i$ es el vector de coeficientes de regresión para el predictor i y $\tilde{\varepsilon}_{i,t}$ es el término de error del predictor.

2. Regresión transversal de los predictores sobre las estimaciones del primer paso (2.11) para determinar los factores que afectan los predictores.

$$x_{i,t} = \ddot{\phi}_{0,i} + \hat{\phi}_i^{'} \ddot{F}_t + \ddot{\varepsilon}_{i,t} \tag{2.11}$$

donde $x_{i,t}$ es el valor del predictor i en el tiempo t, $\ddot{\phi}_{0,i}$ es el intercepto de la regresión para el predictor i, $\hat{\phi}_i^{'}$ son las pendientes obtenidas del paso 1, F_t es el vector de factores comunes en el tiempo t y $\ddot{\varepsilon}_{i,t}$ es el término de error del predictor.

3. Regresión temporal del objetivo sobre los factores estimados para obtener la predicción (2.12).

$$y_{t+h} = \check{\beta}_0 + \hat{F}_t' \check{\beta} + \check{\eta}_{t+h} \tag{2.12}$$

donde y_{t+h} es la variable objetivo a predecir en el tiempo $t+1$ con pasos h adelantes, $\check{\beta}_0$ es el intercepto estimado, $\hat{F}_t'$ son los factores estimados en el paso anterior, $\check{\beta}$ es el vector de coeficientes que mide el impacto de cada factor y $\check{\eta}_{t+h}$ es el término de error de predicción.

Predicción de Valores Presentes y Expectativas del Mercado

Kelly y Pruitt (2012) [45] utilizan la técnica de *Partial Least Squares* (PLS) para abordar la alta dimensionalidad de los predictores en la predicción de retornos del mercado y crecimiento de dividendos. El enfoque se basa en modelar las proporciones de book-to-market como una combinación de factores latentes (factores no observados que explican el comportamiento de un conjunto de activos financieros):

$$v_{i,t} = \phi_{i,0} + \phi_i' F_t + \varepsilon_{i,t}, \tag{2.13}$$

donde: $v_{i,t}$ es el logaritmo de la proporción precio-dividendo de la cartera i en el tiempo t, F_t son los factores latentes aproximados que los autores tratan de identificar, $\varepsilon_{i,t}$ es el error del modelo, $\phi_{i,0}$ es el intercepto de la ecuación que capta la componente constante en la relación entre la proporción precio-dividendo y factores que lo afectan, ϕ_i' es el vector de coeficientes que mide la sensibilidad de la proporción precio-dividendo de la cartera i a los factores latentes F_t. El modelo PLS se utiliza para seleccionar los factores relevantes, mejorando la precisión de las predicciones en comparación con otros enfoques.

En los resultados de su modelo, se logró un buen R^2 fuera de muestra del (13.1 %) para los retornos anuales [45].

Modelo de Cinco Factores

Fama y French (2014) [23] ampliaron su modelo original de tres factores [24] incorporando dos factores adicionales: rentabilidad e inversión, con el objetivo de mejorar la capacidad explicativa de los retornos accionarios. El factor *RMW* (Robust Minus Weak) representa la diferencia entre los retornos de carteras formadas por empresas con alta rentabilidad operativa frente a aquellas con baja rentabilidad. Por su parte, el factor *CMA* (Conservative Minus Aggressive)

mide la diferencia entre los retornos de empresas que invierten de forma conservadora (poco) y aquellas que invierten de forma agresiva (mucho).

Este modelo supone que las empresas pequeñas tienden a generar mayores retornos en promedio que las grandes, que las empresas con altas proporciones de book-to-market (valor) deberían superar en rendimiento a las de baja proporción (crecimiento), que las firmas con alta rentabilidad operativa presentan retornos superiores a las menos rentables, y que las empresas con políticas de inversión conservadoras ofrecen retornos más altos que aquellas con estrategias agresivas.

El modelo de cinco factores ofrece una mejor descripción de los retornos promedio en comparación con el modelo de tres factores. No obstante, el factor *HML* del modelo de tres factores [24] pierde relevancia al volverse redundante en presencia de los factores de rentabilidad e inversión, lo que sugiere que estos capturan parte de la variabilidad previamente explicada por *HML*.

Uso de IPCA Para la Valoración de Activos

Kelly, Pruitt y Su (2018) [47] introducen el uso de *Instrumented Principal Component Analysis* (IPCA) que permite que las cargas dinámicas (sensibilidades a los factores latentes) dependan de características observables de los activos, como tamaño, momentum o book-to-market. Esto permite identificar factores latentes y cómo varían las sensibilidades de los activos a lo largo del tiempo.

La fórmula fundamental del modelo IPCA es:

$$r_{i,t+1} = \alpha_{i,t} + \beta_{i,t} f_{t+1} + \varepsilon_{i,t+1}, \tag{2.14}$$

donde $r_{i,t+1}$ es el retorno en exceso del activo i en el tiempo $t+1$, $\alpha_{i,t}$ es el intercepto del modelo para el activo i en el tiempo t, $\beta_{i,t}$ representan las cargas dinámicas o sensibilidades a los factores latentes f_{t+1}, f_{t+1} son los factores latentes comunes no observables que influyen en los retornos y $\varepsilon_{i,t+1}$ es el término de error que captura la parte del retorno no explicada por los factores latentes.

En esta investigación, aplicaron IPCA a más de 12,000 acciones de 1962 a 2014, con un conjunto de datos de 1.4 millones de observaciones mensuales. Las características incluyen 36 factores conocidos en finanzas, como beta del mercado, book-to-market, momentum y volatilidad idiosincrática.

El modelo IPCA explica un 18.6 % de la variación total en los retornos de las acciones. En términos de predicción, el R^2 predictivo del IPCA es de 0.7 %.

Enfoque del Factor de Descuento Estocástico (SDF)

Kozak, Nagel y Santosh (2018) [51] propusieron un modelo bayesiano para construir un SDF, utilizando técnicas de regularización como Ridge y Elastic Net. Estas técnicas imponen penalizaciones sobre los coeficientes (sensibilidades de los factores), lo cual permite manejar la alta dimensionalidad del problema y reducir el sobreajuste. La penalización dual (L1 y L2) contribuye tanto a la selección de predictores relevantes como al ajuste del modelo, con el objetivo de minimizar errores y maximizar el R^2 fuera de muestra.

El conjunto de datos empleado incluyó 50 carteras que abarcan anomalías conocidas, como la proporción precio-ganancias y el book-to-market; también se incorporaron proporciones financieras extraídos de la base de datos WRDS (Wharton Research Data Services) [51, 80], considerando 80 factores basados en características como capitalización, liquidez, solvencia y rentabilidad, cubriendo el período de 1964 a 2017. Además, se utilizaron datos adicionales generados mediante interacciones no lineales y potencias de las características originales para ampliar el conjunto de predictores.

El modelo SDF propuesto, al integrar regularización para tratar la alta dimensionalidad, demuestra un mejor desempeño fuera de muestra en comparación con los modelos tradicionales de pocos factores.

Autoencoders Como Un Método Para Capturar Relaciones No Lineales

Gu, Kelly y Xiu (2020) [35] proponen un modelo basado en autoencoders, una red neuronal no supervisada que permite identificar factores del mercado financiero. Los autoencoders capturan relaciones entre las características de los activos y los factores de riesgo.

En esta investigación usaron datos mensuales de acciones desde el CRSP (Center of Research Security Prices), cubriendo casi 30,000 acciones entre 1957 y 2016. Lo anterior incluye 94 características predictivas, que abarcan tamaños, valoraciones, momentum, volatilidades, entre otras. Los datos se preprocesan para normalizar las características y evitar sesgos temporales.

Los resultados del modelo autoencoder logró un R^2 total de 14.3 % . El R^2 predictivo es de 0.58 %.

Comparación de Técnicas de Aprendizaje Automático

De nueva cuenta, Gu, Kelly y Xiu (2020) [36] comparan varios métodos de aprendizaje estadístico, como Elastic Net, árboles de decisión (random forests y gradient boosting), redes neuronales y reducción de dimensionalidad (PCR y PLS), para mejorar la predicción de los retornos de activos. Se emplean técnicas de regularización y ensambles para evitar el sobreajuste y mejorar la estabilidad del modelo.

Los datos que usaron fueron los mismos que en su otro artículo [35], pero agregaron 8 predictores macroeconómicos y las interacciones con las 94 características con estos predictores, y consideraron 74 variables dummy sectoriales, dando un total de 920 señales iniciales. Se preprocesaron los datos para ajustar las características y reducir la variabilidad.

Sus mejores resultados fueron con técnicas no lineales, como los árboles de decisión y las redes neuronales, con un R^2 predictivo a nivel de acciones anuales de entre 3.60 % para una red neuronal de 4-capas y 3.28 % para los árboles de regresión, para el R^2 mensual se obtuvo 0.40 % para una red neuronal de 3-capas y 0.34 % para árboles de regresión. Las variables clave fueron las tendencias de precios, la liquidez y la volatilidad.

Los avances revisados en la sección anterior confirman que los modelos enfocados en técnicas de aprendizaje máquina logran capturar parte de la complejidad inherente a los retornos financieros; sin embargo, también ponen en evidencia dos desafíos que persisten al trabajar con bases tan amplias como la de 94-920 características continuas empleadas por Gu, Kelly y Xiu [36, 35]:

1. la **alta dimensionalidad**, que dificulta la interpretación y favorece el sobreajuste, y
2. la **naturaleza continua** de la mayoría de las características, que impide aprovechar directamente clasificadores basados en reglas, árboles o redes bayesianas que operan con variables categóricas.

Una vía históricamente efectiva para mitigar ambos problemas consiste en transformar los predictores numéricos en variables discretas, reduciendo ruido, mejorando la interpretabilidad e, incluso, acelerando el entrenamiento de los modelos [21].

2.2. Discretización

En tareas de clasificación supervisada, muchos algoritmos, como las redes bayesianas, los árboles de decisión o las reglas de asociación, suelen operar sobre características categóricas o discretas [21]. Sin embargo, los conjuntos de datos del mundo real a menudo contienen variables continuas que deben ser transformadas antes de que estos algoritmos puedan aplicarse de manera efectiva [14, 66]. La discretización es el proceso de dividir el dominio de una característica continua en un número finito de intervalos o categorías [30]. Esta transformación puede realizarse de forma supervisada, incorporando la variable objetivo, o de manera no supervisada. Convertir variables numéricas en categóricas mejora la interpretabilidad del modelo, mitiga el ruido en los datos, facilita el uso de algoritmos que son más eficaces con entradas discretas y reduce la complejidad computacional [14, 30].

Discretizar características continuas no solo mejora la eficiencia y precisión de diversos algoritmos de clasificación, sino que también asegura la compatibilidad con métodos como CART, Naive Bayes y redes bayesianas [21]. Al agrupar valores similares, la discretización suaviza las variaciones menores que podrían introducir ruido en el proceso de clasificación, y al mismo tiempo produce modelos más interpretables y adecuados para la extracción de reglas [14, 21].

Los algoritmos de discretización pueden clasificarse de varias maneras [30, 53]. Pueden distinguirse según el número de variables consideradas, los métodos *univariados* tratan cada atributo por separado, mientras que los enfoques *multivariados* consideran interacciones entre características. Otra categorización se basa en el uso de la variable objetivo, diferenciando entre técnicas *supervisadas* y *no supervisadas*. Adicionalmente, los métodos pueden diferenciarse por su enfoque, ya sea *descendente (divisivo)* o *ascendente (aglomerativo)*, y por el momento de su aplicación, siendo los algoritmos *estáticos* (aplicados antes del modelado) o *dinámicos* (aplicados durante el entrenamiento).

Uno de los modelos de discretización más ampliamente utilizados es el algoritmo basado en el *Principio de Mínima Longitud de Descripción* (MDLP) [2, 27, 50], un algoritmo univariado, estático y supervisado cuya popularidad radica en su fundamento en la teoría de la información, lo que le permite identificar cortes que maximizan la ganancia de información al tiempo que controla el sobreajuste mediante un criterio de parada basado en la complejidad del modelo [27]. MDLP emplea medidas teóricas como la entropía y la información mutua, y ha demostrado un rendimiento sólido en estudios comparativos de discretización [2, 27, 50]. No obstante, a pesar de estos beneficios, MDLP se caracteriza por una alta complejidad computacional, lo cual

se vuelve particularmente problemático cuando se aplica a conjuntos de datos grandes o con muchas características [43].

Conociendo la naturaleza de los datos, su cantidad y sus datos únicos, un algoritmo como MDLP sigue siendo muy costoso e ineficiente en cuestión de su complejidad computacional [27], por lo cual, se busca otro algoritmo que sea menos complejo. Por ello, se propone el algoritmo "K-means" [54]. Es un método no supervisado de clustering, donde establece su etiqueta según la agrupación de cada característica tomando en cuenta la varianza mínima de ésta [54].

En este contexto, la elección de los algoritmos MDLP y K-means responde tanto a consideraciones teóricas como prácticas [27, 54]. MDLP permite incorporar la variable objetivo durante el proceso de discretización, maximizando la ganancia de información y mejorando la discriminación entre clases [27], lo cual es particularmente valioso cuando se busca optimizar la precisión del clasificador [2, 50]. Por su parte, K-means ofrece una alternativa eficiente y escalable, capaz de adaptarse a grandes volúmenes de datos sin comprometer significativamente el rendimiento [54]. Al aplicar ambos métodos y contrastar sus efectos, se busca identificar la estrategia de discretización que mejor se adapte a los desafíos de los problemas financieros reales, caracterizados por su alta dimensionalidad y la presencia de ruido estructural, consolidando así estas discretizaciones como opciones adecuadas y complementarias para el presente estudio de Clasificación de Activos Financieros.

Minimum Description Length Principle

MDLP es un método para dividir una característica continua en múltiples intervalos tomando en cuenta la variable objetivo puede resumirse de la siguiente manera [27]:

1. **Ordenar los valores de la característica continua tomando en cuenta la variable objetivo como un conjunto de pares.**

2. **Evaluar todos los puntos de corte**.
   Punto de corte candidato T: Es el punto medio entre valores sucesivos de la variable predictora en los datos de entrenamiento.

3. **Para cada punto de corte potencial, calcular la ganancia de información.** Esto implica:

   - **Entropía:** mide la incertidumbre o impureza de una partición de datos.

   $$H(D) = -\sum_{j=1}^{m} P_j \log_2(P_j), \tag{2.15}$$

   donde P_j es la probabilidad de la clase j en el conjunto de datos D. Valores altos de entropía indican una distribución de clases más uniforme, mientras que valores bajos indican mayor pureza.

   - **Ganancia de Información:** cuantifica la reducción en la entropía lograda al dividir los datos en un punto de corte específico.

   $$Gain(D,T) = H(D) - \sum_{i=1}^{k} \frac{|D_i|}{|D|} H(D_i), \tag{2.16}$$

donde $H(D)$ es la entropía del conjunto de datos original D, D_i representa el subconjunto de D resultante de la división, y $H(D_i)$ es la entropía de ese subconjunto.

Un punto de corte ideal produce particiones con menor entropía, lo que significa que los datos dentro de cada partición son más homogéneos. Si $Gain(D,T) = 0$, la división no mejora la organización de los datos y, por tanto, no es útil.

4. **Seleccionar el punto de corte que maximice la ganancia de información.**
5. **Aplicar recursivamente el proceso** a los subconjuntos resultantes hasta que se cumpla un criterio de parada.

El criterio utilizado para determinar cuándo cesar la discretización adicional es:

$$Gain(A,T) > \frac{\log_2(N-1)}{N} + \frac{\Delta(A,T)}{N}, \tag{2.17}$$

donde T es el punto de corte candidato, N es el número de instancias en T, y $\Delta(A,T)$ se define como:

$$\Delta(A,T) = \log_2(3^k - 2) - [kH(D) - k_1 H(D_1) - k_2 H(D_2)],$$

con k denotando el número de clases de la variable objetivo.

Este criterio indica cuándo una partición adicional no justifica el aumento en la complejidad del modelo. El término $\Delta(A,T)$ penaliza la complejidad en función del número de clases de la variable objetivo (k) y de las entropías de las particiones resultantes $H(D_1)$ y $H(D_2)$, asegurando que sólo se seleccionen aquellas divisiones que realmente mejoran la clasificación, evitando así divisiones innecesarias que podrían llevar al sobreajuste. Si la partición inducida por T introduce una estructura significativa en los datos que justifica la complejidad adicional, entonces $\Delta(A,T) > 0$. Por el contrario, si $\Delta(A,T) < 0$ o es muy pequeña, el modelo prefiere mantener una estructura más simple.

MDLP presenta ventajas claras frente a otros métodos [27, 30, 53], ya que utiliza medidas teóricas sólidas como la entropía y la información mutua, controla el sobreajuste mediante un criterio de parada riguroso, y ha demostrado un buen desempeño en estudios comparativos de discretización.

A pesar de sus ventajas, MDLP tiene una **alta complejidad computacional**, especialmente en conjuntos de datos grandes o con muchas características continuas.

K-means

El algoritmo K-means [54] es un método no supervisado utilizado para agrupar datos en k grupos distintos. Su objetivo es particionar un conjunto de datos en k grupos, minimizando la suma de las distancias al cuadrado de los puntos dentro de cada grupo hacia su centroide correspondiente. Esto lo convierte en una técnica eficiente para discretizar datos continuos de manera no supervisada.

Funcionamiento:

Inicialización: Se seleccionan k puntos, que servirán como los centroides iniciales de los grupos. Estos puntos pueden elegirse al azar o a partir de los primeros puntos de los datos.

Asignación de puntos a grupos: Cada punto de los datos se asigna al grupo cuyo centroide esté más cerca de él, usando la distancia euclidiana como criterio de proximidad.

- La partición mínima en distancia $S(x)$ de los datos en k grupos, se define como:

$$S(x) = \{S_1(x), S_2(x), ..., S_k(x)\}, \tag{2.18}$$

con

$$S_i(x) = \{z \in \mathbb{R}^N : ||z - x_i|| \leq ||z - x_j||, \forall j \neq i\}, \tag{2.19}$$

donde x_i es el centroide del grupo $S_i(x)$, mientras que x_j hace referencia al centroide de otro grupo y z es cualquier punto en el espacio N dimensional.

Recalcular los centroides: Una vez que todos los puntos se han asignado a los grupos, el centroide de cada grupo se recalcula como la media de los puntos en ese grupo. El nuevo centroide x_i del grupo S_i se define como

$$x_i = \frac{1}{|S_i|} \sum_{z \in S_i} z, \tag{2.20}$$

donde $|S_i|$ es el número de puntos en el grupo de S_i.

Optimización Usando Minimización de la Varianza: Minimiza la suma de las varianzas dentro de cada grupo minimizando la función de costo

$$W(s) = \sum_{i=1}^{k} \int_{S_i} ||z - x_i||^2 dp(z), \tag{2.21}$$

donde $W(s)$ es la varianza total dentro de los grupos, S_i son los grupos, x_i son los centroides de cada grupo y $dp(z)$ es la densidad de probabilidad de los datos.

Iteración: Este proceso de asignación y recalculación se repite hasta que los centroides no cambian significativamente o los puntos ya no se reasignan a diferentes grupos.

K-means es ideal para el problema de Clasificación de Activos Financieros debido a su capacidad para agrupar puntos en k categorías basándose en similitudes internas.

1. Propósito de K-means en la discretización: Dividir una característica continua en k intervalos (grupos) de valores con características similares.
   Los intervalos definidos por K-means no dependen de etiquetas externas.

2. Ventajas del uso de K-means: Adaptabilidad: Los grupos se ajustan dinámicamente a la distribución de los datos.
   Reducción de complejidad: Ayuda a simplificar características continuas al representar los datos en un menor número de categorías discretas.
   Flexibilidad: Se puede ajustar el número de grupos k según las necesidades del análisis.

Calidad de Grupos

En el artículo de Davies y Bouldin [19] se introduce una métrica para evaluar la separación entre grupos en un conjunto de datos con el fin de medir la calidad de una partición de datos en grupos.

Se le conoce como el índice de Davies-Bouldin (DBI), se basa en una combinación de dispersión interna de cada grupo y la distancia entre los centroides de grupos.

Su objetivo es evaluar la similitud o separación entre grupos para determinar si es una partición adecuada [19].
Para ello toma en cuenta la dispersión del grupo que es una medida de la extensión del grupo i y se define como la media de cada punto al centroide del grupo

$$S_i = \left(\frac{1}{T_i} \sum_{j=1}^{T_i} ||X_j - A_i||^q \right)^{1/q}, \tag{2.22}$$

donde T_i es un número de puntos en el grupo i, X_j representa cada punto del grupo y A_i es el centroide del grupo. La q es el tipo de la medida de dispersión a usar. De lo anterior nace la separación entre grupos que es la distancia entre los centroide de los grupos i y j (típicamente se usa la distancia euclidiana)

$$M_{ij} = \left(\sum_{k=1}^{n} |a_{ki} - a_{kj}|^p \right)^{1/p}, \tag{2.23}$$

donde a_{ki} y a_{kj} son las componentes k-ésimos de los centroides de los grupos i y j. La p es para determinar el tipo de métrica de distancia a usar.

En combinación de S_i y M_{ij}, el índice de Davies-Bouldin mide la similitud entre los dos grupos i y j :

$$R_{ij} = \frac{S_i + S_j}{M_{ij}}. \tag{2.24}$$

Un valor alto de R_{ij} indica que los grupos son similares o están cercanos mientras que un valor bajo indica que están bien separados.

La métrica global (DBI) es el promedio del número de grupos con respecto al índice R_{ij}

$$DBI = \frac{1}{N} \sum_{i=1}^{N} \max_{j \neq i} R_{ij}, \tag{2.25}$$

donde N es el número total de grupos.

Este índice DBI busca minimizar la medida promedio de similitud entre cada grupo y el grupo más similar a él.

Por otro lado, en el artículo de Calinski & Harabarsz [13] se propone un método de análisis de grupos basado en la construcción de una dendrita (o árbol de mínima expansión) para agrupar puntos en un espacio euclidiano multidimensional. Esta técnica se centra en minimizar la suma de cuadrados de las varianzas de cada variable dentro de los grupos ($WGSS$) al dividir la dendrita en grupos, reduciendo así las posibles particiones y simplificando el proceso de agrupación.

El objetivo fundamental de la métrica de Calinski y Harabarsz [13] es identificar agrupaciones de puntos en un espacio multidimensional utilizando una estructura de dendritas.
La estructura de la dendrita es la construcción del árbol de mínima expansión (MST), donde cada punto se conecta con su vecino más cercano, Se eliminan aristas para crear grupos con menor $WGSS$. Donde $WGSS$ es:

$$WGSS = \sum_{g=1}^{k} Trace(R_g), \tag{2.26}$$

donde R_g es la matriz de dispersión de cada cluster g. Este criterio minimiza la dispersión de los grupos.

Para determinar el mejor número de clusters se usa el criterio de Varianza de Razón (VRC)

$$VRC = \frac{BGSS/(k-1)}{WGSS/(n-k)}, \tag{2.27}$$

donde $BGSS$ es la suma de cuadrados de las varianzas de cada variable dentro de los grupos, k el número de grupos y n es el número total de puntos en el conjunto [13].

$BGSS$ es calculada en función de las distancias promedio entre los puntos de cada grupo y el centroide general. Por último, usa la distancia euclidiana para calcular la separación entre puntos.

Esta métrica es útil para analizar la estructura de los datos y obtener agrupamientos con la menor dispersión posible dentro de cada grupo, eliminando particiones menos óptimas de la dendrita y proporcionando un VRC que ayuda a determinar el número adecuado de grupos [13].

2.3. Selección/Reducción de Características

Otra manera de reducir el tiempo de ejecución de cada algoritmo es haciendo una selección o reducción de características. Se sabe que las características más importantes en el mercado financiero son aquellas que se relacionan con el tamaño, momentum y el book-to-market [47], también se sabe por los resultados en los modelos de Gu et al. [35], que las características que se basan en la tendencia del precio reciente, las variables líquidas y las medidas de riesgo, son las más relevantes para el problema de predicción de retornos. Al tener este conocimiento a priori, se puede hacer la comparación con métodos que desconocen totalmente el significado y relación de cada característica, como lo es el método de Análisis de Componentes Principales (PCA) el cual obtiene, mediante la varianza del conjunto, las características principales que representan los datos utilizando la descomposición en valores singulares (SVD) [65]. Cada característica nueva obtenida es una combinación lineal de las características originales, que ayuda para captura patrones esenciales y en la reducción de ruido y redundancia. Otro método para hacer selección de características es usando redes bayesianas, estas redes encuentran la relación entre características con la variable objetivo usando una métrica o un clasificador bayesiano mismos que se evalúan para encontrar las características más significativas [7] [49].

Además, para fines de clasificación, la selección/reducción de características elimina la redundancia y preserva las relaciones más importantes de los datos, y hace un reducción al riesgo del sobreajuste [8].

Análisis de Componentes Principales

La definición más común del Análisis de Componentes Principales (PCA, por sus siglas en inglés), atribuida a Hotelling [39], establece que, dado un conjunto de datos observados de vec-

tores d-dimensionales $\{x_n\}$, $n \in \{1,...,N\}$, los q ejes principales u_j, $j \in \{1,...,q\}$, son aquellos vectores ortonormales para los cuales la varianza bajo la proyección es máxima. Se ha demostrado que los vectores u_j corresponden a los q vectores propios dominantes de la matriz de covarianza muestral [76]:

$$S = \sum_n \frac{(x_n - \bar{x})(x_n - \bar{x})^T}{N}, \tag{2.28}$$

tal que

$$Su_j = \lambda_j u_j, \tag{2.29}$$

donde $\bar{x}$ es la media muestral, y λ_j son los valores propios.

El vector proyectado se define como

$$\tilde{X_n} = U^T (x_n - \bar{x}), \tag{2.30}$$

donde $U = \left(u_1, u_2, ..., u_q\right)$ representa una representación reducida q - dimensional del vector observado x_n.

Una propiedad complementaria del PCA, relacionada con las discusiones originales de Pearson (1901) [76], es que la proyección en el subespacio principal minimiza el error cuadrático de reconstrucción:

$$\sum_N ||x_n - \bar{x}||^2. \tag{2.31}$$

La reconstrucción lineal óptima utiliza las columnas de U, que abarcan el espacio generado por los q vectores propios principales de S. En este contexto, la proyección del PCA también se conoce como la transformada de Karhunen-Loève [76].

Formulación de máxima varianza:

Considérese un conjunto de datos de observaciones $\{x_n\}$ donde $n = 1,...,N$, y x_n es una variable Euclidiana con dimensionalidad D. El objetivo es proyectar los datos en un espacio de dimensionalidad $M < D$ maximizando la varianza de los datos proyectados. Por el momento, se considera que el valor de M es dado.

Para comenzar, considérese la proyección en un espacio unidimensional ($M = 1$). La dirección de este espacio se define mediante un vector u_1 de dimensión D, el cual, por conveniencia, se elige como un vector unitario tal que $u_1^T u_1 = 1$. Cada punto de datos x_n se proyecta entonces en un valor escalar $u_1^T x_n$. La media de los datos proyectados es $u_1^T \bar{x}$, donde $\bar{x}$ es la media muestral definida como

$$\bar{x} = \frac{1}{N} \sum_{n=1}^{N} x_n, \tag{2.32}$$

y la varianza de los datos proyectados está dada por

$$\frac{1}{N}\sum_{n=1}^{N}\{u_1^T x_n - u_1^T \bar{x}\}^2 = u_1^T S u_1, \tag{2.33}$$

donde S es la matriz de covarianza de los datos definida como

$$S = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})(x_n - \bar{x})^T. \tag{2.34}$$

El siguiente paso consiste en maximizar la varianza proyectada $u_1^T S u_1$ respecto a u_1 [8]. Esta debe ser una maximización con restricción para evitar que $||u_1|| \to \infty$. La restricción apropiada proviene de la condición de normalización $u_1^T u_1 = 1$. Para imponer esta restricción, se introduce un multiplicador de Lagrange denotado por λ_1, y se realiza una maximización no restringida de

$$u_1^T S u_1 + \lambda_1\left(1 - u_1^T u_1\right). \tag{2.35}$$

Al establecer la derivada respecto a u_1 igual a cero, se obtiene un punto estacionario cuando

$$S u_1 = \lambda_1 u_1,$$

lo cual indica que u_1 debe ser un vector propio de S [8]. Multiplicando por la izquierda con u_1^T y haciendo uso de $u_1^T u_1 = 1$, se observa que la varianza está dada por

$$u_1^T S u_1 = \lambda_1, \tag{2.36}$$

y por lo tanto, la varianza será máxima cuando u_1 se iguale al vector propio correspondiente al mayor valor propio λ_1. Este vector propio se conoce como la primera componente principal [8].

Se pueden definir componentes principales adicionales de forma incremental al elegir cada nueva dirección de modo que maximice la varianza proyectada entre todas las posibles direcciones ortogonales a las ya consideradas. Para el caso general de un espacio de proyección M-dimensional, la proyección lineal óptima que maximiza la varianza de los datos proyectados está definida por los M vectores propios $u_1, ..., u_M$ de la matriz de covarianza de los datos S, correspondientes a los M mayores valores propios $\lambda_1, ..., \lambda_M$.

En resumen, el análisis de componentes principales implica calcular la media $\bar{x}$ y la matriz de covarianza S del conjunto de datos, y luego encontrar los M vectores propios de S asociados a los M mayores valores propios [8].

Formulación de mínimo error:

Se introduce un conjunto ortonormal completo de vectores base D - dimensionales $\{u_i\}$, donde $i = 1, ..., D$, que satisfacen $u_i^T u_j = \delta_{ij}$ (la delta de Kronecker). Debido a que esta base es completa, cada punto de datos puede representarse exactamente como una combinación lineal de los vectores base:

$$x_n = \sum_{i=1}^{D}\alpha_{ni} u_i, \tag{2.37}$$

donde los coeficientes α_{ni} serán diferentes para cada punto de datos. Esto simplemente representa una rotación del sistema de coordenadas a un nuevo sistema definido por $\{u_i\}$, en el cual los D componentes originales $\{x_{n1},...,x_{nD}\}$ se reemplazan por un conjunto equivalente $\{\alpha_{n1},...,\alpha_{nD}\}$. Tomando el producto interno con u_j y haciendo uso de la propiedad de ortonormalidad, se obtiene:

$$\alpha_{nj} = x_n^T u_j, \tag{2.38}$$

y por lo tanto, sin pérdida de generalidad, se puede escribir:

$$x_n = \sum_{i=1}^{D} \left(x_n^T u_i\right) u_i. \tag{2.39}$$

El objetivo es aproximar este punto de datos utilizando una representación que involucre un número restringido $M < D$ de variables, correspondientes a una proyección sobre un subespacio de menor dimensionalidad [8]. El subespacio lineal M-dimensional puede representarse mediante las primeras M de las bases vectoriales, y así se aproxima cada punto de datos x_n como:

$$\tilde{x}_n = \sum_{i=1}^{M} z_{ni} u_i + \sum_{i=M+1}^{D} b_i u_i, \tag{2.40}$$

donde $\{z_{ni}\}$ dependen del punto de datos particular, mientras que $\{b_i\}$ son constantes que son las mismas para todos los puntos de datos. Se tiene la libertad de elegir $\{u_i\}$, $\{z_{ni}\}$ y $\{b_i\}$ de modo que se minimice la distorsión introducida por la reducción en dimensionalidad [8].

Como medida de esta distorsión, se utiliza la distancia cuadrática entre el punto de datos original x_n y su aproximación $\tilde{x}_n$, promediada sobre el conjunto de datos. Por lo tanto, el objetivo es minimizar:

$$J = \frac{1}{N} \sum_{n=1}^{N} ||x_n - \tilde{x}_n||^2. \tag{2.41}$$

Primero, se minimiza respecto a las cantidades $\{z_{ni}\}$. Sustituyendo $\tilde{x}_n$, estableciendo la derivada respecto a z_{nj} igual a cero y usando las condiciones de ortonormalidad, se obtiene:

$$z_{nj} = x_n^T u_j, \tag{2.42}$$

donde $j = 1,...,M$. De manera similar, estableciendo la derivada de J respecto a b_i igual a cero, y nuevamente usando la relación de ortonormalidad, se obtiene:

$$b_i = \bar{x}^T u_i, \tag{2.43}$$

donde $i = M+1,...,D$. Sustituyendo z_{ni} y b_i en la fórmula general, se obtiene:

$$x_n - \tilde{x}_n = \sum_{i=M+1}^{D} \left((x_n - \bar{x})^T u_i\right) u_i. \tag{2.44}$$

Esto muestra que el vector de desplazamiento desde x_n hasta $\tilde{x}_n$ yace en el espacio ortogonal al subespacio principal, ya que es una combinación lineal de $\{u_i\}$ para $i = M+1,...,D$. Esto es

de esperar porque los puntos proyectados $\tilde{x}_n$ deben residir dentro del subespacio principal, y el error mínimo está dado por la proyección ortogonal [8].

Por lo tanto, se obtiene una expresión para la medida de distorsión J como una función únicamente de $\{u_i\}$:

$$J = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=M+1}^{D}\left(x_n^T u_i - \bar{x}^T u_i\right)^2 = \sum_{i=M+1}^{D} u_i^T S u_i. \tag{2.45}$$

Resta la tarea de minimizar J respecto a $\{u_i\}$, lo cual debe hacerse bajo la restricción de ortonormalidad, de lo contrario se obtendría el resultado trivial $u_i = 0$ [8].

Las restricciones provienen de las condiciones de ortonormalidad, y la solución se expresa en términos de la expansión en vectores propios de la matriz de covarianza. La solución general para la minimización de J para cualquier D y $M < D$ se obtiene eligiendo $\{u_i\}$ como los vectores propios de la matriz de covarianza [8] que satisfacen:

$$Su_i = \lambda_i u_i,$$

donde $i = 1, ..., D$. Como es habitual, los vectores propios $\{u_i\}$ se eligen ortonormales. El valor correspondiente de la medida de distorsión está dado por:

$$J = \sum_{i=M+1}^{D} \lambda_i, \tag{2.46}$$

que es simplemente la suma de los valores propios de los vectores propios que son ortogonales al subespacio principal. Por lo tanto, el valor mínimo de J se obtiene seleccionando estos vectores propios como aquellos asociados a los $D-M$ valores propios más pequeños, dejando los M valores propios más grandes para definir el subespacio principal.

Aunque se ha considerado $M < D$, el análisis PCA sigue siendo válido para $M = D$, en cuyo caso no hay reducción de dimensionalidad, sino simplemente una rotación de los ejes de coordenadas para alinearlos con las componentes principales.

Uso de descomposición en valores singulares en PCA

Para mejorar la calidad y eficiencia en el cálculo de las componentes principales, se hace uso de la descomposición en valores singulares (SVD, por sus siglas en inglés) [65].

Se sabe de los apartados anteriores que PCA transforma un conjunto de datos de D dimensiones originales a un espacio de menor dimensión k, maximizando la varianza explicada por las nuevas variables. Matemáticamente, esto equivale a:

- Encontrar los vectores propios (u_i) de la matriz de covarianza del conjunto de datos.
- Ordenarlos según sus valores propios (λ_i), que representan la varianza explicada por cada componente.

Sin embargo, calcular directamente la matriz de covarianza y resolver sus valores propios puede ser computacionalmente costoso para grandes conjuntos de datos. La descomposición en valores singulares hace que este proceso sea más eficiente y estable numéricamente [65].

SVD descompone una matriz X, que contiene los datos centrados (es decir, con la media de cada característica restada), en tres componentes fundamentales [33]:

$$X = U\Sigma V^T, \tag{2.47}$$

donde:

- U: Matriz ortonormal cuyos vectores columna son los vectores singulares de las filas de X.
- Σ: Matriz diagonal que contiene los valores singulares de X en orden descendente.
- V^T: Matriz ortonormal cuyos vectores fila son los vectores singulares de las columnas de X.

La conexión entre SVD y PCA radica en que [65]:

- Las columnas de V representan las "direcciones principales" del espacio de datos (las componentes principales).
- Los valores de Σ^2 corresponden a la varianza explicada por cada componente principal.

De este modo, en lugar de calcular explícitamente la matriz de covarianza y resolver sus valores propios, PCA utiliza directamente SVD para descomponer los datos. El proceso se lleva a cabo en los siguientes pasos [65]:

1. **Centrar los datos:** Restar la media de cada característica a la matriz original de datos, obteniendo X, que contiene los datos centrados.
2. **Aplicar SVD:** Descomponer X usando U, Σ y V^T mediante SVD.
3. **Seleccionar las componentes principales:** Tomar las primeras k columnas de V (las direcciones principales) y calcular las proyecciones en el nuevo espacio como:

$$Z = XV_k, \tag{2.48}$$

   donde V_k contiene las primeras k columnas de V.

Este procedimiento permite implementar PCA de manera eficiente, especialmente para conjuntos de datos de alta dimensionalidad, y garantiza la estabilidad numérica en el cálculo de las componentes principales [65].

Método del Codo

Para determinar el número óptimo de agrupaciones (k) en algoritmos como K Vecinos Más Cercanos, K-Means o PCA, una de las técnicas más utilizadas es el **método del codo** [31]. La idea central consiste en analizar la variación de la inercia (*within-cluster sum of squares*) medida que se incrementa el valor de k. La inercia mide qué tan cerca están los puntos de sus

centroides dentro de cada clúster, por lo que valores más bajos indican una mayor compacidad de los grupos.

Al graficar la inercia en función de k, se observa que a medida que se aumenta el número de clústeres, la inercia disminuye rápidamente al inicio y después comienza a decrecer más lentamente. Este comportamiento genera una curva con forma de brazo, en la cual suele aparecer un punto de inflexión denominado **"codo"**. Dicho punto indica el valor de k a partir del cual añadir más clústeres deja de aportar mejoras significativas en la compacidad de los grupos, ya que las reducciones en la inercia son marginales. Esto se aprecia mejor en la figura (2.1) donde se señala el mejor cluster.

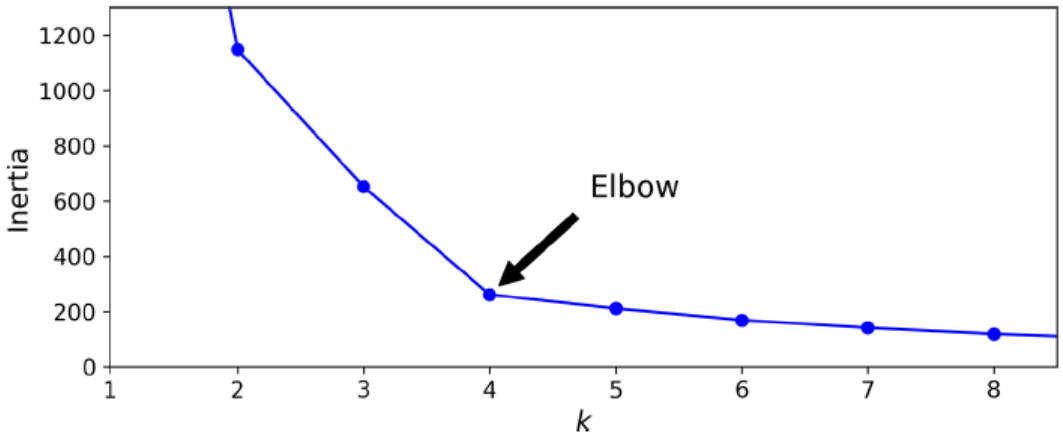


Figura 2.1: Al graficar la inercia en función del número de grupos k, la curva a menudo contiene un punto de inflexión llamado codo. Fuente: [31, p. 246]

De esta forma, el método del codo permite seleccionar un valor de k que equilibra la complejidad del modelo y la calidad de la agrupación. En otras palabras, evita tanto elegir un número demasiado pequeño de clústeres, que mezcle patrones heterogéneos, como un número demasiado grande, que fragmente innecesariamente los datos [31].

Redes Bayesianas

Además del uso de PCA, se propone hacer una selección de características usando redes bayesianas para mejorar los resultados y tiempo de ejecución de cada modelo. Al obtener las características más importantes dada la red bayesiana para cada conjunto de datos, junto con PCA, se puede comparar estos resultados con respecto a la selección de características que indican [35, 47].

Las redes bayesianas son modelos gráficos probabilísticos que representan un conjunto de variables y sus relaciones condicionales mediante un grafo dirigido acíclico (DAG), con $G(V,E)$ donde n es el número de nodos en V que representan las variables y E son las aristas que representas las dependencias de los nodos de todo el conjunto [7]. Cada nodo en el grafo representa una variable, y las aristas dirigidas indican dependencias condicionales entre las variables, facilitando el modelo de sistemas complejos con interacciones no deterministas [7, 49]. Las redes bayesianas representa el conjunto de relaciones de independencia condicional en la distribución

de probabilidad conjunta sobre todas las variables, $P(X)$. Se tienen dos suposiciones sobre el DAG en una red bayesiana [49]:

1. Condición de Markov: toda variable X en G es condicionalmente independiente de sus no descendientes dados sus padres. Esto es equivalente a decir que toda independencia condicional implicada por la separación d en el DAG está presente en la distribución de probabilidad conjunta $P(X)$ que se puede descomponer así:

$$P(X) = \prod_{i=1}^{n} P(X_i|Pa(X_i)) \tag{2.49}$$

   donde $Pa(X_i)$ son los padres de X_i.

2. Condición de minimalidad: no se puede eliminar ninguno de los bordes del DAG sin que el gráfico implique una relación de independencia condicional que no está presente en $P(X)$.

El aprendizaje de estructura de una red bayesiana implica determinar las relaciones entre las variables a partir de los datos. Este problema es NP-Duro debido al crecimiento exponencial del espacio de búsqueda de posibles DAGs con respecto al número de variables [7].

Existen dos enfoques principales para este tipo de aprendizaje [7] [49]:

1. Métodos basados en restricciones: Utilizan pruebas de independencia condicional para construir el grafo, enfocándose en las relaciones presentes en los datos.

2. Métodos basados en puntajes: Formulan el aprendizaje como un problema de optimización, evaluando la calidad de los modelos mediante una función de puntaje.

Selección de características por MDL-FS

En el artículo de Drugan y Wiering, "*Feature selection for Bayesian network classifiers using the MDL-FS score*" [22], elaboran un método de selección de características con Minimum Description Length (MDL) y algoritmos de clasificación, Naive Bayes y Tree Augmented Naive Bayes (TAN).

El principio **MDL** se fundamenta en la teoría de la información y se utiliza para evaluar la calidad de un modelo penalizando la complejidad y favoreciendo aquellos modelos que describen de forma concisa los datos. La métrica **MDL** tradicional se define como:

$$MDL(B|D) = \frac{logN}{2}|B| - LL(B|D) \tag{2.50}$$

donde $|B|$ es el número de parámetros del modelo, $LL(B|D)$ es el logaritmo de la verosimilitud del modelo dado los datos y N es número de instancias. Se tiene una base de datos muy grande el segundo término castigará al primer término. El logaritmo de la verosimilitud se define como:

$$LL(B|D) = -N\sum H_{\hat{p}}(V_i|Pa(V_i)), \tag{2.51}$$

donde $H_{\hat{p}}$ representa la entropía condicional. Cuanto mayor es el *LL*, mejor representa el modelo de los datos. En este caso, **MDL** se usa mantener o eliminar características, siguiendo estrictamente el mismo principio teórico.

Este enfoque favorece modelos que describen de manera concisa los datos, pero presenta una limitación importante para tareas de clasificación: al capturar la distribución conjunta $P(C,A)$ (donde C es la variable de clase y A el conjunto de características), el MDL tradicional no enfatiza la distribución condicional $P(C \mid A)$ que resulta crucial para la clasificación[22].

En el artículo de Drugan y Wiering [22], se utilizan dos clasificadores: Naive Bayes y TAN. En este trabajo se opta por Naive Bayes, el cual se basa en la hipótesis de independencia condicional de las características dado la clase. La estructura de una red Naive Bayes es la siguiente [22]:

- El nodo raíz es la variable de la clase C.
- Todas las características A_i son hijos directos de C y no tienen relaciones entre ellos.

La distribución conjunta de esta red se factoriza como:

$$P(C,A) = P(C)\prod P(A_i|C). \tag{2.52}$$

La probabilidad condicional se obtiene aplicando la regla de Bayes:

$$P(C|A) = \frac{P(C,A)}{P(A)}. \tag{2.53}$$

Sustituyendo la factorización dada:

$$P(C|A) = \frac{P(C)\prod P(A_i|C)}{P(A)}, \tag{2.54}$$

donde la probabilidad marginal $P(A)$ se obtiene sumando sobre todas las clases posibles:

$$P(A) = \sum_C P(C)\prod P(A_i|C). \tag{2.55}$$

Este modelo simple permite realizar cálculos eficientes, pero puede verse afectado si se incluyen características redundantes.

Para abordar el problema de la selección de características, se consideran los siguientes puntos fundamentales [22]:

Selección de características: Dado un conjunto de características A, una variable de clase C y un conjunto de datos D, encontrar un subconjunto mínimo de características $S \subseteq A$ tal que el clasificador construido sobre S maximice una métrica de desempeño R.

Concepto de redundancia en características: La redundancia de una característica significa que su eliminación no afecta la clasificación porque su información ya está contenida en otros atributos.

Redundancia: una característica A_i es redundante con respecto a C dado un conjunto de características S si:

$$P(C|A_i,S) = P(C|S), \tag{2.56}$$

es decir que, una vez conocidos los valores de las características en S, la información de A_i, no cambia la probabilidad de la clase C.
Existen niveles de redundancia que dependen del tamaño del subconjunto S [22]:

1. **Redundancia de nivel** 0: A_i no aporta información sobre C por sí solo.
2. **Redundancia de nivel** m: A_i es redundante dado un conjunto de características.
3. **Redundancia total**: A_i es redundante para cualquier $S \subseteq A$.

Ruido en los datos: Las bases de datos pueden contener ruido (errores en los datos, valores incorrectos, errores de medición). Esto puede afectar la identificación de características redundantes [22].

Además, se reconoce que la presencia de ruido en los datos puede dificultar la identificación exacta de la redundancia, ya que errores o variaciones aleatorias pueden hacer que una característica irrelevante parezca tener cierta dependencia con la clase.

Función MDL-FS

MDL tradicional mide la calidad de un modelo en función de la longitud de su descripción, es decir, penaliza modelos más complejos para evitar el sobreajuste. También, sólo mide la probabilidad conjunta $P(C,A)$ en lugar de la condicional $P(C|A)$, siendo deficiente para problemas de clasificación [22].

Dado lo anterior y combinando varios conceptos ya vistos, los autores definen la métrica **MDL-FS**, que es una modificación del **MDL** tradicional incorporando la probabilidad condicional de una red bayesiana y del **MDL** [22]:

$$MDL\text{-}FS(C,S|D) = \frac{logN}{2}|C| - CALL(C,S|D), \tag{2.57}$$

donde

- C es el clasificador bayesiano,
- S es una red auxiliar que modela la distribución de características $P(A)$,
- $CALL(\cdot)$ es la diferencia entre log-verosimilitud del clasificador $LL(C|D)$ y la log-verosimilitud de la red auxiliar $LL(S|D)$.

Las log-verosimilitudes están definidas de la siguiente forma:

$$LL(S|D) = -N\sum H_{\hat{p}}(A_i|P(A_i)), \tag{2.58}$$

$$LL(C|D) = -NH_{\hat{p}}(C|A). \tag{2.59}$$

La red auxiliar captura las dependencias entre las características, permitiendo eliminar aquellos que resultan redundantes y que el **MDL** tradicional no logra identificar.

En el mismo artículo [22] demuestran que **MDL-FS** se relaciona con la entropía condicional de la clase dada las características.

$$CALL(C,S|D) \approx -NH_{\hat{p}}(C|A), \tag{2.60}$$

lo que significa que **MDL-FS** minimiza la entropía condicional de C, haciéndola más efectiva para la clasificación.

La entropía condicional, en general, está definida como:

$$H_p(X|Y) = \sum P(x,y)\log P(x,y) \tag{2.61}$$

2.4. Métodos de Aplicados a Clasificación

Después de explicar toda la teoría de cómo transformar los datos se sigue con detallar los métodos de inteligencia artificial seleccionados para la clasificación de los datos, posterior a su discretización. Los modelos utilizados incluyen Naive Bayes, CART (*Classification and Regression Trees*), Regresión Logística, K Vecinos Más Cercanos y Redes Neuronales.

1. Naive Bayes [57].
   - Ventajas: Es un modelo rápido y sencillo, especialmente útil en problemas de clasificación de texto y cuando se asume independencia entre las características. También maneja bien datos ruidosos.
   - Limitaciones: Su principal limitación radica en la suposición de independencia entre las características, lo cual puede ser poco realista en muchos contextos. Esto puede llevar a un desempeño subóptimo en datos con alta correlación entre variables.
2. CART [57].
   - **Ventajas:** CART es un modelo interpretable y fácil de visualizar, ya que construye un árbol que representa de manera jerárquica el proceso de clasificación. Resulta adecuado para problemas no lineales y permite capturar interacciones complejas entre las variables. Aplicando entropía como criterio de impureza, es posible cuantificar la ganancia de información en cada partición, lo que proporciona divisiones consistentes con los principios de la teoría de la información.
   - **Limitaciones:** Al igual que otros algoritmos de árboles, CART es susceptible al sobreajuste, especialmente cuando el árbol crece sin restricciones o se trabaja con conjuntos de datos pequeños. Además, en problemas con un número elevado de características o con presencia de ruido, el costo computacional puede aumentar significativamente al evaluar múltiples divisiones posibles. Por ello, es necesario aplicar técnicas de regularización como la poda o limitar la profundidad máxima del árbol.
3. Regresión Logística [8].
   - Ventajas: Este modelo es eficiente y estable en problemas lineales. Es adecuado cuando las clases son linealmente separables y permite interpretar los coeficientes de cada variable.

- Limitaciones: Su capacidad es limitada en problemas no lineales, ya que no puede capturar interacciones complejas entre las variables.

4. *K* Vecinos Más Cercanos (KNN) [71].

   - Ventajas: KNN es intuitivo y funciona bien con datos no lineales, pues clasifica en función de la proximidad a los puntos de entrenamiento. Es versátil y no hace suposiciones sobre la distribución de los datos.
   - Limitaciones: Su desempeño depende mucho de la selección del valor de K, y puede ser computacionalmente costoso cuando el conjunto de datos es grande.

5. Redes Neuronales [8].

   - Ventajas: Las redes neuronales son potentes para capturar relaciones complejas y no lineales en los datos. Son altamente flexibles y pueden adaptarse bien a problemas de gran tamaño y de estructura compleja.
   - Limitaciones: Su entrenamiento puede ser costoso en términos de tiempo y recursos computacionales, y requieren grandes volúmenes de datos para evitar el sobreajuste. No son interpretables en comparación con otros modelos, ya que el funcionamiento de las capas ocultas es difícil de interpretar.

La combinación de estos modelos permitirá capturar tanto relaciones lineales como no lineales.

Naive Bayes

Este clasificador se basa en el Teorema de Bayes [57], asume que las características de entrada son condicionalmente independientes entre sí, esta es la ingenuidad del clasificador, dado el resultado de la clase, lo que simplifica en gran medida el cálculo de probabilidades. La fórmula básica del teorema es:

$$P(C_i|x_i) = \frac{P(x_i|C_i)P(C_i)}{P(x_i)} \tag{2.62}$$

donde $P(C|x)$ es la probabilidad posterior de la clase C dado el vector de características x, $P(x|C)$ es la probabilidad de observar los datos x dado que la clase es C, $P(C)$ es la probabilidad previa de la clase C y $P(x)$ es la probabilidad marginal de observar x.

Como el clasificador asume que cada característica es independiente de las demás, dada la clase, facilita el cálculo $P(x|C)$ como el producto de probabilidades individuales de cada característica:

$$P(x|C) = \prod_k P(x_k|C). \tag{2.63}$$

Además, el clasificador maximiza la probabilidad posterior (MAP), lo cual se puede simplificar omitiendo el denominador de la Ec.2.62 $P(x)$, ya que es constante para todas las clases

$$C* = \underset{C}{\operatorname{argmax}} P(C) \prod_k P(X_k, C) \tag{2.64}$$

con $C*$ para la característica seleccionada.

CART

Para comprender el funcionamiento de los árboles de decisión, es fundamental introducir el concepto de **entropía de la información** [57]. Este fue planteado originalmente por Claude Shannon en su artículo "*A Mathematical Theory of Communication*", donde propone la medida de entropía como una forma de cuantificar la incertidumbre o aleatoriedad en la distribución de las clases dentro de un nodo. En el contexto de clasificación, la entropía H de un conjunto de probabilidades P_i se define como:

$$Entropy(P) = -\sum_i P_i \log_2 P_i,$$

donde el logaritmo se expresa en base 2 porque se considera una codificación binaria en bits.

El algoritmo CART utiliza medidas de impureza como el **índice de Gini** o la **entropía** [11]. En este trabajo se emplea la entropía como criterio de división, lo que permite cuantificar la **ganancia de información** obtenida al particionar los datos con una característica específica. La ganancia de información se define como la entropía del conjunto completo menos la entropía esperada al dividirlo según una característica F [57]. Sea S el conjunto de ejemplos, F una característica y $|S_f|$ el número de instancias en S con valor f en la característica F, la fórmula es:

$$Gain(S,F) = Entropy(S) - \sum_{f \in values(F)} \frac{|S_f|}{|S|} Entropy(S_f). \tag{2.65}$$

Durante el entrenamiento, CART evalúa todas las divisiones posibles para cada característica, selecciona la que produce la mayor reducción de entropía (máxima ganancia de información) y crea un nodo de decisión [11]. CART puede generar tanto **árboles binarios de clasificación** como de regresión, lo que lo convierte en un algoritmo más general. En cada nodo, el conjunto de datos se divide recursivamente en dos subconjuntos hasta cumplir condiciones de parada, como alcanzar una clase pura, superar un número mínimo de muestras o una profundidad máxima.

El procedimiento general de CART es el siguiente:

1. Calcular la impureza (entropía en este caso) para cada división candidata.
2. Seleccionar la división que maximiza la ganancia de información.
3. Particionar el conjunto de datos en dos subconjuntos según la división seleccionada.
4. Repetir el proceso recursivamente en cada subconjunto hasta que:
    - Todos los datos en un subconjunto pertenecen a la misma clase.

- Se alcanza un criterio de parada predefinido (profundidad máxima, número mínimo de muestras, etc.).

El resultado es un árbol binario que refleja las divisiones realizadas, el cual puede interpretarse de manera intuitiva y permite capturar interacciones entre las características [38].

Se eligió la entropía como criterio de impureza en CART por su fundamento en la teoría de la información, permitiendo calcular la ganancia de información, siendo más consistente con la naturaleza de los datos financieros discretizados, en los que es importante capturar señales poco frecuentes pero críticas en el comportamiento de los retornos.

Regresión Logística

La *regresión logística* es un modelo de clasificación que pertenece a la familia de los modelos lineales generalizados [8]. En el artículo de David R. Cox [18] "*The regression analysis of binary sequences*" aborda el problema de analizar secuencias de ensayos con dos posibles resultados (como éxito o fracaso) y estudia cómo la probabilidad de uno de esos resultados depende de una o más variables independientes. Formula del modelo logístico (pionero para crear la regresión logística):

Función de logit: transforma la probabilidad p de éxito de un evento binario en una expresión lineal para poder modelarla con regresión

$$logit(\theta_t) = \log\left(\frac{\theta_t}{1-\theta_t}\right) = \alpha + \beta X_t, \tag{2.66}$$

donde θ_t es la probabilidad de que el resultado de la variable objetivo en el ensayo t sea 1, X_t es una variable asociada al ensayo t, α es el intercepto y β es el coeficiente de regresión [18].

La idea es que el modelo logístico sea lo suficientemente flexible como para captar la dependencia entre las variables y el resultado binario sin caer en los problemas de un modelo lineal estándar, ya que las probabilidades están restringidas al intervalo $[0,1]$ [18].

A partir de la función *logit*, se deriva la probabilidad de que el evento ocurra en función de las variables independientes [18]. Para representar la probabilidad de éxito en un ensayo dado, se tiene la siguiente ecuación:

$$\theta_t = \frac{1}{1+e^{-(\alpha+\beta X_t)}}. \tag{2.67}$$

En el contexto de clasificación binaria, la probabilidad posterior de que una observación pertenezca a una clase C_1 se modela aplicando la *función sigmoide logística* a una combinación lineal de las características del vector de entrada ϕ, lo que da como resultado:

$$p(C_1|\phi) = y(\phi) = \sigma(\mathbf{w}^T\phi), \tag{2.68}$$

donde $\sigma(z) = \frac{1}{1+e^{-z}}$ es la función logística sigmoide (la ecuación 2.67). En este caso, $\sigma(z)$ actúa como una función que restringe los valores de la salida al intervalo $[0,1]$, facilitando su interpretación como probabilidades.

Para *ajustar* los parámetros $\mathbf{w}$ del modelo, se utiliza el método de *máxima verosimilitud*, que maximiza la probabilidad de los datos observados bajo el modelo propuesto. Esto implica definir

una *función de error* basada en la verosimilitud negativa, comúnmente conocida como *entropía cruzada* [8]:

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \ln y_n + (1-t_n)\ln(1-y_n) \right], \tag{2.69}$$

donde $y_n = p(C_1|\phi_n)$ es la probabilidad predicha para la clase C_1 en la instancia n, y t_n es el valor observado (0 o 1). La minimización de esta función de error permite encontrar los valores óptimos de $\mathbf{w}$.

El gradiente de esta función de error respecto a los pesos $\mathbf{w}$ es:

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\phi_n, \tag{2.70}$$

lo que facilita el uso de algoritmos de optimización como el *gradiente descendente* para ajustar los parámetros. Cabe destacar que la función sigmoide hace que la regresión logística se comporte de manera estable y convergente en contextos donde las clases son linealmente separables, aunque este método puede resultar en *sobreajuste* cuando los datos son perfectamente separables [8].

Además, se advierte que el ajuste por máxima verosimilitud puede no ser suficiente cuando hay problemas de sobreajuste o inestabilidad, en cuyo caso se puede utilizar una *regularización* para penalizar valores excesivos de $\mathbf{w}$ y mejorar la generalización del modelo [8].

La reguralización agrega un término de penalización a la función de error con el objetivo de evitar que los coeficientes del modelo tomen valores excesivamente grandes. La forma más simple de esta penalización es la suma de los cuadrados de todos los coeficientes, lo cual lleva a una función de error modificada de la forma [8]:

$$\tilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} (y(\mathbf{x}_n, \mathbf{w}) - t_n)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2, \tag{2.71}$$

donde $\|\mathbf{w}\|^2 = w_0^2 + w_1^2 + \cdots + w_M^2$, x_n representa el vector de características y el coeficiente λ controla la importancia relativa del término de regularización en comparación con el término de error cuadrático.

Es importante notar que, a menudo, el coeficiente w_0 se excluye de la regularización, ya que su inclusión podría hacer que los resultados dependan de la elección del origen para la variable objetivo. Este tipo de técnica es conocido en la literatura estadística como *método de reducción* (*shrinkage*) [8], ya que reduce el valor de los coeficientes. Cuando la regularización cuadrática se aplica de esta forma, se conoce como *regresión de Ridge* ($L2$) [38].

La regularización *Lasso* ($L1$) es una técnica de regularización que corresponde al caso en que el término de penalización en la función de error regularizada utiliza la norma $L1$, es decir, $q = 1$ en el término regularizador [8]. Esto se expresa como:

$$\frac{1}{2}\sum_{n=1}^{N} \left(t_n - \mathbf{w}^T \phi(\mathbf{x}_n) \right)^2 + \lambda \sum_{j=1}^{M} |w_j|, \tag{2.72}$$

donde λ es el coeficiente de regularización, el cual controla la importancia relativa del término de regularización.

Una propiedad importante de *Lasso* es que, si el valor de λ es suficientemente grande, algunos de los coeficientes w_j se reducen exactamente a cero [8]. Esto da lugar a un modelo *sparse* (disperso) en el cual ciertas variables no tienen peso alguno, lo que efectivamente selecciona un subconjunto de las características relevantes y simplifica el modelo [8].

Matemáticamente, minimizar la función de error con la penalización $L1$ es equivalente a minimizar la función de error sin regularización sujeta a la restricción:

$$\sum_{j=1}^{M} |w_j| \leq \eta, \tag{2.73}$$

donde η es un parámetro que regula el límite de la suma de los valores absolutos de los coeficientes.

En resumen, la regularización *Lasso* es un ejemplo de *método de reducción de parámetros* (*shrinkage*), ya que tiende a reducir el valor de los coeficientes. Su principal ventaja es que permite entrenar modelos complejos sin sobreajustar, limitando la complejidad efectiva del modelo mediante la selección de características.

Otro tipo de regularización es ElasticNet la cual es una combinación de las dos anteriores $L1$ y $L2$, ElasticNet selecciona las variables como Lasso y reduce los coeficientes de los predictores correlacionados como Ridge [38].

SAGA: Un Método de Gradiente Incremental Rápido

En el artículo "*SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives*" [20] se propone el algoritmo *SAGA* como un método de optimización incremental propuesto para resolver problemas de minimización de suma finita en aprendizaje de máquina, donde el objetivo es minimizar funciones de la forma:

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x), \tag{2.74}$$

donde cada f_i es convexa y con derivadas Lipschitz continuas.

Algoritmo SAGA [20]: en cada iteración, el algoritmo selecciona aleatoriamente un índice y realiza una actualización de la siguiente forma:

1. Actualiza el gradiente del índice seleccionado en una estructura de almacenamiento.

2. Realiza la actualización de x utilizando el gradiente promedio almacenado y un paso con tamaño γ.

3. Aplica el operador proximal en caso de un término de regularización no diferenciable.

SAGA es especialmente útil para problemas de gran escala en los que el almacenamiento de gradientes puede ser costoso [20].

K Vecinos Más Cercanos

La idea clave de los modelos de vecinos más cercanos [71] es que es probable que las propiedades de un punto de entrada particular x sean similares a las de los puntos cercanos a x. El medir el tamaño de la vecindad o las cercanías de un punto en una distribución de puntos es un problema fundamental para este tipo de algoritmos.
Si la vecindad es muy pequeña, no contendrá ningún punto de los datos; si es muy grande, puede incluir a todos los puntos de los datos, dando lugar a una estimación de la característica que es la misma en cualquier lugar. Un solución es definir la vecindad lo suficiente grande como para incluir k puntos, donde k es lo suficientemente grande como para asegurar una estimación con significado. Para k fijo, el tamaño de la vecindad varía; si los datos están dispersos, la vecindad es grande, pero si los datos están muy juntos, la vecindad es pequeña [71].

Para identificar los vecinos más cercados de un punto, se necesita una métrica para medir la distancia $D(x_t, x_i)$. La distancia euclidiana o la distancia de Mahalanobis son ejemplos de algunos tipos. Para características discretas se puede utlizar la distancia de Hamming que define $D(x_t, x_i)$ como el número de características en las que x_t y x_i difieren [61].

Dado un ejemplo de test como entrada x, la salida $y = h(x)$ se obtiene a partir de los valores y de los k vecinos más cercanos a x. En el caso discreto se obtiene una predicción simple mediante el voto de mayoría.

Lo anterior se explica mejor en el siguiente pseudocódigo [8, 61]:

1. Entrada:
   - Conjunto de entrenamiento D con puntos $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ donde x_i es el vector de características y y_i la etiqueta de clase.
   - Un punto de prueba x_t que se quiere clasificar.
   - Un valor de k que indica el número de vecinos más cercanos a considerar.
2. Calcular la distancia entre x_t y cada punto x_i en el conjunto de entrenamiento D. Generalmente es la distancia Euclidiana:

$$d(x_t, x_i) = \sqrt{\sum_{j=1}^{d} (x_t - x_i)^2}. \tag{2.75}$$

3. Ordenar los puntos del conjunto de entrenamiento D en función de su distancia a x_t, de menor a mayor.
4. Seleccionar los k puntos de datos más cercanos a x_t (los k con menor distancia).
5. Realizar la votación mayoritaria entre las etiquetas de los k vecinos seleccionados.
6. Asignar a x_t la clase que aparezca con mayor frecuencia entre los k vecinos.
7. Salida: La etiqueta de clase predicha para x_t

Redes Neuronales

El término "red neuronal" tiene su origen en los intentos de encontrar representaciones matemáticas del procesamiento de la información en sistemas biológicos. De hecho, se ha utilizado de forma muy amplia para cubrir una amplia gama de modelos diferentes, muchos de los cuales han sido objeto de afirmaciones exageradas sobre su plausibilidad biológica [8].

Las redes neuronales utilizan funciones base que siguen la forma de:

$$y(x,w) = f\left(\sum_{j=1}^{M} w_j \phi_j(x)\right), \tag{2.76}$$

donde $f(\cdot)$ es una función de activación no lineal, $\phi_j(x)$ son funciones bases no lineales con pesos w_j. De modo que cada función base es en sí misma una función no lineal de una combinación lineal de las entradas, donde los coeficientes en la combinación lineal son parámetros adaptativos.

Esto conduce al modelo básico de red neuronal [8], que puede describirse como una serie de transformaciones funcionales. Primero se construye M combinaciones lineales de las variables de entrada $x_1, ..., x_D$ en la forma

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}, \tag{2.77}$$

con $j = 1, ..., M$, el superíndice (1) indica que corresponde a la primera capa de la red, $w_{ji}^{(1)}$ son los pesos y $w_{j0}^{(1)}$ como los sesgos. Las cantidades a_j son conocidas como activaciones [8]. Cada uno de estos elementos se transforma luego utilizando una función de activación no lineal diferenciable $h(\cdot)$ para dar:

$$z_j = h(a_j). \tag{2.78}$$

Estas cantidades corresponden a las salidas de las funciones base y se les llama como unidades escondidas [8].

Estos valores se combinan nuevamente de manera lineal para dar como resultado activaciones de la unidad de salida

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)}, \tag{2.79}$$

donde $k = 1, ..., K$ y K es el número total de salidas. Esta transformación corresponde a la segunda capa de la red [8].

Finalmente, las activaciones de la unidad de salida se transforman utilizando una función de activación adecuada para obtener un conjunto de salidas de red y_k [8]. La elección de la función de activación está determinada por la naturaleza de los datos y la distribución supuesta de las variables objetivo. Para este caso se hablará sobre la función **Soft-max** [57] la cual se usa comúnmente para problemas de clasificación donde se emplea la codificación $1-de-N$. La función reescala las salidas calculando el exponencial de las entradas a cada neurona y dividido

por la suma total de las entradas a todas las neuronas, de modo que las activaciones sumen 1 y estén todas en el rango entre 0 y 1. Esta función se puede escribir así:

$$y_k = g(a_k) = \frac{e^{a_k}}{\sum_{k=1}^{N} e^{a_k}}, \tag{2.80}$$

donde N es el número total de clases o neuronas de salida.

También existe la función de activación "*Rectified Linear Units*" (**ReLu**), propuesta por Nair y Hinton [63], la cual permite a las neuronas expresar una gama continua de valores (mayores o iguales a cero) en lugar de sólo activarse o desactivarse. Las ReLu se definen matemáticamente como

$$y_k = max(0, a_k), \tag{2.81}$$

lo anterior significa que si la entrada a es mayor que 0, la salida es a_k; de lo contrario, la salida es 0. Las ReLu son no lineales [63], pero se comportan de manera lineal para valores positivos, lo que la hace más efectivas para evitar problemas de saturación del gradiente. Éstas tienen mayor capacidad para capturar intensidades, además tienden a tener una representación más estable y preservar información de la intensidad cuando se propaga a través de múltiples capas de la red.

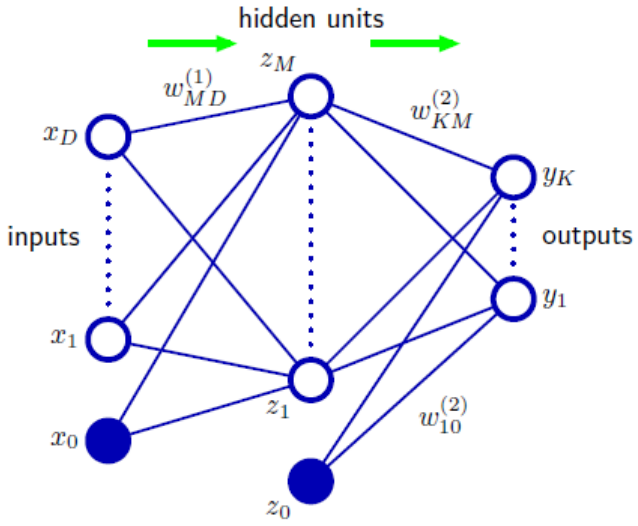


Figura 2.2: Diagrama de red para la red neuronal de dos capas. Las variables de entrada, ocultas y de salida están representadas por nodos, y los parámetros de peso están representados por enlaces entre los nodos, en los que los parámetros de sesgo se denotan por enlaces que provienen de variables de entrada y ocultas adicionales x_0 y z_0. Las flechas indican la dirección del flujo de información a través de la red durante la propagación hacia adelante. Fuente: [8, p. 228]

Se puede combinar estas diversas etapas para obtener la función de red global que tome la forma

$$y_k(x,w) = g\left(\sum_{j=1}^{M} w^{(2)}_{kj} h\left(\sum_{i=1}^{D} w^{(1)}_{ji} x_i + w^{(1)}_{j0}\right) + w^{(2)}_{k0}\right). \tag{2.82}$$

Por lo tanto, el modelo de red neuronal es simplemente una función no lineal de un conjunto de variables de entrada x_i a un conjunto de variables de salida y_k controladas por un vector w de parámetros ajustables [63]. Lo anterior se aprecia mejor en la figura 2.2.

Las raíces teóricas de las redes neuronales de alimentación hacia adelante están dadas por el teorema de representación de Kolmogorov-Arnold de funciones multivariadas[17]. De manera notable, mostraron cómo las redes neuronales, con una capa oculta, son aproximadores universales de funciones no lineales [17].

Para el ajuste de los parámetros de la red, se emplea la minimización de la entropía cruzada como función de pérdida (2.83). Para poder optimizar los resultados de una red neuronal es común que para problemas que tienen múltiples clases con respecto a la variable objetivo se usen probabilidades en vez de resultados booleanos como un "sí" o un "no" [61]. Por ello, la minimización de la entropía cruzada hace que la red emita estimaciones de probabilidad (las de máxima verosimilitud), se define como

$$-\sum_{d\in D}\left(t_d \log O_d + (1-t_d)\log\left(1-O_d\right)\right), \tag{2.83}$$

donde O_d es la estimación de probabilidad que la red genera para el ejemplo de entrenamiento d, y t_d es el valor objetivo (1 ó 0) para el ejemplo de entrenamiento d.

La entropía cruzada mide la discrepancia entre la distribución predicha y la verdadera, permitiendo una penalización significativa cuando las predicciones son erróneas [61]. Esto ayuda a mejorar la precisión del modelo al enfatizar la clasificación correcta de cada clase.

Además, para optimizar el proceso de aprendizaje y minimizar la entropía cruzada, se utiliza el optimizador **Adam** (Adaptive Moment Estimation) [48], que combina las ventajas de los métodos de optimización basados en momentos, adaptando la tasa de aprendizaje para cada parámetro de forma individual.

El algoritmo de **Adam**, en general, se resume en los siguientes pasos [48]:

- Combina las ideas de momento y adaptación de la tasa de aprendizaje basándose en momentos del gradiente.
- Mantiene dos vectores de momento: uno para la media (primer momento) y otro para la varianza (segundo momento no centrado).
- La actualización de parámetros se ajusta con base en estos momentos, aplicando una corrección de sesgo para asegurar una adaptación más precisa durante las primeras etapas del entrenamiento.

Más a detalle de lo anterior se puede revisar en el algoritmo que escribe [48, pp. 2] en su artículo "*Adam: A method for stochastic Optimization*" :

Algorithm 1 Adam, our proposed algorithm for stochastic optimization

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0,1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
$m_0 \leftarrow 0$ (Initialize 1^{st} moment vector)
$v_0 \leftarrow 0$ (Initialize 2^{nd} moment vector)
$t \leftarrow 0$ (Initialize timestep)
while θ_t not converged **do**
    $t \leftarrow t+1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t$ (Update biased 1^{st} moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot g_t^2$ (Update biased 2^{nd} raw moment estimate)
    $\hat{m}_t \leftarrow m_t / (1-\beta_1^t)$ (Compute bias-corrected 1^{st} moment estimate)
    $\hat{v}_t \leftarrow v_t / (1-\beta_2^t)$ (Compute bias-corrected 2^{nd} raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)$ (Update parameters)
end while
return θ_t (Resulting parameters)

Lo importante en los resultados de Adam es que sus pasos de optimización son adaptativos, por ello maneja bien los gradientes dispersos y datos estacionarios.

La descripción previa de la arquitectura, funciones de activación, función de pérdida y optimizador no responde únicamente a un interés teórico, sino a la justificación de los parámetros seleccionados para este estudio. Se eligió la función de activación ReLU en las capas ocultas por su eficiencia computacional y capacidad de mitigar la saturación del gradiente, mientras que en la capa de salida se empleó la función Softmax, adecuada para problemas de clasificación multiclase. Asimismo, se utilizó la entropía cruzada como función de pérdida por su capacidad de penalizar fuertemente las predicciones incorrectas y generar estimaciones probabilísticas consistentes. Finalmente, la elección del optimizador Adam se fundamenta en su estabilidad, adaptabilidad de la tasa de aprendizaje y buen desempeño en problemas de gran escala. De este modo, la configuración adoptada combina teoría con criterios prácticos orientados a maximizar la precisión y la eficiencia en la clasificación de esta investigación.

2.5. Métricas Para Evaluar el Resultado de Clasificación

Para problemas de clasificación, se determina la calidad de los resultados con la **matriz de confusión** [57].

La matriz de confusión es una matriz cuadrada que contiene todas las clases posibles tanto en la dirección horizontal como en la vertical. Se enumeran las clases en la parte superior como las salidas predichas y en la izquierda como los objetivos.

Hágase la suposición de tener una matriz con renglones y columnas (i, j), en i se colocarán los casos en donde se clasificaron esa misma clase (objetivo), pero fueron asignados a la otra

clase j por el modelo en cuestión. Cualquier valor en la diagonal principal es una respuesta correcta.

Tomando el ejemplo de [57, pp. 22]: Supongamos que se tienen tres clases: C_1, C_2 y C_3. Ahora se cuentan cuántas veces la salida fue C_1 cuando el objetivo era C_1, luego C_2, y así sucesivamente hasta llenar la tabla:

Objetivos	C_1	C_2	C_3
C_1	5	1	0
C_2	1	4	1
C_3	2	0	4

En esta tabla mayoría de los ejemplos fueron clasificados correctamente, pero dos ejemplos de la clase C_3 se clasificaron erróneamente como C_1, y así sucesivamente.

El anterior ejemplo sirve para poder explicar las métricas de precisión. Éstas se basarán en la siguiente tabla donde se consideran las salidas de las clases:

Positivos Verdaderos (TP)	**Falsos Positivos (FP)**
Falsos Negativos (FN)	**Verdaderos Negativos (TN)**

Donde un **positivo verdadero** (TP) es una observación correctamente clasificada en la clase 1, mientras que un **falso positivo** (FP) es una observación incorrectamente asignada a la clase 1. Los ejemplos negativos (verdaderos y falsos) se clasifican en la clase 2 [57].

El *accuracy* se define como la suma de los positivos verdaderos y los negativos verdaderos, dividida entre el total de ejemplos [57]:

$$\text{Accuracy} = \frac{\#TP + \#TN}{\#TP + \#FP + \#TN + \#FN}. \tag{2.84}$$

El problema con Accuracy es que no nos dice todo sobre los resultados, ya que convierte cuatro números en solo uno. Existen dos pares complementarios de medidas que pueden ayudar a interpretar el rendimiento de un clasificador, a saber, **Precision** y **Recall**, y su combinación en el **F1-score**.

En problemas multiclase, como en este estudio, estas métricas se calculan de manera independiente para cada clase $c \in C$ y luego se promedian (`average='macro'` en *scikit-learn* [65]). Las definiciones son:

Precision macro:

$$\text{Precision}_{macro} = \frac{1}{|C|} \sum_{c \in C} \frac{TP_c}{TP_c + FP_c}. \tag{2.85}$$

Recall macro:

$$\text{Recall}_{macro} = \frac{1}{|C|} \sum_{c \in C} \frac{TP_c}{TP_c + FN_c}. \tag{2.86}$$

F1-score macro:

$$\text{F1}_{macro} = \frac{1}{|C|} \sum_{c \in C} 2 \cdot \frac{\text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c}. \tag{2.87}$$

Donde $|C|$ es el número total de clases.

Este promedio simple otorga el mismo peso a cada clase, lo cual resulta apropiado en este trabajo dado que las clases están balanceadas. Una observación es que, naturalmente, **F1-score** es una métrica para la evaluación de clases binarias, sin embargo, la implementación en *scikit-learn* [65] permite utilizarla en problemas multiclase.

Cuando se utiliza la validación cruzada k-fold, los datos disponibles se dividen en k particiones (o subconjuntos) de tamaño igual, y se realizan k experimentos de evaluación separados [44].

- En el primer experimento, los datos en el primer fold se usan como conjunto de prueba, y los datos en los otros $k-1$ folds se utilizan como conjunto de entrenamiento.
- Se entrena un modelo usando el conjunto de entrenamiento, y se registran las medidas de desempeño relevantes en el conjunto de prueba.
- Luego, se realiza una segunda evaluación usando los datos en el segundo fold como conjunto de prueba y los datos en los otros $k-1$ folds como conjunto de entrenamiento.

Este proceso continúa hasta que se han llevado a cabo k experimentos de evaluación, y se han registrado k conjuntos de medidas de desempeño. Finalmente, los k conjuntos de medidas de desempeño se agregan para dar un conjunto general de métricas de desempeño [44], en este caso serán las métricas de *Accuracy, Precision, Recall y F1-Score*.

Adicionalmente, la **desviación estándar** de estas métricas se emplea como una medida de incertidumbre, permitiendo cuantificar la variabilidad del desempeño del modelo a través de las diferentes particiones. Su expresión matemática es [41]:

$$\sigma = \sqrt{\frac{1}{k}\sum_{i=1}^{k}(m_i - \bar{m})^2}, \tag{2.88}$$

donde m_i representa el valor de la métrica (Accuracy, Precision, Recall o F1-score) en la i-ésima iteración, $\bar{m}$ es el promedio de los valores obtenidos y k es el número de particiones definidas en la validación cruzada. Aunque el valor de k puede variar, en la práctica la validación cruzada con $k = 10$ es una de las variantes más utilizadas [14]. En consecuencia, en las evaluaciones posteriores se aplicará este esquema con $k = 10$.

2.6. Pruebas estadísticas

Prueba de normalidad de Shapiro-Wilk

La **prueba de Shapiro-Wilk** es un contraste estadístico de bondad de ajuste diseñado específicamente para evaluar si un conjunto de datos procede de una distribución normal [72]. En esta prueba se plantea como *hipótesis nula* (H_0) que la muestra analizada proviene de una población con distribución normal, frente a la *hipótesis alternativa* (H_1) que indica que los datos no siguen la normalidad. Shapiro y Wilk desarrollaron este test como un *"An analysis of variance test for normality"*, cuyo estadístico W evalúa la concordancia de la distribución muestral con la forma teórica de una normal [72].

En términos generales, el estadístico W de Shapiro-Wilk se calcula a partir de los datos ordenados y su media muestral, comparando la variabilidad que presentan los datos con la esperada bajo normalidad. Valores de W cercanos a 1 indican un ajuste bueno a la distribución normal, mientras que valores significativamente menores sugieren desviaciones de la normalidad. Para determinar si la desviación es estadísticamente significativa, se compara el estadístico W calculado con los valores críticos tabulados o se calcula su correspondiente valor p. Un valor p inferior al nivel de significación (por ejemplo, $\alpha = 0.05$) conlleva rechazar H_0 y concluir que los datos no se distribuyen normalmente, es decir, se viola la suposición de normalidad [75]. En cambio, un valor p alto indicaría que no hay evidencia significativa de desviación de la normalidad, por lo que se puede considerar válido el supuesto de normalidad de los datos.

Prueba de homogeneidad de varianzas de Levene

La **prueba de Levene** se emplea para verificar la suposición de *homogeneidad de varianzas* entre varios grupos o tratamientos [75]. Esta suposición es fundamental en pruebas paramétricas como el análisis de varianza (ANOVA), ya que garantiza que la variabilidad residual sea comparable entre grupos [12]. El test de Levene contrasta la *hipótesis nula* de que todas las poblaciones tienen varianzas iguales, frente a la *hipótesis alternativa* de que al menos una difiere [75].

Operativamente, la prueba de Levene calcula, para cada observación, la desviación respecto a la media (o mediana) de su grupo y luego realiza un ANOVA unifactorial sobre estas desviaciones absolutas. Un resultado no significativo (valor p grande) sugiere que no hay evidencia para descartar la igualdad de varianzas entre grupos, cumpliéndose por tanto el supuesto de homogeneidad [75]. Por otro lado, un valor p menor que α indica que al menos un grupo tiene varianza distinta, es decir, se detecta heterogeneidad de varianzas. En tal caso, el supuesto de homogeneidad estaría violado y las conclusiones de pruebas como ANOVA podrían verse comprometidas [12].

Elección de ANOVA o prueba de Friedman según los supuestos

Una vez evaluados los supuestos de normalidad y homogeneidad de varianzas con las pruebas anteriores, se debe decidir qué análisis estadístico es adecuado para comparar las medias de k grupos o condiciones en el estudio. Existen dos escenarios principales [75]:

- Si los datos **no violan** de forma grave la normalidad y se cumple la homogeneidad de varianzas, se puede aplicar un **análisis de varianza (ANOVA) paramétrico**.
- Si por el contrario **se violan** uno o ambos supuestos (distribución notoriamente no normal o varianzas heterogéneas), es más seguro emplear una prueba **no paramétrica**. En este caso, para diseños con medidas repetidas o grupos relacionados, la alternativa apropiada es la **prueba de Friedman**.

Esta decisión se basa en consideraciones de validez y potencia estadística. El ANOVA es conocido por ser relativamente mejor frente a pequeñas desviaciones de la normalidad o varianzas ligeramente desiguales, especialmente si los tamaños muestrales son iguales, de modo que en muchos casos menores violaciones no invalidan sus resultados [75]. Sin embargo, ante

violaciones marcadas de los supuestos, la confiabilidad del ANOVA puede verse comprometida, aumentando por ejemplo el riesgo de errores de Tipo I cuando hay heterogeneidad de varianzas. En tales situaciones, los estadísticos no paramétricos ofrecen una alternativa que no requiere dichas suposiciones de normalidad ni igualdad de varianza [75].

ANOVA (paramétrica)

Para aplicar ANOVA correctamente se asume que: **a)** cada muestra es extraída al azar de su población, **b)** las distribuciones de la variable en cada población son aproximadamente normales, y **c)** las varianzas poblacionales son iguales (homocedasticidad) [75]. Cuando estos supuestos se satisfacen, el estadístico F de Snedecor es apropiado para probar la igualdad de medias. El estadístico F se define como la razón entre la variabilidad entre grupos y la variabilidad dentro de los grupos:

$$F = \frac{MS_{entre}}{MS_{intra}} \tag{2.89}$$

donde MS_{entre} (mean square between) es la varianza de las medias de grupo, y MS_{intra} (mean square within) es la varianza residual promedio dentro de los grupos. Bajo H_0, ambas estimaciones de varianza reflejan la misma cantidad de dispersión y por tanto F tiende a tomar valores alrededor de 1. En cambio, si alguna media difiere sustancialmente, la variabilidad entre grupos será mayor en relación con la variabilidad interna, dando lugar a un F significativamente mayor que 1. En la práctica, se calcula F y se contrasta con el valor crítico de la distribución F de Fisher con grados de libertad $(k-1,,N-k)$ (siendo k el número de grupos y N el total de datos). Alternativamente se obtiene el valor p asociado. Un p pequeño (por debajo de α) indica que F observado es demasiado grande para atribuirlo al azar, llevando a rechazar H_0 y concluir que existen diferencias significativas entre las medias de los grupos [75]. Por el contrario, si p resulta mayor que α, no se rechaza H_0 y se considera que los datos son consistentes con medias iguales [75].

Al reportar un ANOVA, es necesario presentar los componentes de la **tabla ANOVA**. Por ejemplo, un resultado podría informarse como: $F(3,36) = 5.27$, $p = 0.004$, indicando que con 3 y 36 grados de libertad el estadístico F obtenido fue 5.27 y es significativo al nivel del 0.4 %. Este resultado implicaría evidencias para rechazar que todas las μ_i son iguales, sugiriendo que al menos una condición difiere de las otras en su media.

Si las pruebas de Shapiro-Wilk y Levene fueron no significativas, se puede afirmar que no hubo violaciones importantes de normalidad u homogeneidad de varianzas [75], reforzando la validez del ANOVA realizado.

Un ANOVA significativo indica que existen diferencias entre las medias, pero no especifica **entre qué pares de grupos** se encuentran esas diferencias. Por ello, suele ser necesario realizar **comparaciones post hoc** o **comparaciones múltiples** una vez rechazada H_0 en el ANOVA. Un método sencillo consiste en llevar a cabo pruebas t de Student para cada par de medias, conocidas también como pruebas LSD (*Least Significant Difference*) de Fisher [75].

Al presentar los resultados de las comparaciones múltiples, para cada par considerado, se reporta el estadístico t, sus grados de libertad y el valor p asociado. Este tipo de aclaración permite identificar específicamente qué grupos difieren tras un ANOVA significativo.

Prueba de Friedman (no paramétrica)

Fue propuesta por Friedman como un método basado en rangos para evitar la asunción de normalidad inherente al ANOVA paramétrico [75]. En esencia, Friedman evalúa si k tratamientos o condiciones medidas en bloques (por ejemplo, los mismos sujetos bajo diferentes tratamientos) provienen de la misma distribución [28].

La *hipótesis nula* establece que las distribuciones de los k tratamientos son idénticas (lo que implica que las medianas de todos los grupos son iguales), mientras que la *hipótesis alternativa* postula que al menos uno de los tratamientos tiende a producir valores significativamente mayores o menores que al menos otro [28].

El diseño de Friedman considera un conjunto de n bloques, donde cada bloque aporta una observación bajo cada una de las k condiciones o tratamientos. Dentro de cada bloque, los datos se ordenan del 1 al k. Si hay empates, se asigna el promedio de los rangos correspondientes. Luego, se suman los rangos asignados a cada condición a través de los n bloques, obteniendo la suma de rangos R_j para el tratamiento j-ésimo. Si bajo H_0 no hay diferencias reales entre tratamientos, uno esperaría que las sumas de rangos R_j sean aproximadamente iguales. Por el contrario, si un tratamiento tiende a producir valores sistemáticamente más bajos o más altos, acumulará una suma de rangos menor o mayor que las demás [28, 75].

El estadístico de Friedman (Q) se basa en comparar la variabilidad de las sumas de rangos observada contra la esperada bajo H_0 [28]. Una fórmula común es:

$$Q = \frac{12}{nk(k-1)} \sum_{j=1}^{k} R_j^2 - 3n(k+1), \tag{2.90}$$

donde n es el número de bloques y R_j la suma de rangos del tratamiento j. Bajo la hipótesis nula (cuando n es suficientemente grande, $n > 10$ aproximadamente), Q sigue aproximadamente una distribución χ^2 con $k-1$ grados de libertad [75]. Se determina entonces un valor p asociado a Q. Si $p < \alpha$, se rechaza H_0 concluyendo que existen diferencias significativas entre los tratamientos. En caso contrario, no se evidencia diferencia alguna y se retiene H_0, interpretando que los datos no muestran ordenamientos sistemáticos distinguibles entre condiciones.

Al presentar los resultados de Friedman, se debe indicar el valor del estadístico (a menudo denotado Q), los grados de libertad ($k-1$) y el valor p correspondiente. Por ejemplo: "La prueba de Friedman reveló diferencias significativas entre las condiciones ($Q(3) = 11.08$, $p = 0.011$)". Esto indicaría que con 4 condiciones ($k = 4$) y, digamos, $n = 10$ sujetos, el estadístico calculado fue 11.08 excediendo el valor crítico de $Q(3)$ a un nivel de significación del 1.1 %.

Al igual que con el ANOVA, un resultado global significativo en Friedman lleva a preguntar cuáles condiciones difieren específicamente. Para ello existen procedimientos de comparaciones múltiples adaptados a datos de rangos. Un enfoque común es realizar **comparaciones por pares** empleando la distribución normal como aproximación: esencialmente se compara la diferencia entre las sumas de rangos de dos condiciones cualquiera con una **diferencia mínima significativa** obtenida bajo H_0 [75].

Se acompaña cada comparación con su estadístico, por ejemplo, se puede convertir la diferencia de rangos a un valor z o T de Wilcoxon equivalente, y su valor p ajustado [75].

Capítulo 3

Metodología

3.1. Análisis exploratorio y descripción de los datos

Origen y Estructura de los Datos

La base de datos usada fue la misma que se usó en los artículos de [36] y [35], la cual se obtuvo de un investigador de la Universidad de Chicago, Dacheng Xiu. Está disponible de manera libre en su página web https://dachxiu.chicagobooth.edu/. Esta base de datos contiene las acciones individuales del CRSP para todas las bolsas de NYSE, AMEX y NASDAQ [36].

La base de datos numérica consta de 94 características (como el tamaño, momentum, volatilidad, y medidas de liquidez) cuyo significado se encuentra en la tabla (A.1). Como se mencionó en la sección 1.1 de Introducción (págs. 14-15), estas características son variables fundamentales y especulativas. El total de características se puede dividir de la siguiente manera:

- 61 características actualizadas anualmente (Compustat),
- 13 características actualizadas trimestralmente (Compustat),
- 20 características actualizadas mensualmente (Center for Research in Security Prices (CRSP)).

Además contiene otras 3 características extra, que es el número de firma (*permno*), la fecha del registro según el año, mes y día ($DATE$) y el retorno (RET). Este conjunto de datos incluye aproximadamente 30,000 acciones de EE.UU, desde 1926 hasta finalizar el año 2020.

Para esta investigación se recortará la base de datos empezando por el 31 de enero de 2001 hasta el 31 de diciembre de 2020 porque los últimos 20 años de la base de datos no tienen tantas valores faltantes en las características y para que las ejecuciones de cada modelo no sean tan demoradas, recordando que se aplicará validación cruzada con $K = 10$, de esta manera el modelo se entrenará y a su vez se probará en eventos destacados en el mercado financiero como lo es la crisis de 2008 y la pandemia de 2019-2020. Además, se tiene la idea que las firmas actuales no son afectadas por las firmas de hace mucho tiempo atrás. Por lo tanto, el conjunto de datos contiene las siguientes dimensiones: *filas 1487704 con 95 columnas (incluyendo el retorno RET)*. Además, se harán otras dos bases de datos, siguiendo la metodología de [36], donde uno tendrá las mejores 1000 firmas por mes ordenadas por el tamaño de la empresa ($mvel1$) de

forma ascendente, y a su vez se hará el otro conjunto de datos con las peores 1000 firmas por mes ordenadas por el tamaño de la empresa de forma descendente.

En resumen, las bases de datos a usar tienen las siguientes dimensiones:

- All: filas 1487705 y 95 columnas,
- Top: filas 240000 y 95 columnas,
- Bottom: filas 240000 y 95 columnas.

Descripción

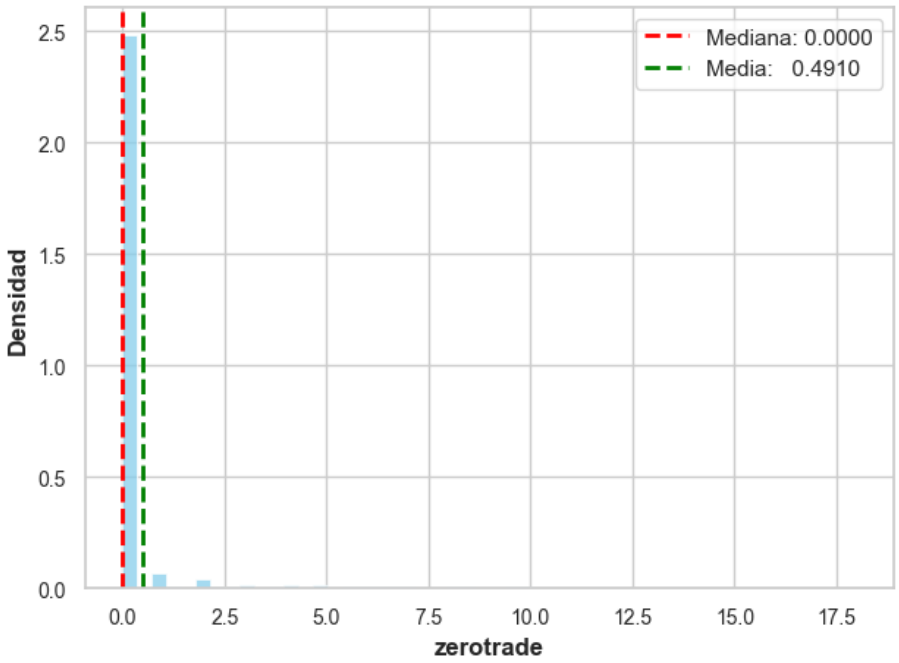
La base de datos tiene las siguientes especificaciones:

- Tiene algunas firmas (*permno*) que se conservan a lo largo de los 20 años tomados. Sin embargo, tiene otras que desaparecen en ese lapso de tiempo o son nuevas en el mismo tiempo.
- No todas las firmas reportaban sus características, es por ello que habrá muchas filas vacías, sin embargo, no se recomienda quitar esas filas, esto por la pérdida de información y pocos ejemplos de firmas con sus características completas.
- Las firmas que sean nuevas, no podrán reportar algunas características esto porque son datos que se obtienen a lo largo del tiempo, ya sea de manera anual, trimestral o mensual.
- No se puede considerar como series de tiempo esta base de datos porque son distintas firmas, algunas desaparecen a lo largo del tiempo y además, muchas presentan bastantes datos vacíos. Por lo cual, se considerará como una base de datos estática.
- La única manera de considerar esta base de datos como series temporales, sería hacer un análisis muy específico para una sola *permno*, lo cual limitaría toda la base de datos.

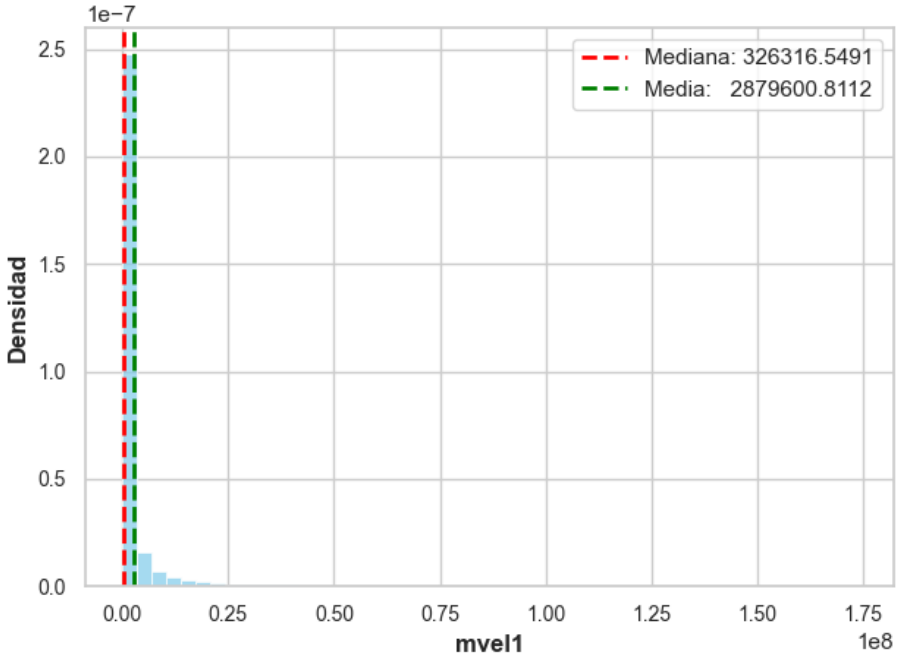
3.2. Primera Exploración y Limpieza

Para hacer una debida estimación de los valores faltantes de la base de datos, se necesita saber cómo están distribuidos en comparación a algunas medidas estadísticas como lo es la mediana y media, lo anterior se aprecia en las Figuras (3.1) y (3.2). Tomando como ejemplo estas 4 características, se observa que, la mediana se ajusta mejor a la mayoría de concentración de los datos, además, la media tiende a inclinarse por los valores atípicos que todas las características tienen. En los gráficos mencionados los datos faltantes se ignoran en las barras producidas para la distribución.

Por ello, se usará la **mediana** como imputación para los datos faltantes de las bases de datos, de esa manera se generará menos ruido debido a que la mediana no está sesgada por los valores atípicos, sólo toma en cuenta la cantidad de datos.

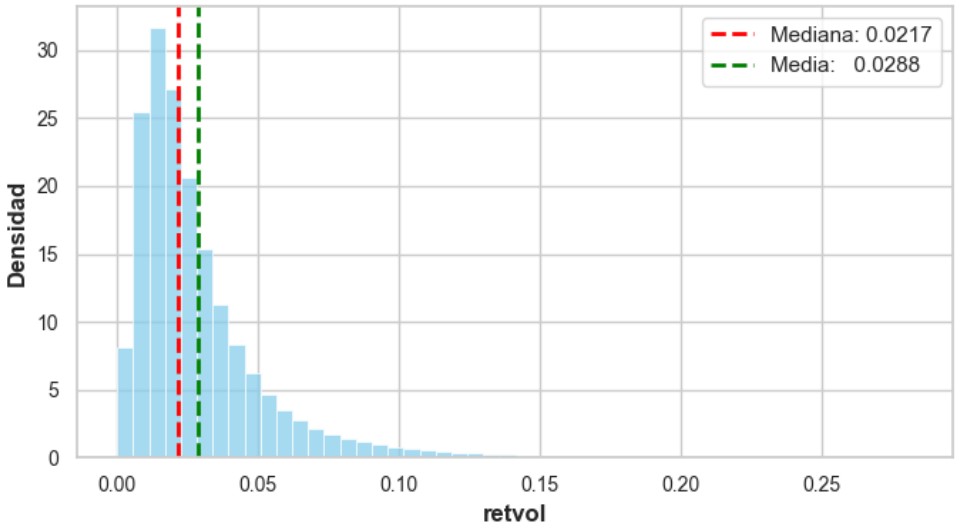


(a) Gráfico de Distribución de la Característica *zerotrade*

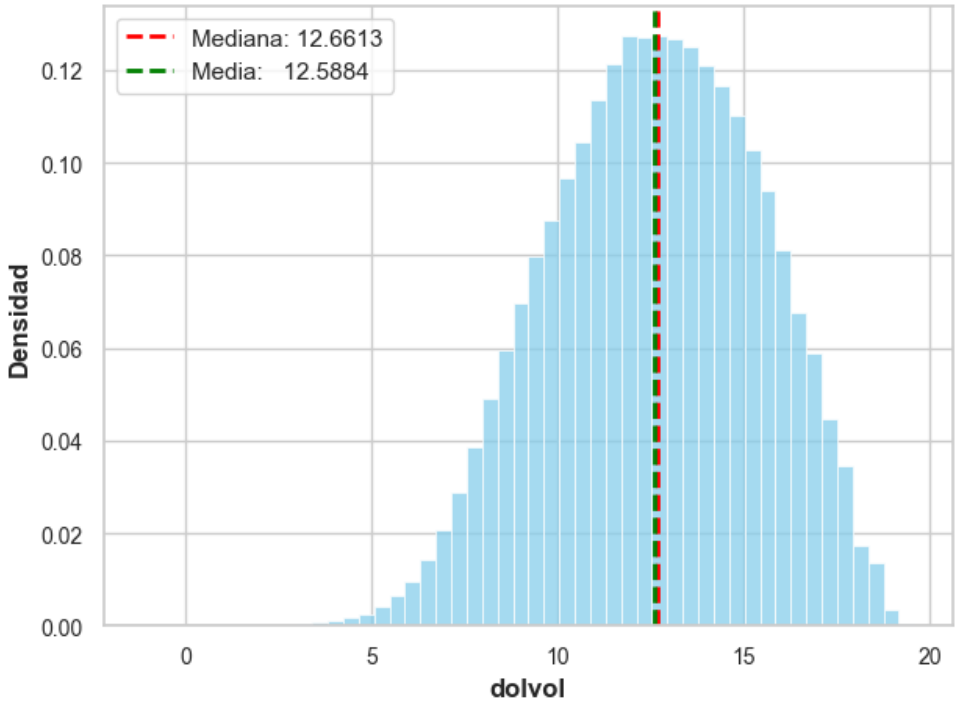


(b) Gráfico de Distribución de la Característica *mvel*1

Figura 3.1: Análisis Estadístico de las Características *zerotrade* y *mvel*1 sin imputación de valores.



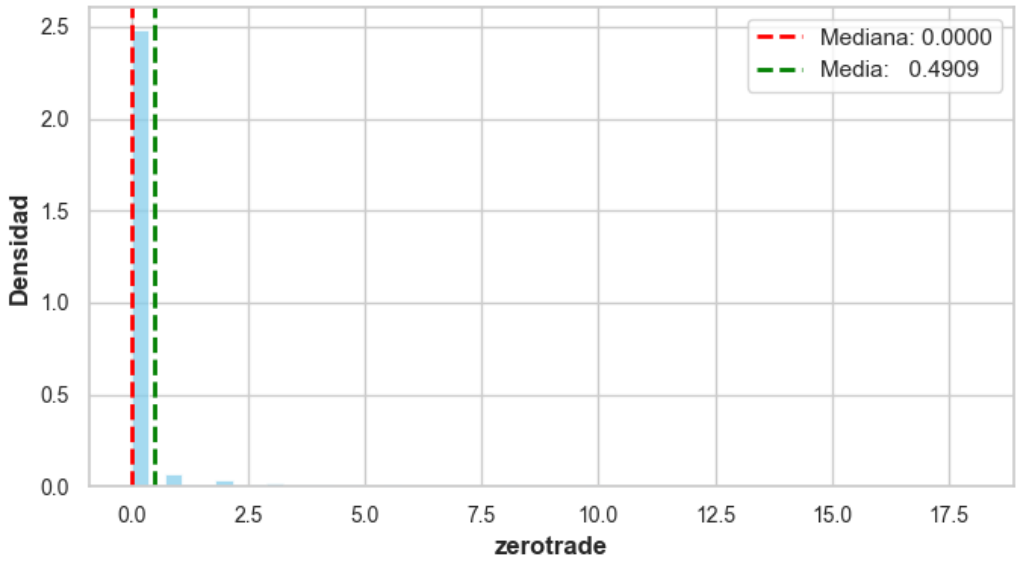
(a) Gráfico de Distribución de la Característica *retvol*



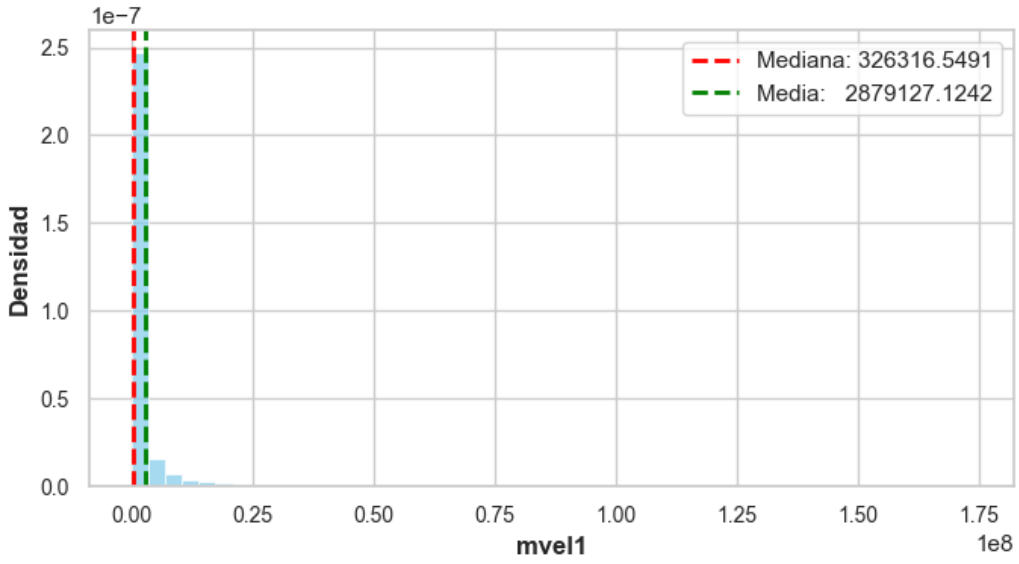
(b) Gráfico de Distribución de la Característica *dolvol*

Figura 3.2: Análisis Estadístico de las Características *retvol* y *dolvol* sin imputación de valores.

Ahora se verán los gráficos de distribución con la imputación, se aprecia que la mediana no modificó significativamente la estadística de los datos (3.3) y (3.4).

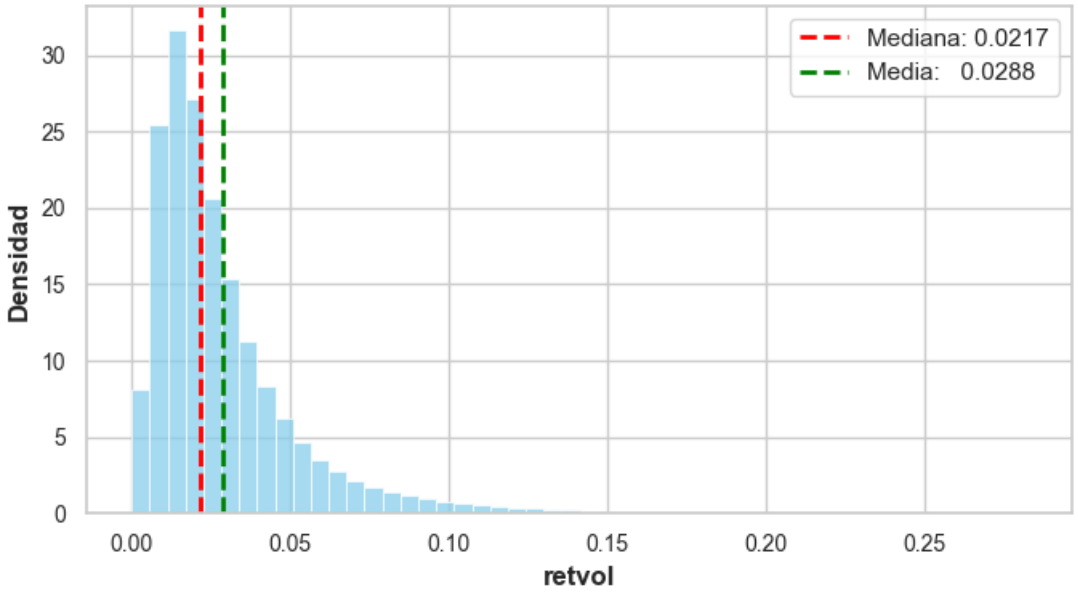


(a) Gráfico de Distribución de la Característica *zerotrade*

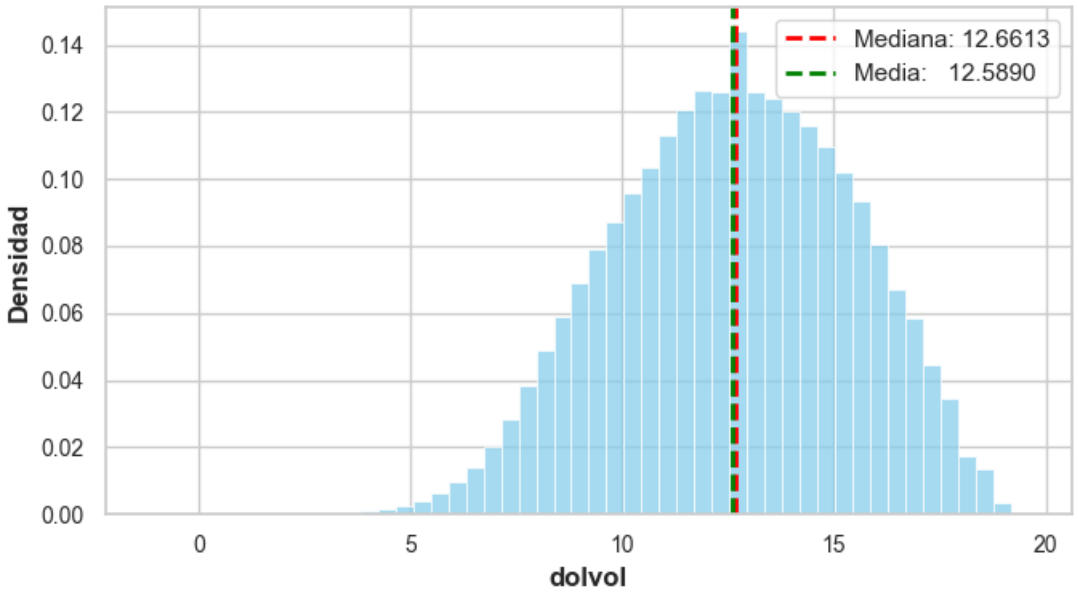


(b) Gráfico de Distribución de la Característica *mvel*1

Figura 3.3: Análisis Estadístico de las Características *zerotrade* y *mvel*1 con imputación de valores.



(a) Gráfico de Distribución de la Característica *retvol*



(b) Gráfico de Distribución de la Característica *dolvol*

Figura 3.4: Análisis Estadístico de las Características *retvol* y *dolvol* con imputación de valores.

3.3. Segunda Exploración

En la sección 3.1 de Metodología (página 60) se había mencionado la parte de que las firmas desaparecían o aparecían por primera vez en el conjunto de datos o, en cambio, las que se mantuvieron todo el conjunto de datos. Por ello, es importante identificar qué tanta desviación

tendrán nuestros datos al tipo de firmas que estén.

En nuestro conjunto de años reducido (20 años), se encontró los siguientes tipos de firmas:

- Firmas que sólo aparecieron un año: Aquellas que aparecieron sólo un año, ya sea que en menos de un año quebraron o fueron nuevas firmas en el último año tomado (2020). Se hallaron: **1611** firmas.
- Firmas que aparecieron más de un año, pero menos de los 20 años tomados: Aquellas que iniciaron en el período del 2001, pero desaparecieron antes del 2020, se puede entender también que son aquellas que aparecieron dentro de los 20 años y siguen en el conjunto de datos o ya no siguen en el mismo. Se hallaron: **11975** firmas.
- Firmas que prevalecieron los 20 años: Normalmente, éstas son firmas muy grandes, las cuales no nada más han existido durante 20 años, sino más de ellos Se hallaron: **2114** firmas.

Lo anterior se puede ver mejor en la siguiente gráfica (Fig. 3.5) donde se aprecia la aparición, desaparición y permanencia de firmas:

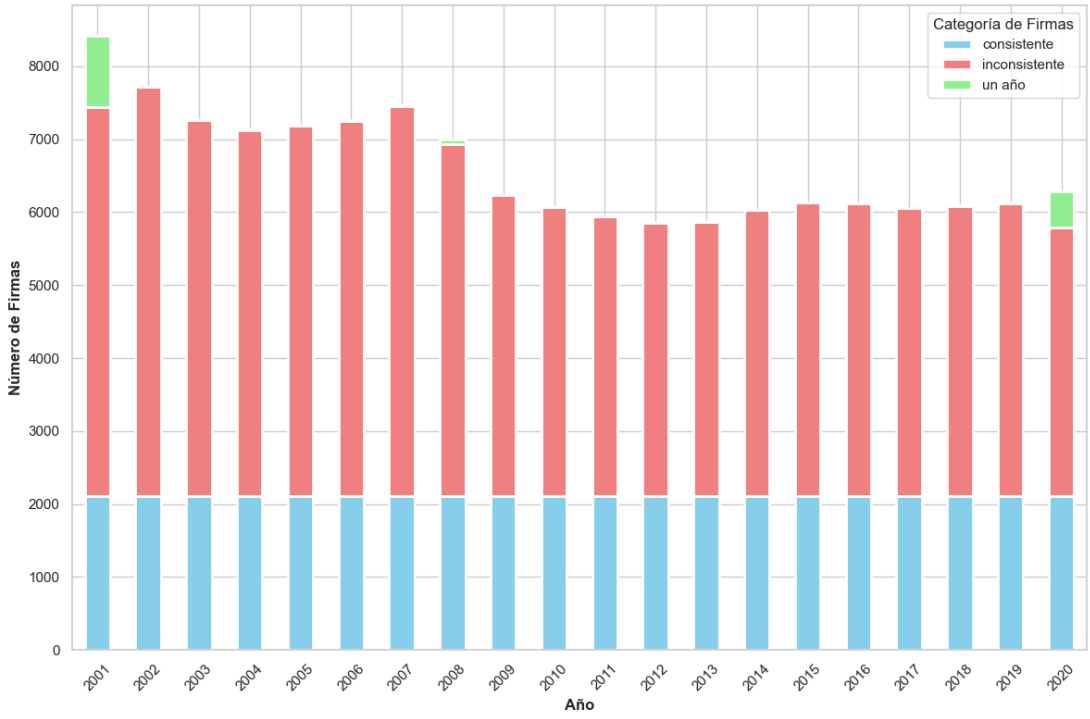


Figura 3.5: Evolución de las Firmas a lo Largo del Tiempo

Lo anterior da a entender que las firmas que tienen mayor tamaño son constantes a lo largo del tiempo, y que con nuestro conjunto de datos no se lidiará con tantas firmas muy pequeñas que sólo aparecieron un único año.

La figura (Fig. 3.5) ayuda a comprender con qué irregularidades de firmas los modelos trabajarán. En esta investigación, se usarán todas las firmas de esa figura, y como aplicación se trabajará con ejemplos de firmas consistentes en un gran período de tiempo.

3.4. Determinación del Umbral (τ)

Marcos Lopez de Prado [67] propone una manera de hacer un etiquetado en series de tiempo financieras. Una observación X_i (un registro o instancia de la serie de tiempo) está asignada a una etiqueta $y_i \in \{-1, 0, 1\}$,

$$y_i = \begin{cases} -1 & \text{if } r_{t_i,0,t_i,0+h} < -\tau \\ 0 & \text{if } |r_{t_i,0,t_i,0+h}| \leq \tau \\ 1 & \text{if } r_{t_i,0,t_i,0+h} > \tau \end{cases} \tag{3.1}$$

donde τ es la constante predefinida como umbral, es decir, el punto de corte, $t_{i,0}$ es el índice de la barra (una barra es una unidad de tiempo o de cotización en la serie de precios) inmediatamente después de que X_i tome lugar, $t_{i,0} + h$ es el índice de la h_{th} barra después de $t_{i,0}$, h es el horizonte temporal en barras, y $r_{t_i,0,t_i,0+h}$ es el precio del retorno durante un horizonte de barras h,

$$r_{t_i,0,t_i,0+h} = \frac{P_{t_i,0+h}}{P_{t_i,0}} - 1 \tag{3.2}$$

$P_{t_i,0+h}$ el precio del retorno después de un cierto horizonte de tiempo h y $P_{t_i,0}$ el precio inicial en el momento $t_i, 0$.

Se optará por hacer 3 y 5 clases siguiendo el enfoque de Marcos Lopez de Prado [67], estas para que cubran casos anómalos de la distribución de los retornos, se muestra la figura que muestra la distribución de los retornos (3.6).

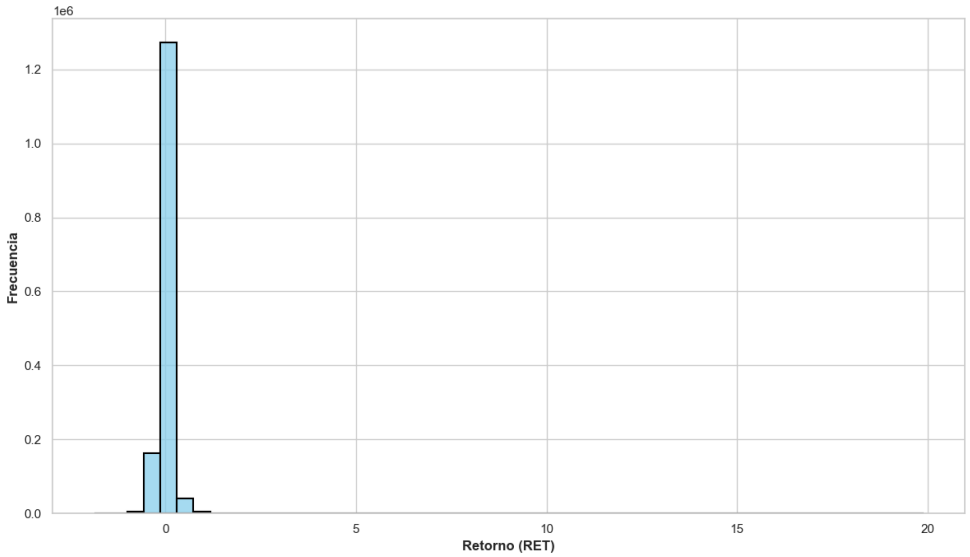


Figura 3.6: Histogramas de los Retornos a lo Largo de 20 Años con Valores Atípicos.

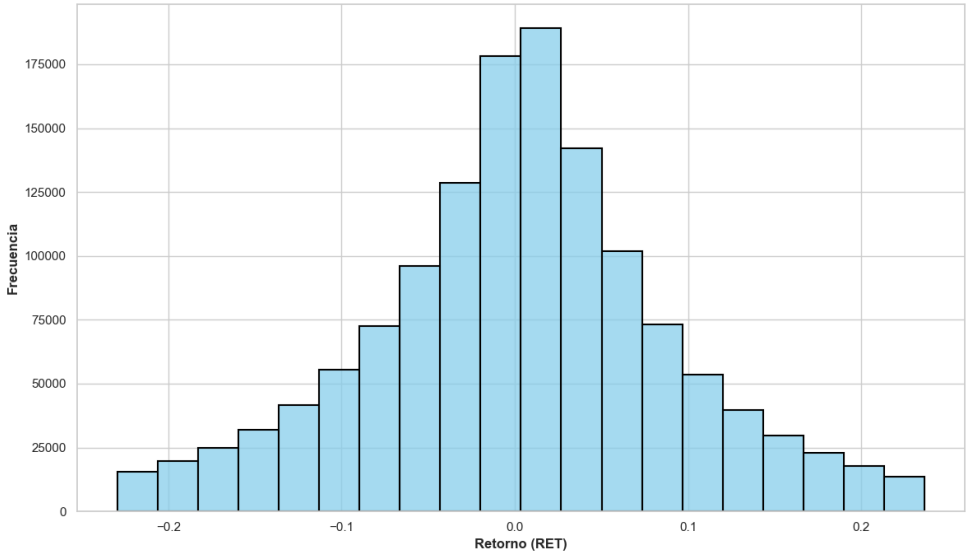


Figura 3.7: Histogramas de los Retornos a lo Largo de 20 Años sin Valores Atípicos.

De la figura anterior (3.7), los retornos están distribuidos, en su mayoría, en el intervalo de $(-0.1, 0.1)$.

Se propone la idea de crear clases que tengan la misma cantidad de datos para incluir a todos y estén balanceadas.

- **Para el caso de 5 clases**: Dos clases estarían en los extremos para identificar los casos atípicos, mientras que las otras verían si caen en el caso positivo o negativo del intervalo de $(-0.1, 0.1)$.
- **Para el caso de 3 clases**: Dos clases estarían en los valores extremos y medios de los casos positivos y negativos, y una clase que considere ambos casos.

Haciendo esas partición los datos quedarían distribuidos de la siguiente manera:
Para todo el conjunto de datos con 5 etiquetas:

Tabla 3.1: Tabla del rango de cada clase (5) de All, su diferencia y números de datos. $|\Delta|$ es la diferencia del intervalo de cada clase.

Clases	Intervalo	$\lvert\Delta\rvert$	# de datos
-2	(-1.8708e+00, -7.5095e-02)	1.7957	297542
-1	(-7.5095e-02, -1.3181e-02)	0.0619	297544
0	(-1.3181e-02, 2.3210e-02)	0.0363	297538
1	(2.3210e-02, 8.1101e-02)	0.0578	297540
2	(8.1101e-02, 1.9883e+01)	19.8024	297541

Cada intervalo de las clases se puede ver como el umbral τ que se explica en la ecuación (3.1), para el caso de 5 clases sólo se colocó una clase intermedia en cada intervalo. En la

tabla (3.1) se aprecian los valores. Los intervalos para el subconjunto de datos denominado Top quedaría así (3.2):

Tabla 3.2: Tabla del rango de cada clase de Top (5), su diferencia y números de datos. $|\Delta|$ es la diferencia del intervalo de cada clase.

Clases	Intervalo	$\|\Delta\|$	# de datos
-2	(-9.9676e-01, -5.3305e-02)	0.9434	48000
-1	(-5.3305e-02, -6.607e-03)	0.0466	48001
0	(-6.607e-03, 2.7135e-02)	0.0337	47999
1	(2.7135e-02, 7.0219e-02)	0.0430	48002
2	(7.0219e-02, 1.6762e+00)	1.6059	47998

Y para el subconjunto denominado Bot (3.3):

Tabla 3.3: Tabla del rango de cada clase de Bot (5), su diferencia y números de datos. $|\Delta|$ es la diferencia del intervalo de cada clase.

Clases	Intervalo	$\|\Delta\|$	# de datos
-2	(-1.8708e+00, -1.1229e-01)	1.7585	48004
-1	(-1.1229e-01, -2.7439e-02)	0.0848	47998
0	(-2.7439e-02, 1.6214e-02)	0.0436	47999
1	(1.6214e-02, 9.3938e-02)	0.0777	47999
2	(9.3938e-02, 1.5984e+01)	15.8905	48000

Para todo el conjunto de datos con 3 clases:

Tabla 3.4: Tabla del rango de cada clase (3) de All, su diferencia y números de datos. $|\Delta|$ es la diferencia del intervalo de cada clase.

Clases	Intervalo	$\|\Delta\|$	# de datos
-1	(-1.8708e+00, -2.9054e-02)	1.8417	495902
0	(-2.9054e-02, 3.7818e-02)	0.0087	495902
1	(3.7818e-02, 1.9883e+01)	19.8457	495901

Cada intervalo de las clases se puede ver como el umbral τ que se explica en la ecuación (3.1). En la tabla (3.4) se aprecian los umbrales.

Los intervalos para el subconjunto de datos denominado Top quedaría así (3.5):

Tabla 3.5: Tabla del rango de cada clase de Top (3), su diferencia y números de datos. $|\Delta|$ es la diferencia del intervalo de cada clase.

Clases	Intervalo	$\|\Delta\|$	# de datos
-1	(-0.9967e+00, -1.9684e-02)	0.9770	80002
0	(-1.9684e-02, 3.9445e-02)	0.0197	79998
1	(3.9445e-02, 1.6762e+00)	1.6367	80000

Y para el subconjunto denominado Bot (3.6):

Tabla 3.6: Tabla del rango de cada clase de Bot (3), su diferencia y números de datos. $|\Delta|$ es la diferencia del intervalo de cada clase.

Clases	Intervalo	$\|\Delta\|$	# de datos
-1	(-1.8708e+00, -4.8813e-02)	1.8220	80000
0	(-4.8813e-02, 3.3056e-02)	0.0157	80000
1	(3.3056e-02, 1.5984e+01)	15.9514	80000

3.5. Estabilidad Temporal de las Clases

El análisis de estabilidad de las clases a lo largo de los distintos periodos permitió evaluar hasta qué punto la categorización del retorno mantiene consistencia en el tiempo. Una vez hecha la categorización de la variable objetivo se calculó la proporción de coincidencias mes a mes para poder analizar la persistencia modal por activo.

En particular, la persistencia modal permitió medir cuántos activos conservaron la misma clase más frecuente entre un periodo y otro.

Para todo el conjunto de datos con 5 clases:

Al calcular la persistencia modal por activo, se observa que los valores se reducen a un rango entre 36 % y 39 % (Tabla 3.7).

Este descenso podría producirse por aumentar el número de clases la distribución de instancias por clase se vuelve más dispersa, lo que incrementa la probabilidad de que los activos cambien de etiqueta entre periodos. En consecuencia, la estabilidad general de las clases disminuye, lo que refleja tanto la naturaleza dinámica de los retornos financieros como la dificultad de capturar patrones cuando el espacio de clases es más amplio.

Tabla 3.7: Persistencia modal por activo entre subperiodos con 5 clases

Comparación de periodos	**Persistencia**
2001-2007 vs. 2008-2013	0.386
2008-2013 vs. 2014-2020	0.379
2001-2007 vs. 2014-2020	0.360

Para todo el conjunto de datos con 3 clases:

Los resultados mostraron que aproximadamente entre el 53 % y el 56 % de los activos mantuvieron la misma clase modal al pasar de un subperiodo a otro, como se resume en la tabla (3.8). Estos valores sugieren un nivel intermedio de estabilidad: aunque una parte considerable de los activos preserva su categoría, también se presentan cambios relevantes que reflejan la naturaleza dinámica y compleja de los mercados financieros. En consecuencia, los modelos de clasificación deben considerar esta variabilidad temporal para mejorar su capacidad de generalización.

Tabla 3.8: Persistencia modal por activo entre subperiodos con 3 clases

Comparación de periodos	**Persistencia**
2001-2007 vs. 2008-2013	0.561
2008-2013 vs. 2014-2020	0.549
2001-2007 vs. 2014-2020	0.538

El análisis de estabilidad de las clases con cinco categorías mostró una menor consistencia temporal en comparación con la configuración de tres clases. Para los casos de *Bot* y *Top*, la persistencia modal se mantiene en los mismos rangos que *All*.

3.6. Escalamiento de los Datos

Para hacer un escalamiento adecuado de los datos se hará la comparación de dos métodos **estandarización, y Mínimo y Máximo** que son clásicos en la literatura [44].

En el caso de **estandarización** se tiene la siguiente ecuación [65] (3.3):

$$X_{scaled} = \frac{x - \mu}{\sigma}, \tag{3.3}$$

donde x es el valor original, μ es la media de la característica y σ es la desviación estándar de la característica. Mientras que para el escalamiento de **mínimo y máximo** (3.4):

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \cdot (f_{max} - f_{min}) + f_{min}, \tag{3.4}$$

donde x_{min} es el valor mínimo de la característica, x_{max} es el valor máximo de la característica, por otra parte, f_{max} y f_{min} son los valores mínimo y máximo deseados para la característica escalada, en este caso $(-1, 1)$. Se escogen esos valores deseados porque nuestras características tienen datos negativos y positivos.

Lo importante es ver qué escalamiento funcionaría mejor y para ello se aplicarán los dos escalamientos, esto se aprecia mejor en un gráfico de calor (Figs. 3.8 y 3.9).

Tomando en cuenta las anteriores figuras, se aprecia que el método de estandarización (3.9) es muy ambiguo, sigue conservando bastantes anomalías las cuales se aprecian en su barra de escala, agregando que se observa la homogeneidad de los datos con este escalamiento. Una posible causa podría ser por la imputación con la mediana (choca con la estandarización ya que considera la media), esto hace que las observaciones reales se sobrerrepresenten y podría

amplificar la distorsión de la distribución de los datos. Mientras que, el método de Mínimo y Máximo (3.8) hace una buena distinción entre toda la distribución de los datos en el intervalo de $(-1,1)$.

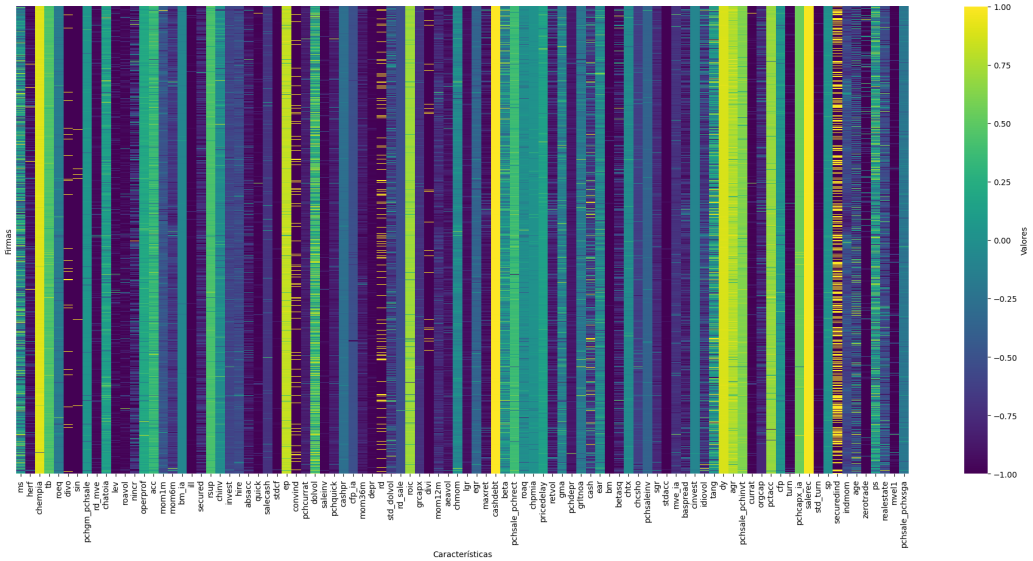


Figura 3.8: Gráfico de Calor de las Características de Toda la Base de Datos Con MixMax Scaler

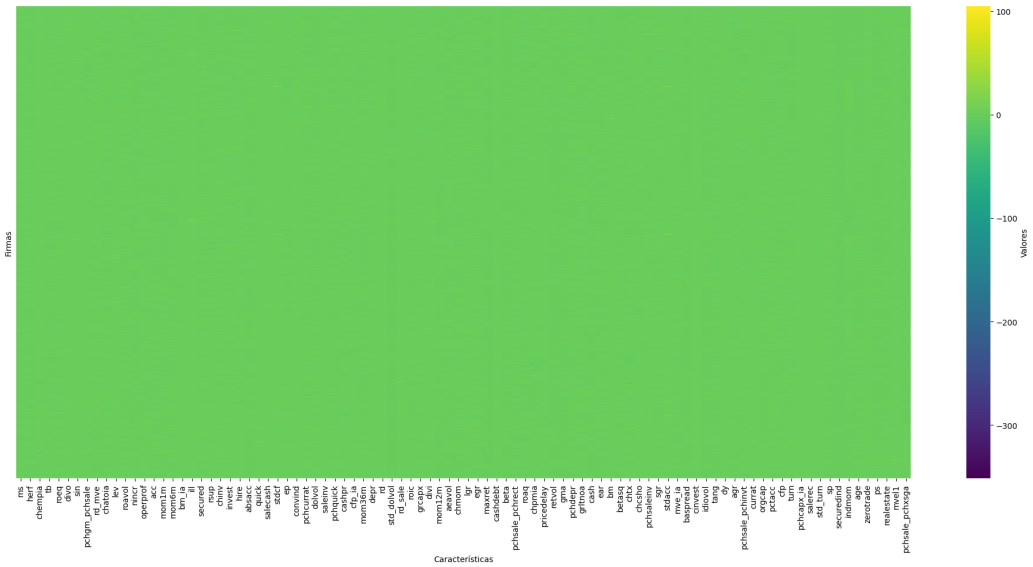


Figura 3.9: Gráfico de Calor de las Características de Toda la Base de Datos Con Standard Scaler

Por lo tanto, el escalamiento de Mínimo y Máximo es el adecuado para la base de datos en un intervalo de $(-1,1)$ *y es el que se usará para aplicar PCA y evaluar la base de datos original sólo con el escalamiento utilizando los 5 modelos antes mencionados.*

3.7. Base de Datos Discretizada

En este apartado se presenta un análisis de las clases generadas por cada discretizador. En particular, se realiza una comparación entre los modelos obtenidos y, para el caso de K-Means, se examina la distribución de clases de acuerdo con la métrica utilizada para evaluar la calidad de los clústeres.

Cabe señalar que se discretizaron en total tres bases de datos: *All*, *Bottom* y *Top*. El número de intervalos no se mantiene fijo en cada discretización, ya que depende directamente de la forma en que operan los algoritmos y de las características específicas de cada conjunto de datos. En el caso de K-Means, se evaluaron seis bases de datos en total: tres de ellas con respecto a la métrica de Calinski–Harabasz y tres con respecto a la métrica de Davies–Bouldin. Por su parte, con el método MDLP únicamente se obtuvieron tres bases de datos.

3.7.1. Por K-means

El uso de K-Means con scikit-learn [65], propicia una discretización de independencia por clases, lo cual es una suposición muy grande, sin embargo, para discretizadores como estos [54], es eficiente en cuestión de costo computacional, realizando esta discretización en poco tiempo. También, se resalta la optimización de este algoritmo [65] que si le das valores altos de k éste no lo respeta si es que tus datos son binarios, entre otros. Lo anterior se puede apreciar en la tabla (3.9), donde se verán el número de etiquetas por cada característica dependiendo de cada métrica.

En esta base de datos se hizo la discretización tomando valores de $2-50$ para el valor de k en K-Means. Se toman estos valores para que haya una búsqueda exhaustiva del mejor k en cada columna. Más adelante se verá el rendimiento de cada discretización según cada métrica.

Métrica de Calinski-Harabasz

El resultado de tener como parámetros la mejor calidad de clusters este métrica de Calinski-Harabasz (C-H) [13] dentro del algoritmo de K-Means, se muestra que entre más clusters haya mejor será el rendimiento en esa métrica. Como se puede ver en la tabla (3.9), la mayoría de número de etiquetas en cada característica es de 50. En esta tabla sólo se ponen 22 características de las 94 sólo para dar un ejemplo. Se aprecia que al tener alguna característica binaria, K-Means lo discretizó en dos clusters. Muy pocas características tuvieron un número de etiquetas distinto a 2 ó 50, como lo fue la característica **turn, ps o rd_sale**.

Para evitar el sobreajuste o la pérdida de información, es importante que las etiquetas en cada característica estén distribuidas de manera relativamente equilibrada, y que no exista una dominancia excesiva de una sola etiqueta, como podría ocurrir en un escenario con, por ejemplo, 50 etiquetas y una sola concentrando la mayoría de los datos. Sin embargo, este comportamiento no puede imponerse de forma rígida a los algoritmos de discretización, ya que se perdería su esencia.

De la tabla (3.10) se desprende que, independientemente del número de K utilizado en el algoritmo, es posible que una etiqueta predomine dentro de una característica, lo cual influye de manera directa en las métricas de evaluación de cada modelo. En particular, en el caso de $C-H$, se observa que la base *Top* presenta una menor tendencia a concentrar los datos en una sola etiqueta, mientras que en *All* y *Bot* ocurre lo contrario. No obstante, en los tres casos persiste un

número considerable de características en las que más del 80 % de la información se concentra en una sola etiqueta.

Tabla 3.9: Tabla de número de valores distintos de algunas características de C-H

Característica	**C-H All**	**C-H Top**	**C-H Bot**
mvel1	50	50	50
beta	50	50	50
betasq	50	50	50
chmom	50	50	50
dolvol	50	50	50
idiovol	50	50	50
indmom	50	50	50
mom1m	50	50	50
turn	50	47	50
agr	49	50	50
cfp	50	50	50
chpmia	50	50	50
convind	2	2	2
currat	50	50	50
depr	50	50	50
divi	2	2	2
divo	2	2	2
ps	9	9	8
quick	50	50	50
rd	2	2	2
rd_mve	50	50	50
rd_sale	49	50	50

Tabla 3.10: Distribución de proporciones de etiquetas por característica en C-H

Base de datos	≤ 0.333	≤ 0.5	≤ 0.8
All	76	32	11
Top	26	14	10
Bot	76	35	10

Métrica de Davies-Bouldin

El resultado de imponer la métrica de Davies-Boulin ($D-B$) como una restricción a escoger los clusters de mejor calidad, da resultados más variados que la métrica $C-H$, ya no se ve el mismo patrón de escoger los mismos números de clusters, lo anterior se refleja en la Tabla (3.11) donde se ven 22 de las características de las 94 que son.

Esta métrica demuestra ser más variada en el número total de clusters que se le da al algoritmo de K-Means. Aunque, hay características con comportamiento muy raro, en la Tabla (3.11) se puede ver en las características **mvel1, pricedelay, pchsaleinv, pctacc, betasq**, entre otras, donde pareciera que en alguna de las bases de datos hay algún dato anómalo que no siga la tendencia de las demás bases de datos.

Tabla 3.11: Tabla de número de valores distintos de algunas características de D-B

Característica	**D-B All**	**D-B Top**	**D-B Bot**
mvel1	3	3	47
beta	21	37	15
betasq	7	43	7
chmom	28	48	48
dolvol	25	42	47
idiovol	11	38	7
indmom	36	36	48
mom1m	26	49	34
turn	41	24	3
agr	2	2	5
cfp	10	26	26
chpmia	2	7	8
convind	2	2	2
currat	2	2	2
depr	3	38	2
divi	2	2	2
divo	2	2	2
ps	10	10	9
quick	2	2	2
rd	2	2	2
rd_mve	41	9	11
rd_sale	2	2	2

Ahora, en la tabla (3.12) se presenta la distribución de proporciones de etiquetas por característica en D-B.

Tabla 3.12: Distribución de proporciones de etiquetas por característica en D-B

Base de datos	≤ 0.333	≤ 0.5	≤ 0.8
All	78	63	55
Top	58	46	45
Bot	81	76	55

En comparación con la Tabla (3.10), la Tabla (3.12) muestra que la discretización mediante K-means, utilizando la métrica de Davies-Bouldin, concentra la mayor parte de la información

en una sola etiqueta para la mayoría de las características. Esto se evidencia especialmente en la cuarta columna, donde 55 características tienen al menos una etiqueta que acumula más del 80 % de la información total.

3.7.2. Por MDLP

El algoritmo MDLP, en su versión original, puede resultar costoso computacionalmente cuando se aplica a bases de datos de gran tamaño. Esto se debe a que, para cada característica continua, primero ordena los valores de menor a mayor y luego evalúa la utilidad de realizar un corte entre cada par de valores distintos. En cada punto candidato se calcula la ganancia de información, lo cual requiere computar tres entropías: (1) la entropía total sin particionar, (2) la entropía del subconjunto izquierdo y (3) la entropía del subconjunto derecho. Posteriormente, se aplica el criterio de longitud mínima de descripción (MDL) para decidir si el corte es aceptado. En caso afirmativo, el algoritmo se ejecuta recursivamente sobre los dos subconjuntos generados, continuando el proceso hasta que no se encuentren más divisiones relevantes.

En el artículo de Chickering et al. [16], se analizan diversas estrategias para mejorar la eficiencia de los algoritmos basados en árboles de decisión. Entre las propuestas destacadas se encuentran:

- Evitar particiones muy pequeñas: Es fundamental asegurar que cada intervalo tenga suficientes datos representativos para prevenir el sobreajuste. En la sección de experimentos del artículo de Chickering et .al [16] se menciona que no se aplican divisiones si una hoja resultante tiene menos de un cierto número de registros.

- Muestreo de puntos de corte: En lugar de evaluar todos los posibles puntos de corte (por ejemplo, todos los puntos medios entre valores consecutivos), resulta conveniente seleccionar un subconjunto de candidatos, lo que preserva la calidad del modelo y acelera el proceso. Esta idea se relaciona con el método k-tile descrito en el mismo artículo de Chickering et al. [16].

En este artículo, se sugieren otras mejoras orientadas a optimizar el algoritmo sin perder información relevante:

- Controlar la complejidad del árbol: Limitar el número de divisiones consecutivas evita que el espacio se divida en demasiados niveles, lo que puede conducir a modelos excesivamente complejos y costosos computacionalmente.

- Muestreo de candidatos: En lugar de determinar cuántos puntos evaluar, es preferible muestrear un subconjunto de candidatos, cubriendo la mayoría del espectro de los datos.

- Cálculo eficiente de estadísticas: Utilizar técnicas como lo *conteos acumulados* permite calcular rápidamente estadísticas sin necesidad de recorrer de manera repetida los datos.

- Paralelización del proceso: Aprovechar todos los recursos computacionales disponibles, minimiza el tiempo de ejecución del algoritmo al distribuir el cómputo entre múltiples tareas.

Estas mejoras se aplicarán en la implementación de MDLP para mejorar su eficiencia, especialmente cuando se requiere discretizar bases de datos de gran volumen, garantizando que la calidad del modelo no se vea comprometida.

Lo anterior se puede resumir en las siguientes estrategias de optimización:

1. **Tamaño mínimo de partición**: Se establece un umbral mínimo m que cada partición debe cumplir. Esta restricción impide generar intervalos con muy pocas instancias, que podrían no ser representativos y conducir a sobreajuste.

2. **Profundidad máxima de recursión**: Se limita la profundidad de las divisiones consecutivas. Si el número de niveles de partición supera un umbral predefinido, el algoritmo detiene la recursión, lo que reduce la complejidad del árbol de cortes.

3. **Submuestreo de puntos candidatos**: En lugar de evaluar todos los valores diferentes de la característica, se selecciona aleatoriamente un subconjunto de puntos candidatos. Esta técnica reduce significativamente el número de evaluaciones, acelerando la ejecución sin comprometer en gran medida la calidad de la discretización.

4. **Conteos acumulados**: Para calcular rápidamente cuántas instancias de cada clase existen a la izquierda y derecha de cada corte, se utiliza una estructura de prefijos acumulados, lo que permite obtener esta información en tiempo constante ($O(1)$) por punto.

5. **Paralelización del proceso**: El algoritmo se implementa de forma paralela, dividiendo el trabajo entre múltiples núcleos del procesador, lo que mejora considerablemente el tiempo de ejecución.

Estas mejoras permiten aplicar MDLP a conjuntos de datos mucho más grandes de lo que sería posible con su versión tradicional, manteniendo una alta calidad en la discretización y mejorando el rendimiento computacional general.

Se muestra un pseudocódigo (Alg. 2) que resume el funcionamiento del algoritmo optimizado.

Los parámetros a utilizar en esta discretización fueron:

1. $min\_size = 50$,

2. $max\_depth = 20$,

3. $candidate\_sample\_rate = 0.8$,

4. $n\_jobs = -1$.

Algorithm 2 MDLP con Mejoras de Eficiencia

Require: *xSorted*: vector de valores ordenados de menor a mayor;
ySorted: vector de clases correspondiente a *xSorted*;
prefixCount: matriz con conteos acumulados de clases;
minSize: tamaño mínimo de partición;
maxDepth: profundidad máxima;
candidateSampleRate: fracción de puntos a evaluar;
$depth \leftarrow 1$.
Ensure: Conjunto de puntos de corte $\mathcal{C}$.
function FINDCUTPOINTS($xSorted, ySorted, prefixCount, depth$)
    $n \leftarrow \text{length}(xSorted)$
    if $(n < 2)$ **or** $(depth > maxDepth)$ **then**
        return $\emptyset$ ▷ No se puede seguir cortando
    end if
    $diffs \leftarrow \{\, i \mid 1 \le i < n,\ xSorted[i] \neq xSorted[i-1] \}$
    if *diffs* está vacío **then**
        return $\emptyset$ ▷ No hay lugares donde cortar
    end if
    $candidateIndices \leftarrow \text{Submuestra}(diffs, candidateSampleRate)$
    $(bestCut, bestEntropy) \leftarrow \text{FindBestCut}(candidateIndices, prefixCount, minSize)$
    if $(bestCut = \text{None})$ **then**
        return $\emptyset$ ▷ Ningún corte válido
    end if
    $gain \leftarrow \text{MDLStop}(prefixCount, bestCut, bestEntropy)$
    if $(gain = \text{None})$ **then**
        return $\emptyset$ ▷ No supera el umbral MDL
    end if
    ▷ **Si se aprueba el corte, dividir y recursear**
    $xLeft \leftarrow xSorted[\,0 : bestCut\,]$
    $yLeft \leftarrow ySorted[\,0 : bestCut\,]$
    $xRight \leftarrow xSorted[\,bestCut : n\,]$
    $yRight \leftarrow ySorted[\,bestCut : n\,]$
    $prefixLeft \leftarrow \text{BuildPrefixCounts}(yLeft)$
    $prefixRight \leftarrow \text{BuildPrefixCounts}(yRight)$
    $\mathcal{C}_{\text{left}} \leftarrow \text{FindCutPoints}(xLeft, yLeft, prefixLeft, depth+1)$
    $\mathcal{C}_{\text{right}} \leftarrow \text{FindCutPoints}(xRight, yRight, prefixRight, depth+1)$
    ▷ Punto de corte en valor medio de x adyacentes
    $cutValue \leftarrow \frac{xSorted[bestCut-1]+xSorted[bestCut]}{2}$
    return $\mathcal{C}_{\text{left}} \cup \{cutValue\} \cup \mathcal{C}_{\text{right}}$
end function

En pocas palabras se tomó que el **Tamaño mínimo de partición** exige que cada partición resultante tenga 50 muestras totales, por otro lado se toma una **profundidad máxima de recur-**

sión de 20, para evitar hacer más subdivisiones innecesarias, mientras en **submuestreo de posibles puntos de corte** se conserva un 80 % del número total de valores únicos en toda la base de datos y, por último, se utilizaron todos los núcleos del procesador del equipo con $n\_jobs = -1$ para acelerar el proceso lo más rápido posible. Teniendo un tiempo de ejecución para *All* de **496.10 seg**, es decir, aproximadamente **8 min** en discretizar una base de datos de más de 1.4 millones de instancias. Para bot y top, fueron aproximadamente **1 min** para cada uno. Siendo notable los cambios que se le hizo al algoritmo de MDLP en el tiempo de ejecución, reduciéndolo drásticamente sin arriesgar tanta pérdida de información comparado a un MDLP sin las mejoras.

Después de aplicarle la discretización a la base de datos, se hizo la indagación de valores únicos en cada característica, es decir, cuántas etiquetas generó en cada columna. MDLP no tiene la restricción de hacer un número de etiquetas como máximo, es decir, tiene la libertad de considerar así como una etiqueta, o, por ejemplo, 1000 etiquetas en una sola columna. lo anterior debido a la manera en cómo hace las particiones, si una partición supera el umbral establecido por el criterio de MDL, entonces se considera como etiqueta, el algoritmo parará hasta que ya no haya cortes óptimos después de cierta profundidad.

Para el caso de 3 clases en la variable objetivo:

Tabla 3.13: Tabla de número de valores distintos de algunas características de MDLP

Característica	**MDLP All**	**MDLP Top**	**MDLP Bot**
mvel1	17	9	10
beta	36	13	16
betasq	31	13	13
chmom	29	9	13
dolvol	12	4	9
idiovol	29	14	16
indmom	1804	94	255
mom1m	48	12	23
turn	14	9	9
agr	24	7	12
cfp	21	11	11
chpmia	82	27	14
convind	2	6	2
currat	21	6	8
depr	14	8	9
divi	2	1	2
divo	2	2	2
ps	5	2	5
quick	15	6	9
rd	2	2	2
rd_mve	11	6	7
rd_sale	9	3	5

El resultado del total de etiquetas en 22 características está representado en la Tabla (3.13) donde se aprecia variabilidad en la mayoría de las características no importando si se trata de la base de datos *All, Top o Bot*. Sin embargo, presenta casos anómalos como la característica **indmom** dando en total 1804 etiquetas generadas en *All*, estas son bastantes comparadas a las otras características, esto pasa porque no halla puntos de corte óptimos en toda la característica. Otra anomalía es que la característica **divi** en *Top* sólo tiene una etiqueta, el algoritmo no encontró puntos de corte óptimos en toda la característica, por lo tanto, se generó una característica constante.

En la tabla (3.14) se presenta cuántas características tienen al menos una etiqueta con más proporción de datos comparado con las otras etiquetas.

Tabla 3.14: Distribución de proporciones de etiquetas por cada característica en MDLP

Base de datos	≤ 0.333	≤ 0.5	≤ 0.8
All	45	17	4
Top	74	40	10
Bot	72	23	5

En la Tab. (3.14) se muestra muy pocas columnas son aproximadamente constantes. Otro dato a resaltar es que en *Top* se encontraron 40 etiquetas que conservan más del 50 % de los datos, siendo un número grande comparado con los otros casos como *All y Bot*.

Para el caso de 5 clases en la variable objetivo:

El resultado del total de etiquetas en 22 características está representado en la Tabla (3.15) donde se aprecia variabilidad en la mayoría de las características no importando si se trata de la base de datos *All, Top o Bot*. Sin embargo, presenta casos anómalos como la característica **indmom** dando en total 1626 etiquetas generadas en *All*, son muchísimas comparadas a las otras características. Otra anomalía es que la característica **divi** en *Top* sólo tiene una etiqueta, el algoritmo no encontró puntos de corte óptimos en toda la característica, por lo tanto, se generó una característica constante.

En la tabla (3.16) se presenta cuántas características tienen al menos una etiqueta con más proporción de datos comparado con las otras etiquetas. Esta muestra un mejor equilibrio en el porcentaje de datos contenidos en las etiquetas de cada columna, teniendo bastante variabilidad.

De la previas tablas de esta sección 3.6, dos características tuvieron comportamientos similares, posiblemente por su definición. De la tabla (A.1) se sabe que, **indmom** son los retornos promedios ponderados de la industria durante 12 meses y **divi** es un variable indicadora que marca si la empresa pagó sus dividendos, siempre es 0 o 1, es decir, es una característica binaria.

En comparación a las otras dos discretizaciones por parte de K-Means, MDLP representa un mejor equilibrio en las etiquetas producidas en cada una de sus características, evitando que se considere como una sola etiqueta para una característica.

Se aplicarán estos tres casos de discretización, K-Means dos veces con distinta métrica y MDLP+, para evaluar la calidad de los modelos de aprendizaje supervisado.

Tabla 3.15: Tabla de número de valores distintos de algunas características de MDLP

Característica	**MDLP All**	**MDLP Top**	**MDLP Bot**
mvel1	20	8	8
beta	39	14	15
betasq	29	12	14
chmom	30	10	13
dolvol	17	8	10
idiovol	35	14	18
indmom	1626	51	273
mom1m	43	14	24
turn	18	10	11
agr	24	7	14
cfp	22	11	12
chpmia	95	24	15
convind	2	2	2
currat	19	7	6
depr	17	9	8
divi	2	1	2
divo	2	2	2
ps	6	9	4
quick	20	6	9
rd	2	2	2
rd_mve	13	6	7
rd_sale	11	4	5

Tabla 3.16: Distribución de proporciones de etiquetas por cada característica en MDLP

Base de datos	≤ 0.333	≤ 0.5	≤ 0.8
All	59	20	5
Top	64	30	7
Bot	78	22	5

3.7.3. Reducción de Dimensionalidad Con PCA

Para aplicar PCA a las bases de datos, se necesita hacer un escalamiento, el escalamiento que representa de mejor manera la naturaleza de los datos es el Min Max Scaler (sección 3.5, págs. 69 y 70), del cual ya se habló en previas secciones.

Al usar PCA en el conjunto de datos, elimina la mayoría de correlación que hay entre cada una de las nuevas características [62]. Se aprecia en la siguiente figura la correlación de las características en el conjunto *All*:

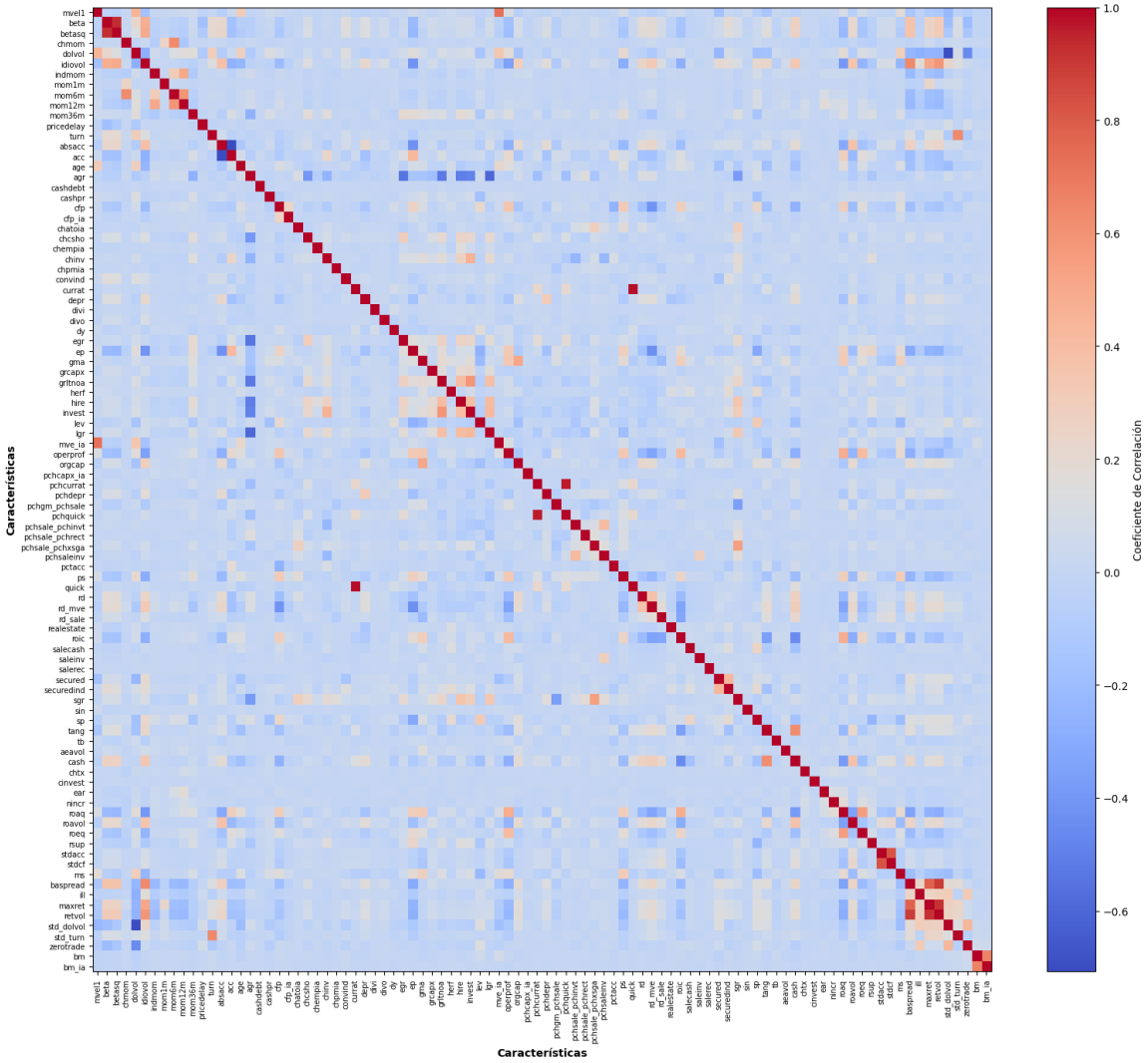


Figura 3.10: Gráfico de Calor de la Matriz de Correlación de All

A pesar de que la figura anterior es para *All*, el mismo gráfico se puede observar para *Top* y *Bot*, no hay grandes diferencias.

Se observa en (3.10) que muy pocas características están altamente correlacionadas, como es el caso de **bm y bm_ia**, que provienen del mismo significado, o también como las características

maxret y retvol. Por el otro lado, hay muy pocas características que están anti-correlacionadas como lo pueden ser **acc** y **absacc**.

Para eliminar esta correlación, un método es aplicar PCA, sin embargo, la pregunta fundamental es determinar cuántas componentes pueden explicar una buena varianza y reducir la dimensión para disminuir el tiempo de ejecución aplicado a cada modelo de aprendizaje supervisado sin arriesgar la pérdida del *accuracy*.
Se buscó cuántas componentes describen cierto porcentaje de la varianza explicada acumulada, en el siguiente gráfico:

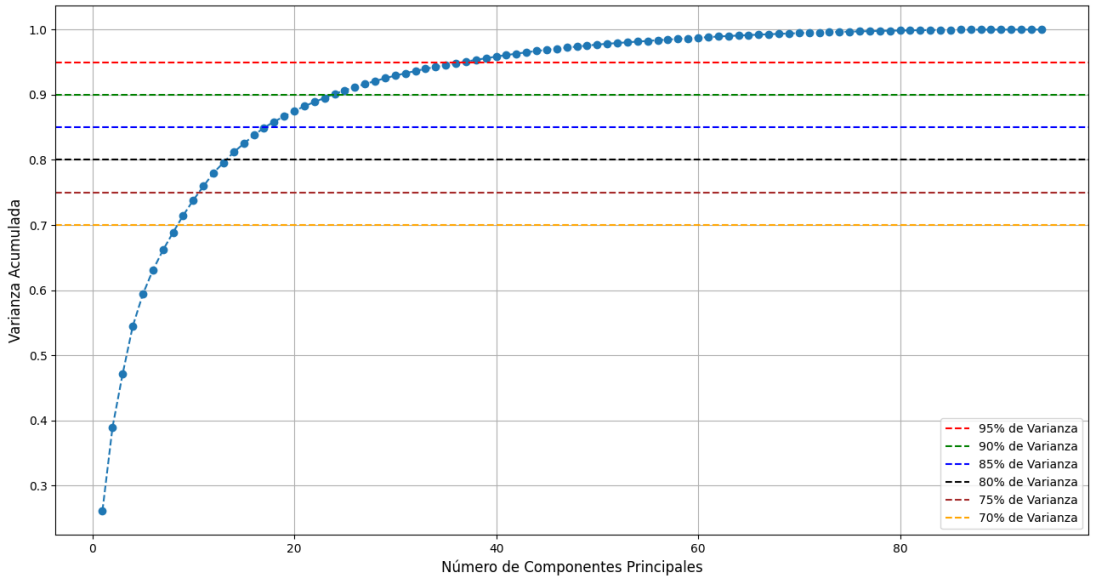


Figura 3.11: Gráfico de Codo Contra la Varianza Explicada Acumulada de All

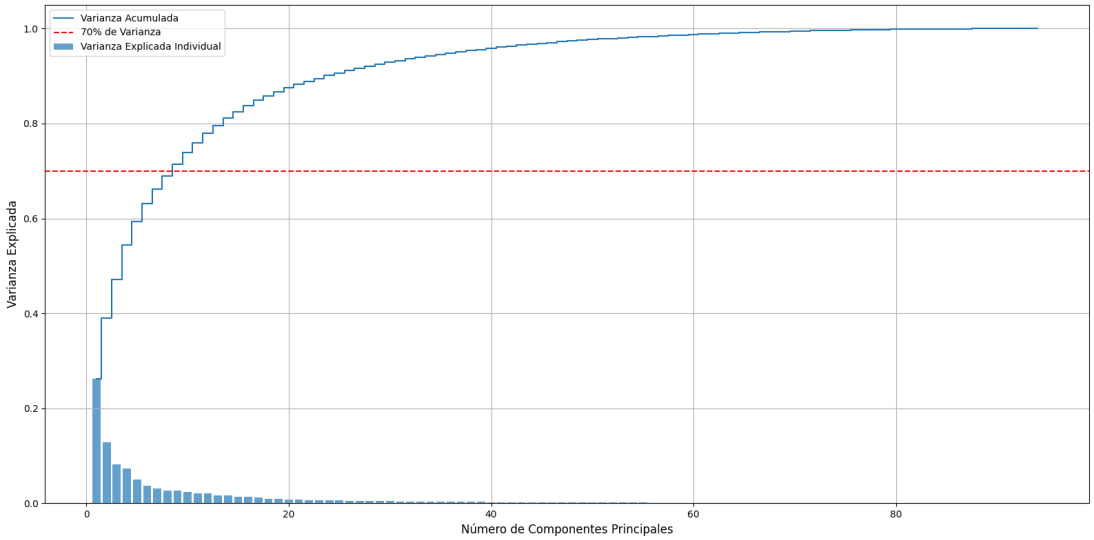


Figura 3.12: Gráfico de la Varianza Explicada Acumulada por Componente de All

Tabla 3.17: Componentes principales necesarios para distintos niveles de varianza explicada

Varianza Explicada Acumulada	**70 %**	**75 %**	**80 %**	**85 %**	**90 %**	**95 %**
Número de componentes principales	9	11	14	18	24	37

En los anteriores gráficos (3.11), (3.12), y la tabla (3.17), se aprecia que para explicar más varianza del 95 % se necesitan 37 componentes principales, es decir, que se redujo a un 35 % la base de datos, reduce el tiempo de ejecución de los modelos y no se consumen tantos recursos computacionales, sin embargo, se puede mejorar.

Para ver las mejoras se hizo lo siguiente:

1. Se calculó el número de componentes principales que explican el $95, 90, 85, 80, 75$ y $70\,\%$ de la varianza de la base de datos.

2. Se guardó en una nueva base de datos esas nuevas componentes principales para poder aplicarle un método de aprendizaje supervisado que no tuviera problemas con que la base de datos fuera numérica y se hizo clasificación para medir el *accuracy*, en total se aplicó lo anterior a 6 bases de datos.

3. Se seleccionó el método que no necesitara tantos parámetros por modificar, es decir, la regresión logística.

4. Los parámetros establecidos para este modelo se pueden observar en el capítulo de Metodología 3.9, la sección **Parámetros y Detalles de Cada Modelo** (pp. 91).

5. Cada accuracy fue calculado y validado con validación cruzada con k-folds igual a 10.

6. Se escogió el número de componentes que tuviera la mejor reducción de dimensionalidad sin afectar drásticamente el *accuracy* e incertidumbre con desviación estándar (std) del modelo. Entre menor dimensionalidad, menor tiempo de ejecución del modelo.

Para el caso de 5 clases en la variable objetivo:
Para la base de datos *All*, se encontraron los siguientes valores:

Tabla 3.18: Tabla de las características de cada conjunto de datos creado con PCA en All

% Varianza	# Componentes	Accuracy $\pm$ Std	Tiempo (seg)
95	37	0.3084 ± 0.0015	2596
90	24	0.3062 ± 0.0011	1978
85	18	0.3043 ± 0.0009	1531
80	14	0.2984 ± 0.0010	561
75	11	0.2979 ± 0.0009	480
70	9	0.2963 ± 0.0012	413

Dada la anterior tabla, se tomará el **70 %** de varianza para generar un número de 9 componentes principales. Esto se eligió con base el gráfico (3.12) y la tabla (3.18), en el gráfico se

observa que a partir de 70 % deja de tener diferencias significativas la precisión del modelo en relación al número de componentes. Con respecto a la precisión, no se pierde mucho entre los distintos valores del porcentaje que explica la varianza de la base de datos, además, se consiguió un buen tiempo de ejecución a pesar de realizar validación cruzada con k-folds igual a 10. La base de datos se reduce a un 10 % de su tamaño con respecto a la base de datos original.

Otro análisis importante sobre estas nuevas componentes es revisar su correlación con la variable objetivo, esto para averiguar que no se haya metido nuevo ruido a la base de datos. Esto se aprecia en la figura (3.13). Se aprecia que no hay correlación significativa con la variable objetivo.

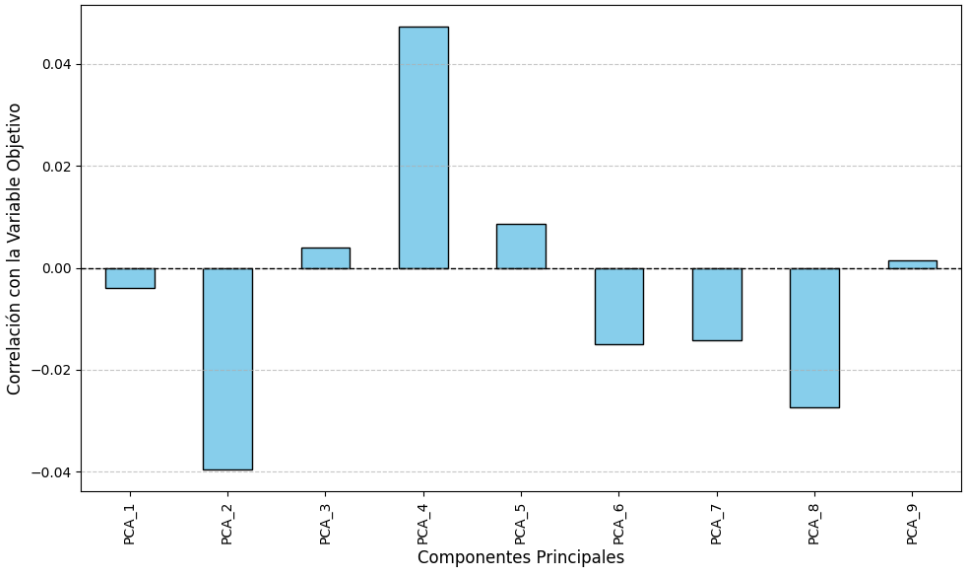


Figura 3.13: Gráfico de Correlación de las Componentes Principales con la Variable Objetivo de All con 9 Componentes Principales (70 % de varianza)

Se hizo un análisis similar para los dos subconjuntos (*Top y Bot*). Para *Bot* se tomará el **75 %** ya que en el *accuracy* se mantiene en el mismo decimal y centésima y el tiempo de ejecución es muy ligero.

Tabla 3.19: Tabla de las características de cada conjunto de datos creado con PCA en Bot

% Varianza	# Componentes	Accuracy $\pm$ Std	Tiempo (seg)
95	42	0.3194 ± 0.0016	124
90	29	0.3173 ± 0.0018	50
85	21	0.3160 ± 0.0018	42
80	17	0.3142 ± 0.0015	36
75	14	0.3124 ± 0.0019	32
70	11	0.3088 ± 0.0022	29

Tabla 3.20: Tabla de las características de cada conjunto de datos creado con PCA en Top

% Varianza	# Componentes	Accuracy ± Std	Tiempo (seg)
95	32	0.2717 ± 0.0037	303
90	20	0.2660 ± 0.0035	218
85	15	0.2586 ± 0.0025	173
80	11	0.2453 ± 0.0014	139
75	9	0.2404 ± 0.0015	120
70	7	0.2386 ± 0.0018	108

Para el subconjunto de *Top* no se tuvo una gran mejoraría en los resultados del *accuracy* al aplicar PCA ni tampoco el tiempo de ejecución, esto comparado con Bot. Por ello se escogerá un punto medio entre *accuracy* y tiempo de ejecución, es decir, se escogerá el conjunto con el **85 %** de varianza.

Se aprecia que en general, el subconjunto Top será el más complicado de clasificar, esto porque su *accuracy* es más bajo comparado a *All y Bot*.

Para el caso de 3 clases en la variable objetivo:

Para la base de datos *All*, se encontraron los siguientes valores:

Tabla 3.21: Tabla de las características de cada conjunto de datos creado con PCA en All con 3 clases

% Varianza	# Componentes	Accuracy ± Std	Tiempo (seg)
95	37	0.4518 ± 0.0011	3267
90	24	0.4497 ± 0.0011	1847
85	18	0.4478 ± 0.0012	1452
80	14	0.4421 ± 0.0012	812
75	11	0.4412 ± 0.0011	590
70	9	0.4394 ± 0.0011	688

Dada la tabla (3.21), se tomará el **80 %** de varianza para generar un número de 14 componentes principales. Esto se eligió con base al *accuracy*, no se pierde mucho entre los distintos valores del porcentaje que explica la varianza de la base de datos, además, se consiguió un buen tiempo de ejecución a pesar de realizar validación cruzada con k-folds. De esta manera la base de datos se reduce a un 10 % de su tamaño con respecto a la base de datos original.

Se hizo un análisis similar para los dos subconjuntos *(Top y Bot)*. Para *Bot*, en la tabla (3.22) se tomará el **70 %** ya que no hay gran mejoría en el *accuracy* y el tiempo de ejecución es muy ligero con respecto al porcentaje de las otras componentes (75 %, 80 % y 85 %).

Para el subconjunto de *Top*, en la tabla (3.23) sí hay un cambio muy grande en el tiempo de ejecución de cada conjunto de datos creado con PCA, se observa en el porcentaje de 80 % al 85 % de varianza, aumentó más del 500 % el tiempo de ejecución, de *accuracy* sólo aumentó una centésima. Por lo tanto, del conjunto *Top* se tomará el conjunto del **80 %** de varianza.

Tabla 3.22: Tabla de las características de cada conjunto de datos creado con PCA en Bot con 3 clases

% Varianza	# Componentes	Accuracy $\pm$ Std	Tiempo (seg)
95	42	0.4690 ± 0.0034	84
90	29	0.4670 ± 0.0026	63
85	21	0.4664 ± 0.0035	47
80	17	0.4648 ± 0.0029	43
75	14	0.4646 ± 0.0024	38
70	11	0.4616 ± 0.0027	33

Tabla 3.23: Tabla de las características de cada conjunto de datos creado con PCA en Top con 3 clases

% Varianza	# Componentes	Accuracy	Tiempo (seg)
95	32	0.4126 ± 0.0048	272
90	20	0.4062 ± 0.0052	182
85	15	0.3993 ± 0.0042	155
80	11	0.3812 ± 0.0029	27
75	9	0.3774 ± 0.0025	26
70	7	0.3766 ± 0.0024	24

En la tabla (3.24) se resume qué características o componentes se usará por cada conjunto de datos:

Tabla 3.24: Comparación de métodos de selección de características

Método / Referencia	**Características seleccionadas**
PCA	Utilizado como referencia para comparar tiempo de ejecución frente a modelos discretizados.
Kelly et al. (2018) [47]	**beta, betasq, bm, bm_ia, lev, turn, idiovol, invest, mom1m, mom6m, mom12m, mom36m, mvei1, mve_ia, chmom**
Gu et al. (2021) [35]	**mom1m, mom12m, chmom, indmom, maxret, mom36m, turn, mvel1, dolvol, ill, zerotrade, baspread, retvol, idiovol, beta, betasq, ep, sp, agr, nincr**
Redes Bayesianas	Comparación adicional para evaluar qué método de selección resulta más adecuado en esta base de datos.

3.8. Selección de Características de Redes Bayesianas: Método basado en MDL-FS

En la sección de evaluaciones se verá cuál fue la mejor elección de características con respecto a PCA, redes bayesianas, la elección de Gu et al. [35] y la elección de Kelly et al. [47].

Respecto al método de MDL-FS [22] para la selección de características se sabe que cuando la evaluación de una característica mediante la función $MDL\text{-}FS(C,S|D)$ arroja resultados negativos, implica que $CALL(C,S|D)$ es negativo, lo que significa que añadir esta característica mejora considerablemente la predicción de la clase C. Esto se debe a una disminución de la entropía condicional $H_{\hat{p}}(C|A)$, lo cual incrementa la log-verosimilitud del clasificador $LL(C|D)$. Por otro lado, puntajes cercanos a cero indican que la característica aporta escasa información, mientras que, valores positivos indican que la característica empeora el modelo, es decir, aumenta la complejidad sin mejorar la capacidad predictiva.

Para facilitar la elección de las características con mejor puntaje, se seleccionaron las 10 más relevantes de cada conjunto y subconjunto de datos, esto debido a la falta de entendimiento en el artículo de Drugan et al. [22], ya que no explica qué umbral se debe seguir para considerar una característica como relevantes. Los resultados se obtuvieron a partir del promedio de los puntajes generados mediante validación cruzada, utilizando k-fold igual a 10 para Top y Bot, y k-fold igual a 5 para All.

Para el caso de 5 clases en la variable objetivo

Como se muestra en la Tabla (3.26), hay características que son ampliamente más informativas que otras, como son las características: **baspread, retvol, maxret y idiovol**, tienen un puntaje muy alto comparado a las demás características, sin embargo, de los tres conjuntos, *All, Top y Bot*, se obtienen en común la mayoría de características.

Tabla 3.26: Tabla de las características seleccionadas $k = 5$

All	Top	Bot
baspread $= -131626$	baspread $= -12704$	baspread $= -26012$
retvol $= -124519$	retvol $= -11011$	retvol $= -25814$
idiovol $= -98487$	idiovol $= -8003$	maxret $= -21036$
maxret $= -98487$	maxret $= -7319$	idiovol $= -19859$
indmom $= -90169$	mom12m $= -4880$	mom12m $= -14964$
mom12m $= -61116$	beta $= -4173$	mom6m $= -13495$
mom6m $= -55196$	betasq $= -4167$	mom1m $= -13150$
herf $= -54827$	turn $= -4060$	ep $= -11340$
mom1m $= -52798$	mom6m $= -3818$	cashdebt $= -11232$
ep $= -46978$	zerotrade $= -3637$	herf $= -10610$

Para el caso de 3 clases en la variable objetivo

Como se observa en la Tabla (3.26) y ahora con la Tabla (3.27), hay características que son ampliamente más informativas que otras, como son las características: **baspread, retvol, maxret y idiovol**. Repitiéndose el mismo patrón no importando cuentas clases se tengan en la

variable objetivo. También, en el caso de *All* se tiene la relevancia de **indmom** la cual fue una de las características más informativas en el conjunto de datos.

Tabla 3.27: Tabla de las características seleccionadas $k = 3$

All	Top	Bot
baspread $= -82918$	baspread $= -7230$	baspread $= -13678$
indmom $= -79977$	retvol $= -6211$	retvol $= -18033$
retvol $= -79442$	idiovol $= -4350$	maxret $= -14973$
maxret $= -64439$	maxret $= -4104$	idiovol $= -13675$
idiovol $= -61127$	mom12m $= -2546$	mom12m $= -10034$
herf $= -38652$	indmom $= -2509$	mom1m $= -9407$
mom12m $= -36037$	beta $= -2281$	mom6m $= -9286$
betasq $= -33332$	betasq $= -2265$	indmom $= -8780$
mom6m $= -33315$	herf $= -2067$	cashdebt $= -8137$
beta $= -33208$	mom6m $= -2023$	herf $= -8030$

3.9. Parámetros y Detalles de Cada Modelo

En este apartado se explicará qué parámetros tendrá cada modelo a utilizar. De forma general, se aplicará validación cruzada con $k - fold$ a cada modelo.

Aunque k puede establecerse en cualquier valor, la validación cruzada con 10 folds es probablemente la variante más común utilizada en la práctica [44]. Por lo tanto, para las siguientes evaluación se usarán 10 folds para cada validación cruzada de su respectivo modelo.

La manera en cómo hacer la división y seleccionar cada k-fold será de manera aleatoria, en este caso se utilizará la función *Random State* de python [77] con el número 42. Además, para cada modelo se usó Numpy [37] para el manejo de bases de datos, Numpy [37] para las operaciones y scikit-learn [65] para aplicar las métricas de evaluación en clasificación. La validación cruzada con k-fold se construyó con ayuda de Numpy [37].

Naive Bayes

En el caso del modelo de Naive Bayes, fue construido con lo explicado en el marco teórico. Sólo se usaron las paqueterías de python: Pandas [58] para el manejo de las bases de datos y Numpy [37] para las operaciones.

CART

Para el modelo de CART se utilizó la implementación `DecisionTreeClassifier` de la biblioteca scikit-learn [65], la cual está basada en el algoritmo propuesto por Breiman et al. [11]. Esta función permite inducir árboles de decisión de forma optimizada, facilitando la reducción del tiempo de ejecución y un mejor manejo de funciones básicas. Cabe destacar que, si bien la implementación en scikit-learn no soporta directamente variables categóricas nominales, lo que

en otros contextos obligaría a aplicar codificación previa, en este trabajo ello no representa una limitación, ya que todas las variables predictoras fueron previamente discretizadas mediante MDLP+, convirtiéndose en categorías ordinales. Este formato es plenamente compatible con las divisiones binarias que CART utiliza, por lo que el rendimiento del clasificador no se ve afectado.

En cuanto al criterio de impureza, se seleccionó `entropy`, debido a que en problemas de clasificación multiclase y con posibles desbalances entre categorías al aplicar validación cruzada, la entropía penaliza más fuertemente las divisiones sesgadas, favoreciendo una separación más equitativa de las clases discretizadas de retorno financiero.

Por último, se fijó el parámetro `random_state` en el valor **42**. Esto no altera la lógica del algoritmo, pero garantiza la *reproducibilidad* de los resultados: con un valor constante, los experimentos generan siempre la misma estructura de árbol y las mismas métricas de desempeño, lo que permite comparar resultados entre modelos y corridas distintas. Si se hubiera dejado en `None`, cada ejecución habría generado ligeras variaciones debido a la aleatoriedad en la selección de características o instancias, dificultando la comparación sistemática en la investigación.

Regresión Logística

Para la regresión logística, se tienen varios parámetros para cambiar. Primero, se usó la función de *LogisticRegression* de scikit-learn [65] sirve para entrenar un modelo de regresión logística con regularización Ridge, Lasso o Elastic-Net (combinación de las dos anteriores), la regularización se escogió como Elastic-Net, esto para evitar el sobreajuste y limitar la complejidad del mismo [8]. Otro parámetro a cambiar es C que es el inverso de la fuerza de regularización, este se quedó en su valor predeterminado por scikit-learn $C = 1.0$, también está el parámetro de *Solver* que es un algoritmo para optimizar el problema dado, en este caso se usó **SAGA** ya que es un buen optimizador para problemas de gran escala, posee una implementación eficiente para problemas de clasificación [20]. Por último, se indicó un máximo de 1000 iteraciones en el modelo, para encontrar una buena solución sin necesidad de gastar mucho tiempo de ejecución.

Cuando se usa el **SAGA**, se tiene que especificar el Random State del modelo, por lo cual se tomó como en todos los modelos, 42.

K Vecinos Más Cercanos

Uno de los únicos modelos que tiene muy pocos parámetros a cambiar, es K vecinos más cercanos. Sin embargo este modelo tiene un problema distinto a los demás, escoger la mejor K según la base de datos.

Para lo anterior, se hizo uso de la función de *KNeighborsClassifier* de scikit-learn [65], en donde se hizo más hincapié a encontrar el mejor K para cada modelo. Para hallarlo, se iteró el modelo de 2 hasta 20 números de vecinos a considerar para seleccionar la clase de la muestra. Se evaluó con la métrica de Accuracy, la K con mejor accuracy se usaba para el modelo y luego aplicarle validación cruzada con k-folds.

Los demás parámetros del modelo se dejaron tal cual viene por default, es decir, para el parámetro de *weights* se dejó en "uniform" que indica que todos los vecinos tienen el mismo peso, el parámetro de *algorithm* se indicó como "auto", es decir, que el modelo escogerá el

mejor algoritmo para la base de datos dada, para este problema se espera que use “brute” (fuerza bruta) ya que es el más apropiado para datos de altas dimensiones, la métrica de distancia a usar fue “euclidean”, es decir, que el modelo usará la métrica euclidiana para calcular la cercanía de los puntos y la distancia entre estos [65].

Redes Neuronales

Tabla 3.28: Parámetros y Detalles de Redes Neuronales.

Parámetro	**Descripción**
Capa de entrada	Una capa densa (`Dense`) con 64 neuronas y función de activación `relu`.
Capas ocultas	El modelo permite agregar un número variable de capas ocultas mediante el parámetro `hidden_layers`. Cada capa contiene 64 neuronas y usa la función de activación `relu`. En este caso, sólo hay una capa oculta.
Capa de salida	Una capa densa con 5 neuronas (para cinco clases) y la función de activación `softmax`, adecuada para clasificación multiclase [57].
Función de pérdida	Se utiliza `categorical_crossentropy`, adecuada para tareas de clasificación multiclase con salidas codificadas en formato one-hot [61].
Optimizador	Se emplea `adam`, un optimizador adaptativo ampliamente usado [48] que combina `RMSprop` (Root Mean Square Propagation) ajusta la tasa de aprendizaje para cada parámetro del modelo de manera individual.Lo anterior usando un promedio móvil de los cuadrados de los gradientes recientes [1], y `momentum` acelera el proceso de convergencia acumulando un promedio ponderado de los gradientes pasados. Esto ayuda al optimizador a “recordar” la dirección hacia la que se ha estado moviendo, evitando oscilaciones y avanzando más rápidamente en esa dirección [1].
Métrica de evaluación	Se mide la `accuracy` para evaluar el desempeño del modelo durante el entrenamiento y validación.
Épocas (`epochs`)	Número de iteraciones completas sobre el conjunto de datos. En este caso, el valor bajo 10.
Tamaño del lote (`batch_size`)	Número de muestras procesadas antes de actualizar los parámetros del modelo [1]. Se establece en 32.

El modelo se construyó con Tensor Flow [1], creando una red neuronal con *Sequential*,

con una capa de entrada, una capa oculta y una de salida usando la semilla de $seed = 42$, $os.environ['PYTHONHASHSEED'] = seed, tf.random.set\_seed(42)$ y $TF\_DETERMINISTIC\_OPS = 1$ para replicabilidad de resultados. Para agregar cada capa se hizo uso de *Dense*. Los hiperparámetros se mostrarán en la Tabla (3.28).

3.9.1. Metodología Para Las Evaluaciones

Para los modelos que tienen parámetros que pueden variar mucho como lo es Redes Neuronales y K Vecinos Más Cercanos, se hizo un procedimiento distinto a los demás modelos. Los demás modelos sólo se ejecutaron como se explicó en la subsección 3.8 de **Parámetros y Detalles de Cada Modelo** (págs. 88-90).

Todas las evaluaciones se harán por separado de cada modelo, esto para hallar el mejor modelo que mejor explique la problemática de los datos, por lo tanto, cada modelo tendrá 8 subconjuntos a probar, 2 para k-means, 1 para MDLP, 1 para PCA, 3 para selección de características y por último, se evaluará la base de datos con el escalamiento de Min Max Scaler de (-1,1) (sin discretizaciones ni elecciones de características) para tratar el problema desde múltiples puntos.

K Vecinos Más Cercanos

Para determinar el valor óptimo de K en el algoritmo KNN para cada subconjunto de datos (*Top* y *Bot*), se empleó una división de los datos en 70 % para entrenamiento y 30 % para prueba. Para cada $K \in [2, 20]$, se calculó el *accuracy* del modelo en la etapa de prueba, lo cual permitió comparar el desempeño entre diferentes valores de K usando la tasa de error (*error_rate*) es el porcentaje de errores de clasificación del KNN en el conjunto de prueba, usado para comparar qué valor de K minimiza esos errores. Este desempeño se muestra en la Figura (3.14).

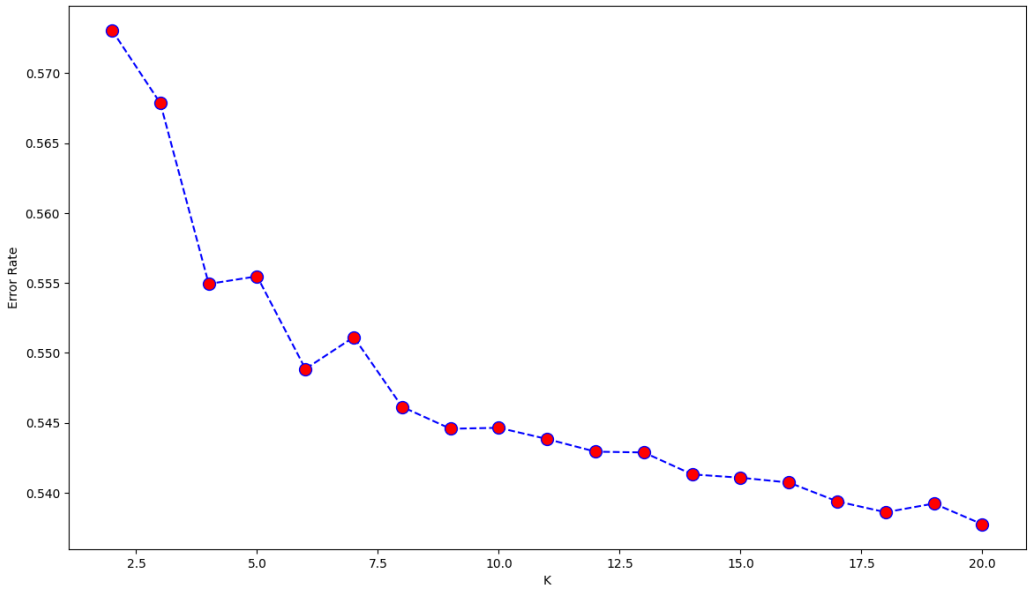


Figura 3.14: Evaluación de la tasa de error para diferentes valores de K en el modelo KNN.

Una vez identificado el valor óptimo de K para cada subconjunto mediante esta estrategia exploratoria, dicho valor se utilizó posteriormente en una validación cruzada con *k-folds*. Este enfoque evitó la necesidad de realizar una búsqueda exhaustiva de K dentro del esquema de validación cruzada, lo cual habría sido computacionalmente costoso.

En particular, la Figura 3.14 muestra que a partir de $K = 13$ la tasa de error comienza a estabilizarse, alcanzando su valor mínimo en $K = 20$. Sin embargo, las diferencias entre los últimos valores de K son marginales. Por ello, se seleccionó $K = 20$, ya que representa un equilibrio adecuado entre bajo error de clasificación y estabilidad del modelo.

Esta selección se realizó de forma individual para cada variable, eligiendo el valor de K con menor tasa de error o aquel donde la curva se estabilizaba visiblemente, siguiendo el criterio del método del codo. Adicionalmente, en el conjunto *All* se utilizó $K = 20$, tanto por razones de eficiencia computacional como por ser el valor más recurrente entre los subconjuntos evaluados.

Redes Neuronales

Para encontrar el número de neuronas adecuado, se procedió como se explicó en la subsección 3.8 de **Parámetros y Detalles de Cada Modelo** (págs. 88-90), lo mismo para el número de épocas. El número de épocas se escogió bajo para evitar las múltiples iteraciones dentro de cada número de capa intermedia.

Para seleccionar el número de capas intermedias, se hizo de manera exhaustiva, se evaluó el modelo con las misma configuración pero aumentando las capas, de 1 hasta 5 capas, como lo hace Gu et al. [36] y se seleccionó el número de capas que logró tener mejores resultados según las cuatro métricas, la desviación estándar de cada una y su tiempo de ejecución. En cada capa se le aplicó validación cruzada con k-folds igual a 10.

3.9.2. Metodología para la Aplicación de Pruebas Estadísticas

Con el objetivo de evaluar la existencia de diferencias estadísticamente significativas entre los métodos comparados, se aplicó una secuencia de pruebas estadísticas siguiendo un enfoque jerárquico, basado en los supuestos de normalidad y homogeneidad de varianzas.

1. **Evaluación de normalidad:** Se aplicó la prueba de Shapiro–Wilk a cada conjunto de resultados por grupo experimental. Esta prueba contrasta la hipótesis nula de normalidad de los datos en cada grupo.

2. **Evaluación de homogeneidad de varianzas:** Si los datos fueron consistentes con la normalidad, se procedió a evaluar la igualdad de varianzas entre los grupos mediante la prueba de Levene. Esta prueba permite verificar si la suposición de homocedasticidad se cumple, condición necesaria para el análisis de varianza clásico.

3. **Selección del procedimiento inferencial:**

    - Si ambas pruebas (Shapiro–Wilk y Levene) no resultaron significativas, es decir, si no se violaron los supuestos paramétricos, se aplicó un análisis de varianza ANOVA de medidas repetidas.

- En caso de que al menos uno de los supuestos fuese violado, se optó por utilizar la prueba no paramétrica de Friedman, que evalúa diferencias entre tratamientos en diseños de bloques completos mediante rangos.

4. **Comparaciones múltiples:**
   - Cuando se utilizó ANOVA y se obtuvo un resultado significativo, se realizaron comparaciones post hoc entre pares mediante la prueba t de Student.
   - Si se utilizó la prueba de Friedman, se aplicaron comparaciones por pares usando pruebas basadas en rangos y distribuciones normalizadas, ajustando el nivel de significancia mediante el método de Dunn-Bonferroni.
5. **Tamaño del efecto:** En cada contraste, se reportó el tamaño del efecto correspondiente. Para ANOVA se consideró el estadístico Q, y para Friedman se calculó el coeficiente de concordancia de Kendall W, así como el coeficiente r derivado de la estadística normalizada z en las comparaciones múltiples.

Este enfoque permitió garantizar la validez de los resultados estadísticos, adaptando los procedimientos a las características empíricas de los datos, y respetando los supuestos requeridos por cada técnica.

Capítulo 4

Resultados

4.1. Evaluaciones y Tiempo de Ejecución de los Modelos con 5 Clases

A continuación se presentan los resultados de los modelos evaluados para el caso de 5 clases en la variable objetivo, se encontrará 3 tablas para cada modelo *(All, Top, Bot)*. Se evaluará el conjunto de datos **Original** el cual sólo es la base de datos con la imputación y escalamiento con MinMax, **K-Means** para ambos casos usando la métrica de Davies-Bouldin (D-B) y Calisnki-Harabasz (C-H), y la discretización con MDLP. Se aplicará validación cruzada k-folds con $k = 10$.

Naive Bayes

Para *All* de la Tabla (4.1), Naive Bayes tiene sus mejores resultados utilizando la discretización de MDLP. En estas evaluaciones se destaca el poco tiempo de ejecución comparado a los demás modelos, sin arriesgar la precisión en los resultados.

Tabla 4.1: Aplicación de Naive Bayes a All con 5 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Original	0.292 ± 0.001	0.284 ± 0.001	0.282 ± 0.001	0.292 ± 0.001	1868
K-Means(C-H)	0.299 ± 0.001	0.293 ± 0.001	0.289 ± 0.001	0.299 ± 0.001	875
K-Means(D-B)	0.308 ± 0.002	0.296 ± 0.002	0.294 ± 0.002	0.308 ± 0.002	**811**
MDLP	$\mathbf{0.310 \pm 0.001}$	$\mathbf{0.311 \pm 0.001}$	$\mathbf{0.300 \pm 0.001}$	$\mathbf{0.310 \pm 0.001}$	1586

Para la Tabla (4.2) de *Top*, Naive Bayes disminuyó la precisión comparado a *All*, agregando que los mejores resultados los obtuvo la discretización de MDLP. Se destaca su tiempo de ejecución ya que en la mayoría de los casos tardó aproximadamente **2 min** en evaluar.

Tabla 4.2: Aplicación de Naive Bayes a Top con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0.257 ± 0.003	0.256 ± 0.003	0.256 ± 0.003	0.257 ± 0.003	283
K-Means(C-H)	0.274 ± 0.002	0.269 ± 0.003	0.265 ± 0.003	0.274 ± 0.003	127
K-Means(D-B)	0.276 ± 0.002	0.269 ± 0.002	0.265 ± 0.002	0.276 ± 0.002	121
MDLP	$\mathbf{0.281 \pm 0.001}$	$\mathbf{0.278 \pm 0.002}$	$\mathbf{0.270 \pm 0.002}$	$\mathbf{0.281 \pm 0.001}$	**118**

Las evaluaciones de *Bot* reportadas en la Tabla (4.3) de Naive Bayes tienen mejores resultados que en el *Top*, incluso compite con los resultados obtenidos en *All* siendo el mejor subconjunto para describir el comportamiento del mercado usando Naive Bayes. Los mejores resultados dentro de este subconjunto de datos fue con la discretización de K-means con la métrica de Davies-Bouldin, en este caso en particular, la discretización con MDLP se quedó por debajo de las discretizaciones con K-Means.

Tabla 4.3: Aplicación de Naive Bayes a Bot con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0.294 ± 0.002	0.282 ± 0.003	0.281 ± 0.003	0.294 ± 0.002	312
K-Means(C-H)	0.303 ± 0.004	0.286 ± 0.006	0.274 ± 0.004	0.303 ± 0.004	126
K-Means(D-B)	$\mathbf{0.315 \pm 0.003}$	$\mathbf{0.294 \pm 0.003}$	$\mathbf{0.291 \pm 0.003}$	$\mathbf{0.315 \pm 0.003}$	**119**
MDLP	0.299 ± 0.005	0.281 ± 0.007	0.264 ± 0.005	0.299 ± 0.005	128

En los tres conjuntos de datos Naive Bayes representó muy buenos resultados y tiempo de ejecución, siendo un modelo sencillo de entender, sin múltiples hiperparámetros y poco consumo de recursos computacionales.

CART

Las evaluaciones de CART para el conjunto *All* se reflejan en la tabla (4.4), las cuales fueron bajas comparadas a Naive Bayes. La evaluación de la discretización con MDLP obtuvo los mejores resultados. CART evaluó en poco tiempo de ejecución a pesar de la alta dimensionalidad de la base de datos y conservó casi los mismos valores para cada métrica de evaluación.

Tabla 4.4: Aplicación de CART a All con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0.282 ± 0.001	0.282 ± 0.001	0.282 ± 0.001	0.282 ± 0.001	5054
K-Means(C-H)	0.253 ± 0.002	0.256 ± 0.001	0.256 ± 0.002	0.253 ± 0.002	1852
K-Means(D-B)	0.256 ± 0.001	0.256 ± 0.001	0.256 ± 0.001	0.256 ± 0.001	**849**
MDLP	$\mathbf{0.284 \pm 0.001}$	$\mathbf{0.284 \pm 0.001}$	$\mathbf{0.284 \pm 0.001}$	$\mathbf{0.284 \pm 0.001}$	1601

Las evaluaciones del subconjunto *Top* usando CART no mejoraron en comparación del conjunto *All*, bajaron los valores de las métricas. En la Tabla (4.5) se reflejan los resultados, lo único rescatable de esta evaluación es el tiempo de ejecución, comparado a *All* disminuyó drásticamente. Los mejores resultados se obtuvieron con la base de datos original.

Tabla 4.5: Aplicación de CART a Top con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	$\mathbf{0.245 \pm 0.002}$	$\mathbf{0.246 \pm 0.001}$	$\mathbf{0.246 \pm 0.001}$	$\mathbf{0.246 \pm 0.001}$	598
K-Means(C-H)	0.226 ± 0.002	0.226 ± 0.002	0.226 ± 0.002	0.226 ± 0.002	168
K-Means(D-B)	0.226 ± 0.002	0.226 ± 0.002	0.226 ± 0.002	0.226 ± 0.002	**93**
MDLP	0.246 ± 0.005	0.246 ± 0.005	0.246 ± 0.005	0.246 ± 0.005	136

Los resultados en *Bot* (Tab. 4.6) fueron muy competitivos al conjunto *All* asimilándose en los resultados de las métricas. Sus mejores resultados se obtuvieron con la discretización de MDLP, aunque, al evaluar el conjunto original también obtuvo un buen resultado.

Tabla 4.6: Aplicación de CART a Bot con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0.278 ± 0.003	0.278 ± 0.003	0.278 ± 0.003	0.278 ± 0.003	516
K-Means(C-H)	0.253 ± 0.002	0.253 ± 0.002	0.253 ± 0.002	0.253 ± 0.002	145
K-Means(D-B)	0.256 ± 0.001	0.256 ± 0.001	0.256 ± 0.001	0.256 ± 0.001	**75**
MDLP	$\mathbf{0.281 \pm 0.002}$	$\mathbf{0.281 \pm 0.002}$	$\mathbf{0.281 \pm 0.002}$	$\mathbf{0.281 \pm 0.002}$	208

En general, el modelo de CART no es muy recomendable para hacer clasificación con esta base de datos, esto por sus resultados muy bajos, sin embargo, sus tiempos de ejecución son cortos, siendo una alternativa para encontrar la importancia de características hecha por el árbol de decisión, misma para analizar a trabajo futuro.

Regresión Logística

Para el conjunto *All* en regresión logística, sus evaluaciones están reportadas en la Tabla (4.7), en ella se aprecia que la discretización con MDLP fue la que mejor obtuvo resultados, sin embargo, el tiempo de ejecución fue muy grande comparado a los modelos Naive Bayes y CART, siendo aproximadamente **15 horas** de ejecución por una sola evaluación. Se destaca que la base de datos original obtuvo el mejor tiempo de ejecución a pesar de ser bastante tardada cada evaluación.

Tabla 4.7: Aplicación de Regresión Logística a All con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0.313 ± 0.001	0.299 ± 0.001	0.296 ± 0.001	0.313 ± 0.001	**6051**
K-Means(C-H)	0.275 ± 0.001	0.264 ± 0.001	0.258 ± 0.001	0.275 ± 0.001	7762
K-Means(D-B)	0.281 ± 0.002	0.270 ± 0.002	0.263 ± 0.002	0.281 ± 0.001	44207
MDLP	$\mathbf{0.321 \pm 0.001}$	$\mathbf{0.307 \pm 0.001}$	$\mathbf{0.306 \pm 0.001}$	$\mathbf{0.321 \pm 0.001}$	53335

Para *Top*, las evaluaciones están reportadas en la Tabla (4.8). Se destaca que tanto la base de datos original como la discretizada por MDLP obtuvieron resultados muy similares, sin embargo, el tiempo de evaluación destaca más en la base de datos original siendo 8 veces menor que la evaluada por MDLP.

Tabla 4.8: Aplicación de Regresión Logística a Top con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0.279 ± 0.003	$\mathbf{0.271 \pm 0.004}$	0.263 ± 0.004	0.279 ± 0.003	**1238**
K-Means(C-H)	0.233 ± 0.003	0.230 ± 0.003	0.227 ± 0.003	0.233 ± 0.002	2696
K-Means(D-B)	0.244 ± 0.002	0.238 ± 0.003	0.232 ± 0.002	0.204 ± 0.002	6419
MDLP	$\mathbf{0.279 \pm 0.002}$	$\mathbf{0.270 \pm 0.003}$	$\mathbf{0.264 \pm 0.002}$	$\mathbf{0.279 \pm 0.002}$	9329

El subconjunto de *Bot* obtuvo mejores resultados que en *All*, en la Tabla (4.9) se aprecia que la discretización con MDLP fue la más alta, pero al igual que *Top y All* tiene un tiempo de ejecución muy alto, siendo la evaluación de la base de datos original como la mejor alternativa a evaluar con Regresión Logística.

Tabla 4.9: Aplicación de Regresión Logística a Bot con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0324 ± 0.002	0.305 ± 0.003	0.306 ± 0.003	0.324 ± 0.003	**1345**
K-Means(C-H)	0.279 ± 0.003	0.261 ± 0.003	0.259 ± 0.003	0.279 ± 0.002	5945
K-Means(D-B)	0.291 ± 0.003	0.277 ± 0.003	0.274 ± 0.003	0.291 ± 0.003	7028
MDLP	$\mathbf{0.333 \pm 0.003}$	$\mathbf{0.319 \pm 0.003}$	$\mathbf{0.318 \pm 0.003}$	$\mathbf{0.333 \pm 0.003}$	6835

Regresión logística, para esta clasificación, fue muy ineficaz en el tiempo de ejecución y la vez mejoró un poco las métricas con respecto a Naive Bayes. Es uno de los modelos más tardado y de los que consume muchos recursos computacionales. Se obtuvieron buenos resultados y, al igual que CART, este modelo puede servir para extraer la importancia de las características de la base de datos, esto como futuro trabajo.

K Vecinos Más Cercanos

Las evaluaciones del conjunto de *All* están reportadas en la Tabla (4.10) donde la discretización con MDLP obtuvo los mejores resultados. Sin embargo, el costo computacional y el tiempo de ejecución fueron muy altos para obtener estos resultados. Destacando que sólo se evaluó un valor de K, no se hizo el ciclo de 2 vecinos hasta llegar a 20 como en los demás subconjuntos de datos esto debido a su alto costo computacional y tiempo de ejecución. El subconjunto de *Top* y *Bot* sirvieron para encontrar la mejor K y evaluarla en la base de datos *All*.

Tabla 4.10: Aplicación de KNN a All con 5 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Original	0.304 ± 0.001	0.300 ± 0.001	0.300 ± 0.001	0.304 ± 0.001	**2426**	20
K-Means(C-H)	0.279 ± 0.001	0.274 ± 0.001	0.274 ± 0.001	0.279 ± 0.001	39423	20
K-Means(D-B)	0.251 ± 0.001	0.248 ± 0.001	0.247 ± 0.001	0.251 ± 0.001	35130	20
MDLP	$\mathbf{0.365 \pm 0.001}$	$\mathbf{0.365 \pm 0.001}$	$\mathbf{0.364 \pm 0.001}$	$\mathbf{0.365 \pm 0.001}$	42757	20

Para el subconjunto *Top* se encontró que la discretización con MDLP reporta los mejores resultados en las métricas de evaluación, visto en la Tabla (4.11), también, su tiempo de ejecución redujo drásticamente siendo el más rápido el conjunto original, pero sigue siendo comparable al del conjunto de *All* ya que evalúa distintos números de K y se queda con la mejor K en las métricas evaluadas.

Tabla 4.11: Aplicación de KNN a Top con 5 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Original	0.256 ± 0.003	0.256 ± 0.003	0.254 ± 0.003	0.256 ± 0.003	**72**	20
K-Means(C-H)	0.246 ± 0.003	0.244 ± 0.003	0.239 ± 0.002	0.246 ± 0.003	856	6
K-Means(D-B)	0.219 ± 0.002	0.218 ± 0.001	0.215 ± 0.001	0.219 ± 0.002	761	10
MDLP	$\mathbf{0.287 \pm 0.003}$	$\mathbf{0.287 \pm 0.003}$	$\mathbf{0.285 \pm 0.003}$	$\mathbf{0.287 \pm 0.004}$	964	20

Para el subconjunto de *Bot*, sus resultados se reportan en la Tabla (4.12) donde se halla que MDLP como discretización tuvo los mejores resultados en las métricas. Siendo el subconjunto *Bot* el más adecuado para describir el mercado financiero en este modelo. Con respecto al tiempo de ejecución, se redujo comparado a *Top*.

Tabla 4.12: Aplicación de KNN a Bot con 5 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Original	0.308 ± 0.003	0.302 ± 0.003	0.301 ± 0.003	0.308 ± 0.003	**71**	19
K-Means(C-H)	0.285 ± 0.003	0.279 ± 0.003	0.278 ± 0.003	0.285 ± 0.002	769	18
K-Means(D-B)	0.253 ± 0.002	0.250 ± 0.002	0.250 ± 0.002	0.253 ± 0.002	748	19
MDLP	$\mathbf{0.331 \pm 0.004}$	$\mathbf{0.330 \pm 0.004}$	$\mathbf{0.328 \pm 0.004}$	$\mathbf{0.331 \pm 0.004}$	774	19

K vecinos más cercanos fue un modelo muy costoso y tardado para hacer clasificación con esta base de datos. Sin embargo, se obtuvieron los mejores resultados de métricas comparados a cualquier modelo aplicado. Lo complicado de este modelo es encontrar la mejor *K* ya que puede cambiar dependiendo el tipo de discretización que se haga y el conjunto de datos aplicado. Se dejó como $K = 20$ para evaluar en *All* ya que, observando las tablas de los subconjuntos, los mejores resultados fueron con *K* grandes.

Redes Neuronales

Para *All*, sus resultados en la discretización de MDLP comparado a la base de datos original fueron muy similares, estos se aprecian en la Tabla (4.13). El tiempo de ejecución fue grande porque se aplicó el ciclo de buscar las mejores *K* capas, buscando entre 1 a 5, siendo 1 o 5 capas la mejor para esta discretización.

Tabla 4.13: Aplicación de RN a All con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Original	$\mathbf{0.322 \pm 0.002}$	0.318 ± 0.008	$\mathbf{0.309 \pm 0.007}$	$\mathbf{0.321 \pm 0.002}$	**2440**	2
K-Means(C-H)	0.284 ± 0.001	0.280 ± 0.004	0.255 ± 0.003	0.284 ± 0.001	3071	5
K-Means(D-B)	0.303 ± 0.002	0.299 ± 0.007	0.277 ± 0.007	0.303 ± 0.002	3090	5
MDLP	0.323 ± 0.002	$\mathbf{0.321 \pm 0.007}$	0.309 ± 0.006	0.323 ± 0.002	3099	5

Para el subconjunto de *Top*, de nueva cuenta, las evaluaciones con MDLP y la base de datos original fueron muy cercanas unas a otras. El tiempo de ejecución fue menor comparado a *All*, 1 y 2 capas fueron los mejores resultados para la evaluación de este subconjunto, lo anterior se ve reflejado en la Tabla (4.14).

Tabla 4.14: Aplicación de RN a Top con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Original	0.274 ± 0.005	$\mathbf{0.268 \pm 0.009}$	$\mathbf{0.247 \pm 0.009}$	$\mathbf{0.275 \pm 0.004}$	**394**	1
K-Means(C-H)	0.234 ± 0.002	0.247 ± 0.063	0.195 ± 0.014	0.234 ± 0.002	451	4
K-Means(D-B)	0.238 ± 0.003	0.228 ± 0.024	0.201 ± 0.018	0.238 ± 0.003	721	4
MDLP	$\mathbf{0.274 \pm 0.004}$	0.258 ± 0.018	0.240 ± 0.014	$\mathbf{0.274 \pm 0.003}$	553	2

En *Bot* se resalta que se obtuvieron los mejores resultados en las métricas en estos 3 conjuntos de base de datos, siendo la discretización de MDLP y la base de datos original los mejores para describir las métricas de evaluación, lo anterior se observa en la Tabla (4.15), 3 y 4 capas fueron las mejores selecciones para tener un buen resultado.

Tabla 4.15: Aplicación de RN a Bot con 5 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Original	0.331 ± 0.002	0.323 ± 0.004	$\mathbf{0.312 \pm 0.006}$	0.331 ± 0.002	619	4
K-Means(C-H)	0.288 ± 0.002	0.281 ± 0.012	0.257 ± 0.012	0.288 ± 0.002	629	4
K-Means(D-B)	0.302 ± 0.005	0.301 ± 0.005	0.280 ± 0.007	0.302 ± 0.005	631	4
MDLP	$\mathbf{0.333 \pm 0.002}$	$\mathbf{0.333 \pm 0.007}$	0.312 ± 0.007	$\mathbf{0.333 \pm 0.002}$	**611**	5

Redes neuronales es un modelo muy destacado en IA [61], siendo un modelo con buenos resultados para clasificación o predicción, sin embargo, por el tiempo de ejecución, los distintos hiperparámetros del modelo y el consumo de recursos de cómputo, hace que redes neuronales no sea un método viable para clasificar en esta base de datos. Este modelo es competitivo contra Naive Bayes, Regresión Logística y KNN.

4.2. Evaluaciones y Tiempo de Ejecución de los Modelos con 3 Clases

A continuación se presentan los resultados de los modelos evaluados para el caso de 3 clases en la variable objetivo, se encontrará 3 tablas para cada modelo *(All, Top, Bot)*. Se evaluará el conjunto de datos **Original** el cual sólo es la base de datos con la imputación y escalamiento con MinMax, **K-Means** para ambos casos usando la métrica de Davies-Bouldin (D-B) y Calisnki-Harabasz (C-H), y la discretización con MDLP. Se aplicará validación cruzada k-folds con $k = 10$.

Naive Bayes

Para *All* de la Tabla (4.16), Naive Bayes tiene sus mejores resultados con la discretización de MDLP. En estas evaluaciones se destaca el poco tiempo de ejecución comparado a los demás modelos, sin arriesgar la precisión en los resultados.

Tabla 4.16: Aplicación de Naive Bayes a All con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Original	0.430 ± 0.001	0.424 ± 0.001	0.423 ± 0.001	0.430 ± 0.001	1033
K-Means(C-H)	0.435 ± 0.001	0.437 ± 0.001	0.434 ± 0.001	0.435 ± 0.001	549
K-Means(D-B)	$\mathbf{0.446 \pm 0.001}$	0.439 ± 0.001	0.440 ± 0.001	0.446 ± 0.001	507
MDLP	0.443 ± 0.001	$\mathbf{0.450 \pm 0.001}$	$\mathbf{0.444 \pm 0.001}$	$\mathbf{0.443 \pm 0.001}$	**484**

Para la Tabla (4.17) de *Top*, Naive Bayes disminuyó un poco la precisión comparado a *All*, agregando que los mejores resultados los obtuvo la discretización de MDLP. Se destaca su tiempo de ejecución ya que en la mayoría de los casos tardó menos de 2 min en evaluar y con la evaluación de MDLP consiguió el mejor tiempo de ejecución.

Tabla 4.17: Aplicación de Naive Bayes a Top con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0.390 ± 0.003	0.388 ± 0.003	0.388 ± 0.003	0.390 ± 0.003	118
K-Means(C-H)	0.413 ± 0.003	0.410 ± 0.003	0.400 ± 0.003	0.413 ± 0.002	82
K-Means(D-B)	0.415 ± 0.004	0.412 ± 0.003	0.402 ± 0.003	0.402 ± 0.003	156
MDLP	$\mathbf{0.422 \pm 0.003}$	$\mathbf{0.422 \pm 0.002}$	$\mathbf{0.412 \pm 0.002}$	$\mathbf{0.422 \pm 0.002}$	**80**

Las evaluaciones de *Bot* reportadas en la Tabla (4.18) de Naive Bayes tienen mejores resultados que en el *Top* y *All*. Los mejores resultados dentro de este subconjunto de datos fueron por parte de la discretización de K-Means con la métrica de Davies-Bouldin y también el menor tiempo de ejecución.

Tabla 4.18: Aplicación de Naive Bayes a Bot con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0.446 ± 0.003	0.437 ± 0.003	0.437 ± 0.003	0.446 ± 0.003	178
K-Means(C-H)	0.450 ± 0.003	0.444 ± 0.003	0.443 ± 0.003	0.450 ± 0.003	88
K-Means(D-B)	$\mathbf{0.462 \pm 0.003}$	$\mathbf{0.448 \pm 0.004}$	$\mathbf{0.447 \pm 0.004}$	$\mathbf{0.462 \pm 0.003}$	**73**
MDLP	0.447 ± 0.004	0.443 ± 0.004	0.440 ± 0.004	0.447 ± 0.004	105

En los tres conjuntos de datos, Naive Bayes representó muy buenos resultados y tiempo de ejecución. Además, el subconjunto *Bot* obtuvo mejores resultados y el menor tiempo de evaluación comparado a los mejores resultados en cualquier conjunto de datos.

CART

Las evaluaciones de CART para el conjunto *All* se reflejan en la Tabla (4.19). La evaluación de la discretización con MDLP obtuvo los mejores resultados en el desempeño de las métricas.

Tabla 4.19: Aplicación de CART a All con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	0.416 ± 0.001	0.416 ± 0.001	0.416 ± 0.001	0.416 ± 0.001	4340
K-Means(C-H)	0.389 ± 0.001	0.389 ± 0.001	0.389 ± 0.001	0.389 ± 0.001	1232
K-Means(D-B)	0.392 ± 0.001	0.392 ± 0.001	0.392 ± 0.001	0.392 ± 0.001	**652**
MDLP	$\mathbf{0.419 \pm 0.001}$	$\mathbf{0.419 \pm 0.001}$	$\mathbf{0.419 \pm 0.001}$	$\mathbf{0.419 \pm 0.001}$	1351

Las evaluaciones del subconjunto *Top* usando CART no mejoraron en comparación del conjunto *All*, bajaron los valores de las métricas. En la Tabla (4.20) se reflejan los resultados, lo

único rescatable de esta evaluación es el tiempo de ejecución, comparado a *All* disminuyó drásticamente. Los mejores resultados se obtuvieron con la base de datos original aunque la evaluación con la discretización con MDLP obtuvo valores muy equiparables.

Tabla 4.20: Aplicación de CART a Top con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	$\mathbf{0.377 \pm 0.002}$	$\mathbf{0.377 \pm 0.002}$	$\mathbf{0.377 \pm 0.002}$	$\mathbf{0.377 \pm 0.002}$	795
K-Means(C-H)	0.356 ± 0.002	0.356 ± 0.002	0.356 ± 0.002	0.356 ± 0.002	237
K-Means(D-B)	0.357 ± 0.002	0.357 ± 0.002	0.357 ± 0.002	0.357 ± 0.002	**128**
MDLP	0.376 ± 0.003	0.376 ± 0.003	0.376 ± 0.003	0.376 ± 0.003	141

Los resultados en *Bot* (Tab. 4.21) fueron muy competitivos al conjunto *All* asimilándose en los resultados de las métricas. Sus mejores resultados se obtuvieron con la base de datos originales y, repitiéndose lo mismo con las evaluaciones de *Top*, la evaluación con MDLP fue muy similar a la mencionada.

Tabla 4.21: Aplicación de CART a Bot con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Original	$\mathbf{0.418 \pm 0.003}$	$\mathbf{0.418 \pm 0.003}$	$\mathbf{0.418 \pm 0.003}$	$\mathbf{0.418 \pm 0.003}$	547
K-Means(C-H)	0.395 ± 0.004	0.395 ± 0.004	0.395 ± 0.004	0.395 ± 0.004	167
K-Means(D-B)	0.397 ± 0.002	0.397 ± 0.002	0.397 ± 0.002	0.397 ± 0.002	**133**
MDLP	0.417 ± 0.004	0.417 ± 0.004	0.417 ± 0.004	0.417 ± 0.004	149

En general, el modelo de CART no es muy recomendable para hacer clasificación con esta base de datos, esto por sus resultados muy bajos, sin embargo, sus tiempos de ejecución son cortos, siendo una alternativa para encontrar la importancia de características hecha por el árbol de decisión.

Regresión Logística

Para el conjunto *All* en regresión logística, sus evaluaciones están reportadas en la Tabla (4.22), en ella se aprecia que la discretización con MDLP fue la que mejor obtuvo resultados, sin embargo, el tiempo de ejecución fue muy grande comparado a los demás modelos, siendo aproximadamente 5 horas de ejecución por una sola evaluación. Por otro lado, evaluando la base de datos original se consiguió un resultado muy similar al de MDLP, destacando su tiempo de ejecución que fue menor por 3 veces al de MDLP.

Tabla 4.22: Aplicación de Regresión Logística a All con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Original	0.455 ± 0.001	0.447 ± 0.001	0.443 ± 0.001	0.455 ± 0.001	6358
K-Means(C-H)	0.415 ± 0.001	0.410 ± 0.001	0.409 ± 0.001	0.415 ± 0.001	**4379**
K-Means(D-B)	0.423 ± 0.001	0.417 ± 0.001	0.414 ± 0.001	0.423 ± 0.001	30430
MDLP	$\mathbf{0.462 \pm 0.001}$	$\mathbf{0.454 \pm 0.001}$	$\mathbf{0.455 \pm 0.001}$	$\mathbf{0.462 \pm 0.001}$	19937

Para *Top*, las evaluaciones están reportadas en la Tabla (4.23). Se destaca que la base de datos original obtuvo un resultado muy similar al de MDLP, y la base de datos original tiene un tiempo de ejecución muy menor al de las otras evaluaciones, siendo el mejor candidato a evaluar en Regresión Logística.

Tabla 4.23: Aplicación de Regresión Logística a Top con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Original	0.419 ± 0.005	0.415 ± 0.005	0.404 ± 0.005	0.419 ± 0.005	**840**
K-Means(C-H)	0.369 ± 0.004	0.366 ± 0.003	0.362 ± 0.003	0.369 ± 0.003	1880
K-Means(D-B)	0.379 ± 0.003	0.377 ± 0.003	0.369 ± 0.003	0.379 ± 0.003	6195
MDLP	$\mathbf{0.422 \pm 0.004}$	$\mathbf{0.417 \pm 0.004}$	$\mathbf{0.414 \pm 0.004}$	$\mathbf{0.422 \pm 0.004}$	5709

El subconjunto de *Bot* obtuvo mejores resultados que en *All*, en la Tabla (4.24) se aprecia que la discretización con MDLP fue la más alta en las métricas de evaluación. Destacando nuevamente los valores muy cercanos de las métricas entre la base de datos original y MDLP, con menor tiempo de ejecución la base de datos original comparado a cualquier otra evaluación.

Tabla 4.24: Aplicación de Regresión Logística a Bot con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Original	0.472 ± 0.003	0.459 ± 0.004	0.458 ± 0.003	0.472 ± 0.003	**1027**
K-Means(C-H)	0.431 ± 0.004	0.421 ± 0.003	0.419 ± 0.003	0.431 ± 0.003	2735
K-Means(D-B)	0.440 ± 0.004	0.430 ± 0.004	0.429 ± 0.004	0.440 ± 0.004	5063
MDLP	$\mathbf{0.480 \pm 0.002}$	$\mathbf{0.467 \pm 0.002}$	$\mathbf{0.464 \pm 0.002}$	$\mathbf{0.480 \pm 0.002}$	4560

Regresión logística, para esta clasificación, fue en varios casos ineficaz en el tiempo de ejecución, sin embargo, si sólo se evalúa la base de datos original, se obtienen buenos resultados en cuestión del rendimiento del modelo y su tiempo de ejecución. Este modelo puede servir para extraer la importancia de las características de la base de datos.

K Vecinos Más Cercanos

Las evaluaciones del conjunto de *All* están reportadas en la Tabla (4.25) donde la discretización con MDLP obtuvo los mejores resultados. Sin embargo, el costo computacional y el tiempo

de ejecución fueron muy altos para obtener esos resultados. Destacando que sólo se evaluó un valor de *K*, no se hizo el ciclo de 2 vecinos hasta llegar a 20 como en los demás conjuntos de datos. El subconjunto de *Top* y *Bot* sirvieron para encontrar la mejor *K* y evaluarla en la base de datos *All*, coincidiendo ambos subconjuntos que la *K* óptima fuera la más alta (20).

Tabla 4.25: Aplicación de KNN a All con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Original	0.444 ± 0.001	0.438 ± 0.001	0.438 ± 0.001	0.444 ± 0.001	**2163**	20
K-Means(C-H)	0.420 ± 0.001	0.415 ± 0.001	0.415 ± 0.001	0.420 ± 0.001	36406	20
K-Means(D-B)	0.394 ± 0.001	0.393 ± 0.001	0.391 ± 0.001	0.394 ± 0.001	36935	20
MDLP	$\mathbf{0.513 \pm 0.001}$	$\mathbf{0.513 \pm 0.001}$	$\mathbf{0.512 \pm 0.001}$	$\mathbf{0.513 \pm 0.001}$	38648	20

Para el subconjunto *Top* se encontró que la discretización con MDLP reporta los mejores resultados en las métricas de evaluación, visto en la Tabla (4.26), también, su tiempo de ejecución redujo drásticamente, pero sigue siendo comparable al del conjunto de *All* ya que evalúa distintos números de *K*. Se destaca que el tiempo de ejecución de la base de datos original fue el que obtuvo el menor tiempo de evaluación.

Tabla 4.26: Aplicación de KNN a Top con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Original	0.392 ± 0.005	0.388 ± 0.005	0.387 ± 0.005	0.392 ± 0.005	**75**	19
K-Means(C-H)	0.379 ± 0.004	0.377 ± 0.004	0.375 ± 0.004	0.379 ± 0.004	916	9
K-Means(D-B)	0.351 ± 0.002	0.350 ± 0.003	0.347 ± 0.003	0.351 ± 0.002	810	4
MDLP	$\mathbf{0.437 \pm 0.004}$	$\mathbf{0.436 \pm 0.004}$	$\mathbf{0.435 \pm 0.004}$	$\mathbf{0.437 \pm 0.004}$	761	19

Para el subconjunto de *Bot*, sus resultados se reportan en la Tabla (4.27) donde se halla que MDLP tuvo los mejores resultados en las métricas. Y, de nueva cuenta, la base de datos original tuvo el menor tiempo de ejecución, incluso su rendimiento en las métricas de evaluación son comparables a las de MDLP.

Tabla 4.27: Aplicación de KNN a Bot con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Original	0.461 ± 0.004	0.451 ± 0.004	0.450 ± 0.004	0.461 ± 0.004	**77**	20
K-Means(C-H)	0.437 ± 0.003	0.430 ± 0.003	0.429 ± 0.003	0.437 ± 0.003	818	19
K-Means(D-B)	0.403 ± 0.003	0.399 ± 0.003	0.399 ± 0.003	0.403 ± 0.003	794	18
MDLP	$\mathbf{0.476 \pm 0.005}$	$\mathbf{0.472 \pm 0.004}$	$\mathbf{0.470 \pm 0.004}$	$\mathbf{0.476 \pm 0.004}$	758	19

K vecinos más cercanos fue un modelo muy costo y tardado para hacer clasificación con esta base de datos. Sin embargo, se obtuvieron los mejores resultados de métricas comparados a

cualquier modelo aplicado. Lo complicado de este modelo es encontrar la mejor *K* ya que puede cambiar dependiendo el tipo de discretización que se haga y el conjunto de datos aplicado. Se destaca que si evalúa la base de datos original se obtienen mejores resultados equilibrados en cuestión del rendimiento de las métricas y el tiempo de ejecución.

Redes Neuronales

Para *All*, sus resultados se comportan muy similarmente a la base de datos original y con la discretización de MDLP, estos se aprecian en la Tabla (4.28), el más alto en sus métricas fue con la discretización de MDLP, sin embargo, el tiempo de ejecución fue grande porque se aplicó el ciclo de buscar las mejores *K* capas, buscando entre 1 a 5, siendo 3 y 4 capas las mejores para obtener los mejores resultados.

Tabla 4.28: Aplicación de RN a All con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Original	0.461 ± 0.003	0.458 ± 0.008	$\mathbf{0.453 \pm 0.007}$´	0.461 ± 0.003	3316	1
K-Means(C-H)	0.424 ± 0.001	0.425 ± 0.005	0.421 ± 0.002	0.424 ± 0.001	**2911**	5
K-Means(D-B)	0.441 ± 0.002	0.442 ± 0.004	0.438 ± 0.004	0.441 ± 0.001	2962	4
MDLP	$\mathbf{0.463 \pm 0.002}$	$\mathbf{0.458 \pm 0.005}$	0.451 ± 0.012	$\mathbf{0.463 \pm 0.002}$	3301	4

Para el subconjunto de *Top*, de nueva cuenta, las distintas evaluaciones fueron muy cercanas unas a otras, la mejor fue utilizando MDLP, pero la evaluación de la base de datos original también obtuvo resultados buenos. El tiempo de ejecución fue menor comparado a *All*, y 2 capas fue el mejor resultado para esta evaluación, lo anterior se ve reflejado en la Tabla (4.29).

Tabla 4.29: Aplicación de RN a Top con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Original	$\mathbf{0.414 \pm 0.004}$	0.414 ± 0.010	$\mathbf{0.390 \pm 0.021}$	$\mathbf{0.414 \pm 0.004}$	**409**	1
K-Means(C-H)	0.368 ± 0.003	0.313 ± 0.064	0.309 ± 0.026	0.368 ± 0.003	477	5
K-Means(D-B)	0.373 ± 0.004	0.356 ± 0.035	0.338 ± 0.020	0.373 ± 0.003	766	5
MDLP	0.416 ± 0.007	$\mathbf{0.418 \pm 0.008}$	0.388 ± 0.022	$\mathbf{0.416 \pm 0.006}$	456	1

En *Bot* se resalta que se obtuvieron los mejores resultados en las métricas en estos 3 conjuntos de base de datos, siendo la discretización de MDLP y la base de datos original los mejores para describir las métricas de evaluación, en la Tabla (4.30) se aprecia que en ambos casos, 1 y 2 capas fueron las mejores selecciones para tener un buen resultado.

Tabla 4.30: Aplicación de RN a Bot con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Original	0.480 ± 0.003	0.474 ± 0.005	$\mathbf{0.467 \pm 0.009}$	0.480 ± 0.003	**476**	1
K-Means(C-H)	0.442 ± 0.004	0.442 ± 0.006	0.429 ± 0.008	0.442 ± 0.003	662	4
K-Means(D-B)	0.453 ± 0.003	0.451 ± 0.006	0.440 ± 0.012	0.453 ± 0.004	633	4
MDLP	$\mathbf{0.483 \pm 0.004}$	$\mathbf{0.478 \pm 0.007}$	0.468 ± 0.013	$\mathbf{0.483 \pm 0.003}$	639	4

Redes neuronales es un modelo muy destacado en IA [61], siendo un modelo con buenos resultados para clasificación o predicción, sin embargo, por el tiempo de ejecución, los distintos hiperparámetros del modelo y el consumo de recursos de cómputo, hace que redes neuronales no sea un método viable para clasificar en esta base de datos. Además, los demás modelos, en su mayoría, representaron buenos resultados sin tanta búsqueda de los mejores parámetros para esta base de datos.

4.3. Evaluaciones con Selección/Reducción de Características con 5 Clases

A continuación, se mostrará las tablas con los resultados de las evaluaciones con selección de características según [22], [35] y [47], y de reducción de dimensionalidad con PCA. En estas evaluaciones destacan qué modelos y características son las más importantes para modelar esta clasificación de 5 clases. Para la evaluación de la selección de características únicamente se muestra el mejor modelo de todas las discretizaciones. En cada evaluación se aplicó validación cruzada con k-folds de 10.

Naive Bayes

Para *All* se muestra en la Tabla (4.31) que el método de selección de características con Redes Bayesiandas fue el mejor en el tiempo de ejecución y en la descripción de las métricas de evaluación. Por otro lado, PCA muestra un pésimo rendimiento con el algoritmo de Naive Bayes.

Tabla 4.31: Evaluación de All con Naive Bayes para Seleccionar/Reducir Características con 5 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	$\mathbf{0.349 \pm 0.001}$	$\mathbf{0.340 \pm 0.001}$	$\mathbf{0.340 \pm 0.001}$	$\mathbf{0.349 \pm 0.001}$	**127**
PCA	0.199 ± 0.000	0.040 ± 0.000	0.066 ± 0.000	0.200 ± 0.000	211
Fts MDLP [47]	0.304 ± 0.001	0.295 ± 0.001	0.293 ± 0.001	0.304 ± 0.001	326
Fts MDLP [35]	0.344 ± 0.001	0.335 ± 0.001	0.332 ± 0.001	0.344 ± 0.001	305

Para *Top* de la Tabla (4.32) se observa que las selecciones de características son relativamente similares, predominando la selección por [35] por una centésima, en tiempo de ejecución dominó la selección por MDL-FS [22].

Tabla 4.32: Evaluación de Top con Naive Bayes para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Redes Bayesianas [22]	0.271 ± 0.003	0.264 ± 0.003	0.256 ± 0.003	0.271 ± 0.003	**23**
PCA	0.196 ± 0.001	0.039 ± 0.000	0.066 ± 0.000	0.200 ± 0.000	35
Fts MDLP [47]	0.270 ± 0.004	0.263 ± 0.003	0.256 ± 0.003	0.270 ± 0.003	42
Fts MDLP [35]	$\mathbf{0.281 \pm 0.002}$	$\mathbf{0.274 \pm 0.002}$	$\mathbf{0.267 \pm 0.002}$	$\mathbf{0.281 \pm 0.002}$	34

Para *Bot* de la tabla (4.33) se aprecia que la selección de características por [35] y [22] son las que mejores resultados dan, mientras que [22] da el mejor resultado en tiempo de ejecución.

Tabla 4.33: Evaluación de Bot con Naive Bayes para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Redes Bayesianas [22]	0.323 ± 0.004	0.301 ± 0.005	0.296 ± 0.004	0.323 ± 0.004	**22**
PCA	0.197 ± 0.001	0.039 ± 0.000	0.066 ± 0.000	0.200 ± 0.000	33
Fts (C-H) [47]	0.315 ± 0.004	0.298 ± 0.004	0.291 ± 0.003	0.315 ± 0.003	28
Fts (D-B) [35]	$\mathbf{0.324 \pm 0.003}$	$\mathbf{0.302 \pm 0.003}$	$\mathbf{0.298 \pm 0.003}$	$\mathbf{0.324 \pm 0.002}$	35

Naives Bayes tuvo muchas complicaciones para usar la base de datos con PCA, arrojó resultados muy bajos comparados a los otros métodos. Mientras que la selección por [22] y [35] tuvieron resultados altos en el rendimiento de las métricas, y [22] fue el único con poco tiempo de ejecución.

CART

Para *All*, se aprecia en la Tabla (4.34) que Redes Bayesianas fue el mejor método en la evaluación del modelo y en el tiempo de ejecución, sin embargo CART muestra deficiencia en clasificar esta base de datos.

Tabla 4.34: Evaluación de All con CART para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Redes Bayesianas [22]	$\mathbf{0.286 \pm 0.001}$	$\mathbf{0.287 \pm 0.001}$	$\mathbf{0.287 \pm 0.001}$	$\mathbf{0.286 \pm 0.001}$	**205**
PCA	0.248 ± 0.001	0.248 ± 0.001	0.248 ± 0.001	0.248 ± 0.001	1279
Fts MDLP [47]	0.260 ± 0.001	0.260 ± 0.001	0.260 ± 0.001	0.260 ± 0.001	238
Fts MDLP [35]	0.282 ± 0.001	0.282 ± 0.001	0.282 ± 0.001	0.282 ± 0.001	496

Para *Top* de la Tabla (4.35) se aprecia que CART tuvo un bajo rendimiento en la selección de características por parte de [47] y reducción de dimensionalidad con PCA. Los mejores resultados los obtuvo [35], mientras que [22] tuvo el mejor rendimiento en tiempo de ejecución y obtuvo valores muy próximos a [35].

Tabla 4.35: Evaluación de Top con CART para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Redes Bayesianas [22]	0.236 ± 0.002	0.236 ± 0.002	0.236 ± 0.002	0.236 ± 0.002	**16**
PCA	0.226 ± 0.003	0.226 ± 0.003	0.226 ± 0.006	0.226 ± 0.003	178
Fts MDLP[47]	0.232 ± 0.003	0.232 ± 0.003	0.232 ± 0.003	0.232 ± 0.003	27
Fts MDLP[35]	$\mathbf{0.244 \pm 0.003}$	$\mathbf{0.245 \pm 0.003}$	$\mathbf{0.244 \pm 0.003}$	$\mathbf{0.244 \pm 0.003}$	33

Para *Bot* de la Tabla (4.36) se rescata el mismo comportamiento que *Top* donde la selección por parte de [35] obtuvo los mejores resultados y [22] obtuvo el mejor rendimiento en tiempo de ejecución.

Tabla 4.36: Evaluación de Bot con CART para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Redes Bayesianas [22]	0.264 ± 0.003	0.264 ± 0.002	0.264 ± 0.003	0.264 ± 0.003	**19**
PCA	0.259 ± 0.003	0.259 ± 0.003	0.259 ± 0.003	0.259 ± 0.003	162
Fts MDLP[47]	0.265 ± 0.003	0.265 ± 0.003	0.265 ± 0.003	0.265 ± 0.003	40
Fts MDLP[35]	$\mathbf{0.280 \pm 0.003}$	$\mathbf{0.280 \pm 0.003}$	$\mathbf{0.280 \pm 0.003}$	$\mathbf{0.280 \pm 0.003}$	46

CART, como en las previas secciones, ha demostrado un bajo rendimiento ante esta problemática de clasificación. No importa si se le aplica una selección de características o reducción de dimensionalidad, el algoritmo queda como insuficiente. Los mejores métodos para seleccionar características fueron [22] y [35].

Regresión Logística

Para *All*, en la Tabla (4.37) se aprecia que este modelo es muy costoso en tiempo de ejecución para las bases de datos originales o los datos discretizados, sin embargo, si se aplica PCA se consigue un buen rendimiento entre el tiempo de ejecución y las métricas de evaluación.

Tabla 4.37: Evaluación de All con Regresión Logística para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Redes Bayesianas [22]	0.309 ± 0.001	0.288 ± 0.001	0.284 ± 0.001	0.309 ± 0.001	13595
PCA	0.296 ± 0.001	0.277 ± 0.002	0.269 ± 0.001	0.296 ± 0.001	**507**
Fts MDLP [47]	$0.\ 299 \pm 0.001$	0.284 ± 0.001	0.284 ± 0.001	0.299 ± 0.001	16000
Fts MDLP [35]	$\mathbf{0.316 \pm 0.001}$	$\mathbf{0.299 \pm 0.001}$	$\mathbf{0.298 \pm 0.001}$	$\mathbf{0.316 \pm 0.001}$	20704

Para *Top* se muestra en la Tabla (4.38). Para cualquier selección de características y reducción de dimensionalidad se obtuvieron resultados muy similares, PCA destaca más por su tiempo de ejecución menor al de todos, haciendo un buen equilibrio entre rendimiento de las métricas y tiempo de ejecución, esto se da por cómo funciona Regresión Logística ante bases de datos con características numéricas.

Tabla 4.38: Evaluación de Top con Regresión Logística para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Redes Bayesianas [22]	0.270 ± 0.004	0.258 ± 0.004	0.245 ± 0.004	0.270 ± 0.004	464
PCA	0.257 ± 0.002	0.249 ± 0.002	0.234 ± 0.003	0.259 ± 0.002	**171**
Fts MDLP [47]	0.265 ± 0.003	0.256 ± 0.003	0.247 ± 0.003	0.266 ± 0.002	1144
Fts MDLP [35]	$\mathbf{0.273 \pm 0.004}$	$\mathbf{0.263 \pm 0.004}$	$\mathbf{0.254 \pm 0.004}$	$\mathbf{0.274 \pm 0.003}$	2283

Para *Bot* sus resultados se reflejan en la Tabla (4.39), en donde pasa lo mismo que con *Top*, las distintas selecciones de características y reducción de dimensionalidad obtienen resultados muy próximos entre sí. Destacando nuevamente PCA por su poco tiempo de ejecución comparado a los otros casos.

Tabla 4.39: Evaluación de Bot con Regresión Logística para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**
Redes Bayesianas [22]	0.319 ± 0.003	0.301 ± 0.003	0.298 ± 0.003	0.319 ± 0.003	691
PCA	0.312 ± 0.002	0.289 ± 0.002	0.287 ± 0.002	0.313 ± 0.00	**38**
Fts MDLP[47]	0.309 ± 0.003	0.291 ± 0.003	0.288 ± 0.003	0.309 ± 0.003	340
Fts MDLP[35]	$\mathbf{0.330 \pm 0.003}$	$\mathbf{0.314 \pm 0.003}$	$\mathbf{0.312 \pm 0.003}$	$\mathbf{0.330 \pm 0.003}$	2609

En general, Regresión Logística es un modelo complejo si no se le aplica un preprocesamiento antes a la base de datos, por ejemplo, en esta sección los resultados más equilibrados fueron con PCA, tanto en el rendimiento de las métricas de evaluación, así como en el tiempo de ejecución.

K Vecinos Más Cercanos

Para *All*, la selección de características por [22] arroja los mejores resultados tanto en las métricas de evaluación así como en el tiempo de ejecución, incluso, se vuelve óptimo al disminuir tanto su tiempo de ejecución comparado a las demás selecciones de características, lo anterior se aprecia en la Tabla (4.40).

Tabla 4.40: Evaluación de All con KNN para Seleccionar/Reducir Características con 5 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Redes Bayesianas [22]	**0.373 ± 0.001**	**0.372 ± 0.001**	**0.369 ± 0.001**	**0.373 ± 0.001**	**402**	20
PCA	0.289 ± 0.001	0.284 ± 0.001	0.284 ± 0.001	0.289 ± 0.001	665	20
Fts MDLP [47]	0.307 ± 0.001	0.304 ± 0.001	0.304 ± 0.001	0.307 ± 0.001	3966	20
Fts MDLP [35]	0.354 ± 0.001	0.354 ± 0.001	0.352 ± 0.001	0.354 ± 0.001	34527	20

Para *Top* en la Tabla (4.41) se observa que la selección de características por [35] domina en el rendimiento de las métricas de los demás, sin embargo, PCA es el método que tiene el menor tiempo de ejecución.

Tabla 4.41: Evaluación de Top con KNN para Seleccionar/Reducir Características con 5 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Redes Bayesianas [22]	0.266 ± 0.002	0.265 ± 0.002	0.264 ± 0.002	0.266 ± 0.001	**139**	19
PCA	0.248 ± 0.003	0.246 ± 0.003	0.245 ± 0.003	0.248 ± 0.003	155	18
Fts MDLP [47]	0.263 ± 0.003	0.263 ± 0.003	0.262 ± 0.003	0.263 ± 0.003	444	19
Fts MDLP [35]	**0.284 ± 0.004**	**0.283 ± 0.004**	**0.282 ± 0.004**	**0.284 ± 0.004**	743	20

Para *Bot* en la Tabla (4.42), se nota que se repite el mismo patrón que en *Top*, donde la selección de características por [35] domina en el rendimiento de las métricas de los demás, mientras que PCA le da los datos de una forma más accesible a KNN.

Tabla 4.42: Evaluación de Bot con KNN para Seleccionar/Reducir Características con 5 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Redes Bayesianas [22]	0.310 ± 0.003	0.306 ± 0.003	0.302 ± 0.003	0.310 ± 0.003	**105**	20
PCA	0.304 ± 0.004	0.298 ± 0.004	0.297 ± 0.004	0.304 ± 0.004	190	20
Fts MDLP [47]	0.309 ± 0.003	0.304 ± 0.003	0.302 ± 0.003	0.309 ± 0.003	172	20
Fts MDLP [35]	**0.331 ± 0.003**	**0.331 ± 0.003**	**0.327 ± 0.003**	**0.331 ± 0.003**	726	19

KNN es un algoritmo del cual puede tener complicaciones dependiendo las características y complejidad de la base de datos. El uso de Redes Bayesianas por parte de [22] ha demostrado ser la configuración más aceptada por este modelo, teniendo un buen equilibrio entre el rendimiento de las métricas de evaluación y el tiempo de ejecución.

Redes Neuronales

Para *All* en la Tabla (4.43) se muestran los resultados, estos reflejan que los cuatro métodos son muy similares en tiempo de ejecución y rendimiento en las métricas. En los cuatro casos se requirió al menos 3, 4 ó 5 capas ocultas para obtener los mejores resultados.

Tabla 4.43: Evaluación de All con RN para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Redes Bayesianas [22]	0.312 ± 0.002	0.312 ± 0.006	0.295 ± 0.009	0.312 ± 0.002	3867	4
PCA	0.309 ± 0.001	0.305 ± 0.005	0.292 ± 0.003	0.309 ± 0.003	**2771**	3
Fts MDLP [47]	0.313 ± 0.002	0.307 ± 0.005	0.298 ± 0.007	0.313 ± 0.002	3783	5
Fts MDLP [35]	$\mathbf{0.319 \pm 0.003}$	$\mathbf{0.319 \pm 0.009}$	$\mathbf{0.304 \pm 0.010}$	$\mathbf{0.319 \pm 0.003}$	3676	5

Para *Top* se obtuvieron resultados en las métricas y el tiempo de ejecución muy similares en cualquier método de selección de características, esto se muestra en la Tabla (4.44). Además, la mayoría de casos obtuvieron los mejores resultados con dos o más capas ocultas.

Tabla 4.44: Evaluación de Top con RN para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Redes Bayesianas [22]	0.274 ± 0.003	0.273 ± 0.006	0.251 ± 0.008	0.274 ± 0.003	481	3
PCA	0.263 ± 0.003	0.260 ± 0.006	0.252 ± 0.005	0.263 ± 0.003	511	2
Fts MDLP [47]	0.270 ± 0.002	0.268 ± 0.005	0.253 ± 0.006	0.270 ± 0.003	**457**	2
Fts MDLP [35]	$\mathbf{0.280 \pm 0.002}$	$\mathbf{0.279 \pm 0.005}$	$\mathbf{0.261 \pm 0.010}$	$\mathbf{0.280 \pm 0.003}$	516	4

Para *Bot*, se obtuvo un poco diferente los resultados de las métricas como en *Top*, sólo la selección de carcterísticas con [22] y [35] se obtuvieron los mejores resultados, esto se muestra en la Tabla (4.45). En estas evaluaciones, de nueva cuenta, apuntaron a usar pocas capas ocultas.

Tabla 4.45: Evaluación de Bot con RN para Seleccionar/Reducir Características con 5 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Redes Bayesianas [22]	0.326 ± 0.003	0.325 ± 0.004	0.303 ± 0.011	0.326 ± 0.003	823	2
PCA	0.325 ± 0.003	0.318 ± 0.006	0.309 ± 0.007	0.325 ± 0.002	550	2
Fts MDLP [47]	0.325 ± 0.005	0.322 ± 0.005	0.306 ± 0.007	0.325 ± 0.004	445	3
Fts MDLP [35]	$\mathbf{0.333 \pm 0.003}$	$\mathbf{0.335 \pm 0.006}$	$\mathbf{0.319 \pm 0.006}$	$\mathbf{0.333 \pm 0.003}$	**487**	2

Redes Neuronales es un algoritmo que sigue siendo muy costoso en recursos y tiempo de ejecución para esta tarea de clasificación, aunque en este caso se equiparo en términos de los valores en las métricas al mejor algoritmo que es KNN.

4.4. Evaluaciones con Selección/Reducción de Características con 3 Clases

A continuación, se mostrará las tablas con los resultados de las evaluaciones con selección de características según [22], [35] y [47], y de reducción de dimensionalidad con PCA. En estas evaluaciones destacan qué modelos y características son las más importantes para modelar esta clasificación de $k = 3$.

Naive Bayes

Para *All*, Naive Bayes obtuvo los mejores resultados con la selección de características usando Redes Bayesianas. Los resultados están totalmente equilibrados entre la calidad de las métricas de evaluación y el tiempo de ejecución, esto se muestra en la Tabla (4.46).

Tabla 4.46: Evaluación de All con Naive Bayes para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	**0.493 ± 0.001**	**0.489 ± 0.001**	**0.489 ± 0.001**	**0.493 ± 0.001**	**91**
PCA	0.332 ± 0.001	0.111 ± 0.000	0.166 ± 0.000	0.333 ± 0.000	178
Fts MDLP [47]	0.443 ± 0.001	0.437 ± 0.001	0.437 ± 0.001	0.443 ± 0.001	124
Fts MDLP [35]	0.486 ± 0.001	0.482 ± 0.001	0.482 ± 0.001	0.486 ± 0.001	269

Para describir el subconjunto de *Top* se muestra la Tabla (4.47). Se aprecia que la selección de características con MDL-FS [22] es el tiene los mejores rendimientos en las métricas de evaluación y el tiempo de evaluación más corto, la selección con [35] tiene valores muy próximos al método de redes bayesianas.

Tabla 4.47: Evaluación de Top con Naive Bayes para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	**0.429 ± 0.006**	**0.427 ± 0.006**	**0.418 ± 0.005**	**0.429 ± 0.005**	**17**
PCA	0.330 ± 0.002	0.110 ± 0.001	0.166 ± 0.001	0.333 ± 0.000	21
Fts MDLP [47]	0.408 ± 0.003	0.405 ± 0.003	0.392 ± 0.003	0.408 ± 0.003	**17**
Fts MDLP [35]	0.425 ± 0.004	0.424 ± 0.003	0.413 ± 0.003	0.425 ± 0.003	20

Para *Bot* se desglosa en la Tabla (4.48), la cual [35] y redes bayesianas [22] dan los mejores resultados con este subconjunto, siendo equilibrado en el rendimiento de las métricas de evaluación y el tiempo de ejecución.

Tabla 4.48: Evaluación de Bot con Naive Bayes para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	**0.477 ± 0.004**	**0.462 ± 0.004**	**0.453 ± 0.004**	**0.477 ± 0.003**	21
PCA	0.331 ± 0.001	0.110 ± 0.000	0.166 ± 0.000	0.333 ± 0.000	21
Fts (C-H) [47]	0.463 ± 0.003	0.451 ± 0.003	0.449 ± 0.003	0.471 ± 0.003	**20**
Fts (D-B) [35]	0.471 ± 0.003	0.452 ± 0.004	0.445 ± 0.004	0.474 ± 0.002	21

El algoritmo de Naive Bayes arroja muy buenos resultados con poco tiempo de ejecución y consumo muy bajo de recursos computacionales. Para $k = 3$ en esta selección de características, tuvo sus resultados un poco variados entre [22] y [35].

CART

Para *All*, la selección de características con Redes Bayesianas mejoró ligeramente los resultados en las métricas de evaluación, además, consiguió el menor tiempo de ejecución comparado a los otros modelos, como se muestra en la Tabla (4.49).

Tabla 4.49: Evaluación de All con CART para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	**0.426 ± 0.002**	**0.426 ± 0.002**	**0.426 ± 0.002**	**0.426 ± 0.002**	**185**
PCA	0.384 ± 0.001	0.384 ± 0.001	0.384 ± 0.001	0.384 ± 0.001	1630
Fts MDLP [47]	0.394 ± 0.002	0.394 ± 0.002	0.394 ± 0.002	0.394 ± 0.002	199
Fts MDLP [35]	0.416 ± 0.001	0.416 ± 0.001	0.416 ± 0.001	0.416 ± 0.001	291

Para *Top* de la Tabla (4.50), se aprecia que los mejores resultados se obtienen con MDL-FS, tanto en rendimiento medido con las métricas de evaluación, así como el tiempo de ejecución de la evaluación.

Tabla 4.50: Evaluación de Top con CART para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	**0.385 ± 0.003**	**0.385 ± 0.003**	**0.385 ± 0.003**	**0.385 ± 0.003**	**18**
PCA	0.356 ± 0.002	0.356 ± 0.002	0.356 ± 0.002	0.356 ± 0.002	168
Fts MDLP[47]	0.364 ± 0.002	0.364 ± 0.002	0.364 ± 0.002	0.364 ± 0.002	20
Fts MDLP[35]	0.375 ± 0.003	0.375 ± 0.003	0.375 ± 0.003	0.375 ± 0.003	30

Para *Bot* en la Tabla (4.51) se observa que los resultados en las métricas fueron muy similares entre [22] y [35], donde [35] fue ligeramente mejor.

Tabla 4.51: Evaluación de Bot con CART para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	0.414 ± 0.002	0.414 ± 0.002	0.414 ± 0.002	0.414 ± 0.002	**20**
PCA	0.399 ± 0.002	0.399 ± 0.002	0.399 ± 0.002	0.399 ± 0.002	155
Fts MDLP[47]	0.405 ± 0.003	0.404 ± 0.003	0.404 ± 0.003	0.405 ± 0.003	24
Fts MDLP[35]	$\mathbf{0.417 \pm 0.003}$	$\mathbf{0.416 \pm 0.003}$	$\mathbf{0.416 \pm 0.003}$	$\mathbf{0.417 \pm 0.003}$	32

A pesar de que la selección de características en los activos financieros se usó para mejorar los rendimientos de las métricas de evaluación y reducir el tiempo de ejecución para dicha evaluación, CART no es un buen algoritmo para clasificar esta base de datos, dando resultados muy bajos en el rendimiento, pero buenos en el tiempo de ejecución.

Regresión Logística

Para *All* en la tabla (4.52) se observan los distintos resultados al evaluar el modelo. Se encuentra que el modelo es muy complejo para selección de características con los datos discretizados, sin embargo con PCA muestra un buen equilibrio entre el rendimiento de las métricas de evaluación y el tiempo de ejecución del modelo.

Tabla 4.52: Evaluación de All con Regresión Logística para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	0.455 ± 0.001	0.440 ± 0.001	0.436 ± 0.002	0.455 ± 0.001	10458
PCA	0.442 ± 0.001	0.432 ± 0.001	0.422 ± 0.001	0.442 ± 0.001	**907**
Fts MDLP [47]	0.442 ± 0.001	0.430 ± 0.001	0.427 ± 0.001	0.442 ± 0.001	10330
Fts MDLP [35]	$\mathbf{0.459 \pm 0.001}$	$\mathbf{0.449 \pm 0.001}$	$\mathbf{0.449 \pm 0.001}$	$\mathbf{0.459 \pm 0.001}$	13062

Para *Top* en la selección de características o reducción de dimensionalidad, la Tabla (4.53) refleja los resultados, donde la selección con [35] da los mejores resultados en las métricas de evaluación, pero PCA da los mejores resultados en tiempo de ejecución siendo una alternativa para equilibrar rendimiento y tiempo. También, las redes bayesianas con MDL-FS tuvieron resultados muy próximos al de [35].

Tabla 4.53: Evaluación de Top con Regresión Logística para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	0.412 ± 0.004	0.404 ± 0.005	0.396 ± 0.004	0.412 ± 0.004	1409
PCA	0.381 ± 0.003	0.376 ± 0.003	0.366 ± 0.002	0.381 ± 0.003	**28**
Fts MDLP [47]	0.406 ± 0.003	0.401 ± 0.003	0.396 ± 0.003	0.406 ± 0.003	290
Fts MDLP [35]	$\mathbf{0.415 \pm 0.004}$	$\mathbf{0.408 \pm 0.003}$	$\mathbf{0.403 \pm 0.003}$	$\mathbf{0.415 \pm 0.003}$	1421

Para *Bot* los resultados se muestran en la Tabla (4.54), los mejores resultados los obtuvo la selección de [35], pero la selección con [22] muestran los resultados más equilibrados en rendimiento de las métricas y tiempo de ejecución.

Tabla 4.54: Evaluación de Bot con Regresión Logística para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Redes Bayesianas [22]	0.471 ± 0.003	0.453 ± 0.003	0.449 ± 0.003	0.471 ± 0.002	226
PCA	0.462 ± 0.003	0.443 ± 0.003	0.437 ± 0.003	0.462 ± 0.003	**45**
Fts MDLP[47]	0.463 ± 0.002	0.447 ± 0.003	0.440 ± 0.003	0.463 ± 0.002	339
Fts MDLP[35]	$\mathbf{0.477 \pm 0.003}$	$\mathbf{0.462 \pm 0.003}$	$\mathbf{0.458 \pm 0.003}$	$\mathbf{0.477 \pm 0.003}$	1074

Regresión logística dentro de la selección de características y reducción de dimensionalidad dan un diferente panorama para que sea un modelo a usar, ya que reduce su tiempo de ejecución y mejora los resultados vistos en las Tablas (4.52), (4.53) y (4.54).

K Vecinos Más Cercanos

Para *All*, KNN obtuvo muy buenos resultados en tiempo de ejecución y las métricas de evaluación usando la selección de características por parte de [22], disminuyó drásticamente el tiempo de ejecución comparado con (4.25), se observan los resultados en la Tabla (4.55).

Tabla 4.55: Evaluación de All con KNN para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Redes Bayesianas [22]	$\mathbf{0.519 \pm 0.001}$	$\mathbf{0.520 \pm 0.001}$	$\mathbf{0.519 \pm 0.001}$	$\mathbf{0.519 \pm 0.001}$	360	20
PCA	0.432 ± 0.001	0.428 ± 0.001	0.428 ± 0.001	0.432 ± 0.001	2937	20
Fts MDLP [47]	0.445 ± 0.001	0.442 ± 0.001	0.442 ± 0.001	0.445 ± 0.001	3261	20
Fts MDLP [35]	0.490 ± 0.002	0.490 ± 0.002	0.490 ± 0.002	0.489 ± 0.002	30640	20

Para *Top* se halló que los mejores resultados tanto en el rendimiento de las métricas de evaluación y el tiempo de ejecución fue la selección de características con MDL-FS, esto se refleja en la Tabla (4.56).

Tabla 4.56: Evaluación de Top con KNN para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Redes Bayesianas [22]	**0.434 ± 0.003**	**0.433 ± 0.003**	**0.433 ± 0.004**	**0.434 ± 0.003**	**90**	16
PCA	0.373 ± 0.004	0.371 ± 0.003	0.370 ± 0.003	0.373 ± 0.003	**90**	13
Fts MDLP [47]	0.399 ± 0.003	0.397 ± 0.003	0.396 ± 0.003	0.399 ± 0.003	418	19
Fts MDLP [35]	0.424 ± 0.002	0.423 ± 0.002	0.421 ± 0.002	0.424 ± 0.002	765	20

Para *Bot* los mejores resultados en las métricas de evaluación fueron obtenidos con la selección de características proporcionadas por [35], sin embargo, los resultados por [22] fueron muy similares a los de [35], pero con un tiempo de ejecución mucho menor, siendo la selección más óptima en esta relación, lo anterior se refleja en la Tabla (4.57).

Tabla 4.57: Evaluación de Bot con KNN para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Redes Bayesianas [22]	0.474 ± 0.002	0.471 ± 0.002	0.468 ± 0.002	0.474 ± 0.002	**56**	20
PCA	0.456 ± 0.002	0.448 ± 0.002	0.447 ± 0.002	0.456 ± 0.002	88	20
Fts MDLP [47]	0.459 ± 0.002	0.452 ± 0.002	0.451 ± 0.002	0.459 ± 0.001	200	20
Fts MDLP [35]	**0.477 ± 0.003**	**0.474 ± 0.003**	**0.471 ± 0.003**	**0.477 ± 0.003**	803	20

K vecinos más cercanos mejoró exhaustivamente los resultados en la selección de características y reducción de dimensionalidad comparando a las bases de datos originales o que simplemente están discretizados, siendo ya un modelo a considerar porque se equilibró su tiempo de ejecución con el rendimiento de las métricas de evaluación.

Redes Neuronales

Para *All*, Redes Neuronales muestra ser un buen modelo para esta clasificación, esto se aprecia en la Tabla (4.58), donde los valores de las métricas de evaluación son muy similares entre la selección de características y reducción de dimensionalidad, lo mismo con el tiempo ejecución.

Tabla 4.58: Evaluación de All con RN para Seleccionar/Reducir Características con 3 Clases

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Redes Bayesianas [22]	0.453 ± 0.002	0.446 ± 0.005	0.441 ± 0.009	0.453 ± 0.002	4034	5
PCA	0.452 ± 0.004	0.448 ± 0.004	0.448 ± 0.004	0.452 ± 0.001	**2754**	3
Fts MDLP [47]	0.453 ± 0.002	0.449 ± 0.006	0.443 ± 0.008	0.453 ± 0.002	3500	4
Fts MDLP [35]	**0.460 ± 0.002**	**0.455 ± 0.004**	**0.449 ± 0.011**	**0.460 ± 0.002**	3662	4

Para *Top* los resultados fueron muy similares entre todos al igual que el tiempo de ejecución, donde [35], fue el que obtuvo los rendimientos más altos en las métricas, usando 3 capas ocultas. Lo anterior se ve reflejado en la Tabla (4.59).

Tabla 4.59: Evaluación de Top con RN para Seleccionar/Reducir Características con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Redes Bayesianas [22]	0.410 ± 0.005	0.428 ± 0.017	0.359 ± 0.025	0.410 ± 0.005	**447**	2
PCA	0.393 ± 0.003	0.388 ± 0.005	0.376 ± 0.008	0.393 ± 0.002	463	1
Fts MDLP [47]	0.409 ± 0.004	0.407 ± 0.004	0.391 ± 0.011	0.409 ± 0.003	461	2
Fts MDLP [35]	$\mathbf{0.417 \pm 0.004}$	$\mathbf{0.422 \pm 0.011}$	$\mathbf{0.394 \pm 0.018}$	$\mathbf{0.417 \pm 0.004}$	516	4

Para *Bot*, de nueva cuenta, los resultados fueron muy cerrados en las selecciones de características y reducción de dimensionalidad. La selección de características propuesta por [35] fue la que dio los resultados más óptimos en las métricas. En la tabla (4.60) se aprecia la similitud entre los valores obtenidos de cada caso.

Tabla 4.60: Evaluación de Bot con RN para Seleccionar/Reducir Características con 3 Clases

Conjunto	**Accuracy**	**Precision**	**F1-Score**	**Recall**	**E-T (S)**	**K**
Redes Bayesianas [22]	0.475 ± 0.003	0.466 ± 0.005	0.445 ± 0.019	0.475 ± 0.002	**397**	1
PCA	0.472 ± 0.002	0.461 ± 0.004	0.455 ± 0.009	0.472 ± 0.002	406	1
Fts MDLP [47]	0.474 ± 0.004	0.468 ± 0.005	0.454 ± 0.014	0.474 ± 0.003	694	4
Fts MDLP [35]	$\mathbf{0.483 \pm 0.003}$	$\mathbf{0.478 \pm 0.004}$	$\mathbf{0.461 \pm 0.015}$	$\mathbf{0.483 \pm 0.002}$	534	4

El uso de redes neuronales para esta clasificación dio buenos resultados en las métricas para los conjuntos de *All, Top y Bot*, sin embargo, su búsqueda de los hiperparámetros óptimos y su tiempo de ejecución siguen siendo un gran factor para no usar este modelo como el principal para la clasificación.

4.5. Pruebas estadísticas de los métodos

Para analizar el efecto de las discretizaciones sobre el desempeño de los modelos de aprendizaje supervisado se aplicó una prueba de normalidad de Shapiro-Wilk y la prueba de homogeneidad de varianzas de Levene. Dependiendo de los supuestos se procedió a hacer pruebas paramétricas o no paramétricas. Para el caso de la prueba paramétrica, se aplica ANOVA de medidas repetidas y prueba t pareada con corrección Holm. En el caso de la prueba no paramétrica se aplica Friedman y prueba de Wilcoxon con corrección Holm. Lo anterior únicamente se aplicó para los resultados obtenidos con 3 clases en la categorización de la variable objetivo ya que, con ese número de clases se hallaron los resultados más elevados.

Naive Bayes

Para *All*, los contrastes de Shapiro–Wilk y Levene mostraron que los supuestos paramétricos se cumplen en *f1–score*, *precision* y *recall*, pero no en *accuracy*. En consecuencia, se aplicó un **ANOVA de medidas repetidas** para las tres primeras métricas y la **prueba de Friedman** para *accuracy*. En *accuracy* se obtuvo $Q = 30.00$ con $p < 0.0001$; en las demás métricas los resultados también fueron altamente significativos, por ejemplo en *precision* se registró $F(3,27) = 1424.32$, $p < 0.0001$, $\eta^2_G = 0.9893$ y $\varepsilon = 0.7701$, lo que indica que la discretización explica cerca del 99 % de la varianza del rendimiento.

Las comparaciones post-hoc se realizaron con corrección de Holm: prueba de Wilcoxon para *accuracy* y pruebas *t* pareadas para el resto. La Tabla (4.61) resume los resultados; para *accuracy* se reporta el estadístico Z de Wilcoxon y su tamaño del efecto $r = Z/\sqrt{N}$.

Tabla 4.61: Comparaciones pareadas entre discretizaciones para NB All en todas las métricas

A	B	Accuracy†			F1-Score		Precision		Recall	
		Z	*r*	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-2.80	-0.89	0.0117	-14.29	$<10^{-4}$	-7.90	$<10^{-4}$	-31.30	$<10^{-4}$
K Means-CH	MDLP	-2.80	-0.89	0.0117	-38.31	$<10^{-4}$	-55.83	$<10^{-4}$	-35.68	$<10^{-4}$
K Means-CH	Ori	-2.80	-0.89	0.0117	31.28	$<10^{-4}$	32.15	$<10^{-4}$	11.38	$<10^{-4}$
K Means-DB	MDLP	-2.80	-0.89	0.0117	-8.26	$<10^{-4}$	-21.76	$<10^{-4}$	7.69	$<10^{-4}$
K Means-DB	Ori	-2.80	-0.89	0.0117	37.84	$<10^{-4}$	34.81	$<10^{-4}$	36.52	$<10^{-4}$
MDLP	Ori	-2.80	-0.89	0.0117	52.47	$<10^{-4}$	59.93	$<10^{-4}$	31.86	$<10^{-4}$

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{N}$.

Para *Top*, se cumplieron los supuestos de normalidad y homocedasticidad en todas las métricas, por lo que se aplicó una prueba paramétrica. En particular, la prueba de Shapiro-Wilk arrojó valores $p > 0.05$ para todas las métricas, y la prueba de Levene también fue no significativa ($p > 0.05$), lo que indica que se cumple la homogeneidad de varianzas.

Se aplicó una prueba **ANOVA de medidas repetidas** que mostró diferencias estadísticamente significativas entre las discretizaciones en todas las métricas. Por ejemplo, para *precision* se obtuvo un valor de $F(3,27) = 698.94$, $p < 0.0001$, con un tamaño del efecto generalizado

$\eta^2_G = 0.953$ y corrección por esfericidad $\varepsilon = 0.883$. Resultados similares se obtuvieron para las métricas de *accuracy*, *f1-score* y *recall*.

Las comparaciones post-hoc se realizaron mediante pruebas *t* **pareadas con corrección de Holm**, cuyos resultados se muestran en la Tabla (4.62).

Tabla 4.62: Comparaciones pareadas entre discretizaciones para NB Top en todas las métricas

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-3.6354	0.0054	-2.2343	0.0523	-2.0275	0.0732	-3.6309	0.0055
K Means-CH	MDLP	-16.4361	<0.0001	-17.3682	<0.0001	-15.7645	<0.0001	-16.2972	<0.0001
K Means-CH	Ori	31.5885	<0.0001	16.4484	<0.0001	30.0956	<0.0001	36.4250	<0.0001
K Means-DB	MDLP	-10.4240	<0.0001	-13.4567	<0.0001	-12.9427	<0.0001	-10.4774	<0.0001
K Means-DB	Ori	36.3143	<0.0001	21.1792	<0.0001	40.1865	<0.0001	41.8871	<0.0001
MDLP	Ori	39.4367	<0.0001	27.4480	<0.0001	40.8297	<0.0001	43.8593	<0.0001

Para *Bot*, *Naive Bayes* presentó supuestos mixtos. La normalidad y homocedasticidad se violaron en *accuracy*, *f1-score* y *recall*, de modo que se empleó la **prueba de Friedman**. Los estadísticos fueron $Q_{\text{accuracy}} = 26.91$, $Q_{\text{f1-score}} = 27.84$, $Q_{\text{recall}} = 26.04$, todos con $p < 0.0001$. En *precision* sí se cumplieron los supuestos, aplicándose un **ANOVA de medidas repetidas** con $F(3,27) = 41.18$, $p < 0.0001$, $\eta^2_G = 0.587$ y $\varepsilon = 0.722$, lo que indica que la discretización explica casi el 59 % de la varianza en dicha métrica.

Las comparaciones post-hoc se corrigieron con Holm: prueba de Wilcoxon para las métricas no paramétricas, y prueba *t* pareada para *precision*. Los resultados se muestran en la Tabla (4.63).

Tabla 4.63: Comparaciones pareadas entre discretizaciones para NB Bot en todas las métricas

A	B	Accuracy†			F1-Score†			Precision‡		Recall†		
		Z	*r*	*p*	*Z*	*r*	*p*	*t*	*p*	*Z*	*r*	*p*
K Means-CH	K Means-DB	-2.803	-0.89	0.0117	-2.701	-0.85	0.0117	-3.868	0.0114	-2.803	-0.89	0.0117
K Means-CH	MDLP	-2.803	-0.89	0.0273	-2.803	-0.89	0.0117	2.189	0.0564	-2.803	-0.89	0.0117
K Means-CH	Ori	-2.803	-0.89	0.0273	-2.803	-0.89	0.0117	8.897	$<10^{-4}$	-2.701	-0.85	0.0117
K Means-DB	MDLP	-2.803	-0.89	0.0117	-2.803	-0.89	0.0117	3.798	0.0114	-2.803	-0.89	0.0117
K Means-DB	Ori	-2.803	-0.89	0.0117	-2.803	-0.89	0.0117	9.161	$<10^{-4}$	-2.803	-0.89	0.0117
MDLP	Ori	-1.478	-0.47	0.1602	-2.192	-0.69	0.0273	5.576	0.0014	-1.631	-0.52	0.1309

† Wilcoxon pareado ($N = 10$) con corrección de Holm; $r = Z/\sqrt{N}$.

‡ Prueba *t* pareada; se muestran *t* y el valor-*p* corregido.

CART

Para *All* se cumplieron los supuestos para realizar una prueba paramétrica: los valores *p* para la prueba de Shapiro-Wilk fueron mayores a 0.05 en todas las métricas, y la prueba de Levene

también arrojó un valor $p > 0.05$, lo que indica homocedasticidad entre grupos.

El análisis mediante **ANOVA de medidas repetidas** reveló diferencias estadísticamente significativas en todas las métricas. Por ejemplo, en la métrica de *accuracy* se obtuvo $F(3,27) = 260.28$, $p < 0.0001$, con un tamaño del efecto generalizado $\eta^2_G = 0.9511$ y un valor de esfericidad $\varepsilon = 0.5822$. Debido a que la esfericidad fue violada, se aplicó la corrección de Greenhouse-Geisser. Resultados similares se obtuvieron para las métricas de *f1-score*, *precision* y *recall*.

Las comparaciones post-hoc se realizaron mediante pruebas *t* pareadas con corrección de Holm, cuyos resultados se resumen en la Tabla (4.64).

Tabla 4.64: Comparaciones pareadas entre discretizaciones para CART All en todas las métricas

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-1.62	0.2779	-5.13	0.0006	-5.14	0.0006	-5.07	0.0007
K Means-CH	MDLP	-14.89	<0.0001	-56.60	<0.0001	-57.32	<0.0001	-56.27	<0.0001
K Means-CH	Ori	-31.72	<0.0001	-64.39	<0.0001	-65.81	<0.0001	-63.23	<0.0001
K Means-DB	MDLP	-15.25	<0.0001	-51.41	<0.0001	-51.86	<0.0001	-49.80	<0.0001
K Means-DB	Ori	-29.88	<0.0001	-53.74	<0.0001	-54.76	<0.0001	-51.87	<0.0001
MDLP	Ori	-0.95	0.3672	8.88	<0.0001	9.41	<0.0001	8.97	<0.0001

Para el conjunto *Top*, se cumplieron los supuestos de normalidad y homocedasticidad, lo que permitió aplicar **ANOVA de medidas repetidas**. En todas las métricas evaluadas se observaron diferencias estadísticamente significativas. En el caso de *accuracy*, el resultado fue $F(3,27) = 260.28$, $p < 0.0001$, con un tamaño del efecto generalizado $\eta^2_G = 0.9511$ y una corrección de esfericidad $\varepsilon = 0.5822$. Resultados similares se observaron para *f1-score*, *precision* y *recall*.

En la Tabla (4.65) se presentan las comparaciones pareadas con prueba *t* y corrección de Holm. Se observa que tanto *K Means-CH* como *K Means-DB* obtienen resultados significativamente inferiores en comparación con las discretizaciones MDLP y Original, mientras que no se encuentran diferencias significativas entre estas últimas.

Tabla 4.65: Comparaciones pareadas entre discretizaciones para CART Top en todas las métricas

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-1.62	0.278	-1.64	0.273	-1.63	0.277	-1.68	0.256
K Means-DB	MDLP	-14.89	<0.0001	-14.94	<0.0001	-14.94	<0.0001	-14.94	<0.0001
K Means-CH	Ori	-31.72	<0.0001	-31.93	<0.0001	-31.90	<0.0001	-31.83	<0.0001
K Means-DB	MDLP	-15.25	<0.0001	-15.33	<0.0001	-15.31	<0.0001	-15.15	<0.0001
K Means-DB	Ori	-29.88	<0.0001	-28.81	<0.0001	-28.81	<0.0001	-29.23	<0.0001
MDLP	Ori	-0.95	0.367	-0.96	0.362	-0.95	0.368	-0.96	0.362

Para el conjunto *Bot*, se cumplieron los supuestos de normalidad y homocedasticidad, por

lo que se aplicó **ANOVA de medidas repetidas** como prueba global. Los resultados mostraron diferencias estadísticamente significativas entre las discretizaciones, con un valor $F(3,27) = 113.53$, $p < 0.0001$, $\eta^2_G = 0.9077$ y $\varepsilon = 0.5764$ para la métrica *accuracy*. Esto indica que aproximadamente el 91 % de la variabilidad del desempeño se debe a la discretización utilizada. Resultados consistentes se observaron también para las métricas *precision*, *recall* y *f1-score*.

En la Tabla (4.66) se muestran las comparaciones pareadas entre discretizaciones para CART Bot en todas las métricas.

Tabla 4.66: Comparaciones pareadas entre discretizaciones para CART Bot en todas las métricas

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-1.57	0.3041	-1.50	0.3360	-1.43	0.2793	-1.58	0.2986
K Means-CH	MDLP	-9.17	0.0000	-9.13	0.0000	-9.06	0.0000	-9.19	0.0000
K Means-CH	Ori	-18.14	0.0000	-17.88	0.0000	-17.54	0.0000	-18.20	0.0000
K Means-DB	MDLP	-13.25	0.0000	-13.29	0.0000	-13.22	0.0000	-13.27	0.0000
K Means-DB	Ori	-17.99	0.0000	-17.84	0.0000	-17.59	0.0000	-18.02	0.0000
MDLP	Ori	-0.60	0.5634	-0.58	0.5765	-0.56	0.5660	-0.59	0.5667

Regresión Logística

Para *All*, los contrastes de Shapiro–Wilk y Levene mostraron que los supuestos paramétricos se cumplen en *accuracy*, *precision* y *recall*, pero no en *f1–score*. En consecuencia, se aplicó un **ANOVA de medidas repetidas** a las tres primeras métricas y la **prueba de Friedman** a *f1–score*. En *f1–score* se obtuvo $Q = 30.00$ con $p < 0.0001$ en las métricas paramétricas los resultados también fueron altamente significativos, por ejemplo en *precision* se registró $F(3,27) = 4351.94$, $p < 0.0001$, $\eta^2_G = 0.9962$ y $\varepsilon = 0.8424$, lo que indica que la discretización explica más del 99 % de la varianza del rendimiento.

Las comparaciones post-hoc se corrigieron con Holm: prueba de Wilcoxon para *f1–score* y pruebas *t* pareadas para las demás métricas. La Tabla (4.67) resume los resultados.

Tabla 4.67: Comparaciones pareadas entre discretizaciones para RL All en todas las métricas

A	B	Accuracy		F1-Score†			Precision		Recall	
		t	*p*	*Z*	*r*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-15.4041	0.0000	-2.803	-0.886	0.0117	-15.1595	0.0000	-15.5720	0.0000
K Means-CH	MDLP	-123.1648	0.0000	-2.803	-0.886	0.0117	-98.8689	0.0000	-120.1500	0.0000
K Means-CH	Ori	-85.6221	0.0000	-2.803	-0.886	0.0117	-74.9719	0.0000	-84.9402	0.0000
K Means-DB	MDLP	-85.4451	0.0000	-2.803	-0.886	0.0117	-73.1352	0.0000	-85.4662	0.0000
K Means-DB	Ori	-68.2712	0.0000	-2.803	-0.886	0.0117	-58.5507	0.0000	-66.9797	0.0000
MDLP	Ori	22.5604	0.0000	-2.803	-0.886	0.0117	23.2605	0.0000	21.8579	0.0000

† Wilcoxon pareado ($N = 10$); $r = Z/\sqrt{N}$, corrección de Holm.

Para *Top*, las pruebas de Shapiro–Wilk y Levene confirmaron normalidad y homocedasticidad en las cuatro métricas, de modo que se aplicó **ANOVA de medidas repetidas** a cada una de ellas. Los resultados fueron altamente significativos; por ejemplo, en *accuracy* se obtuvo $F(3,27) = 576.63$, $p < 0.0001$, $\eta^2_G = 0.9733$ y $\varepsilon = 0.7417$, indicando que la discretización explica más del 97 % de la varianza. Valores similares se observaron en *precision*, *f1–score* y *recall*, respectivamente.

Las comparaciones post-hoc se realizaron con pruebas *t* pareadas y corrección de Holm. La Tabla (4.68) muestra el estadístico *t* y el valor-*p* corregido para cada par de discretizaciones en las cuatro métricas.

Tabla 4.68: Comparaciones pareadas entre discretizaciones para RL All en todas las métricas

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-6.051	0.0004	-4.125	0.0026	-5.714	0.0006	-6.022	0.0004
K Means-CH	MDLP	-33.722	$< 10^{-4}$	-33.674	$< 10^{-4}$	-31.975	$< 10^{-4}$	-33.630	$< 10^{-4}$
K Means-CH	Ori	-25.636	$< 10^{-4}$	-21.641	$< 10^{-4}$	-23.609	$< 10^{-4}$	-26.254	$< 10^{-4}$
K Means-DB	MDLP	-29.307	$< 10^{-4}$	-28.266	$< 10^{-4}$	-24.582	$< 10^{-4}$	-29.156	$< 10^{-4}$
K Means-DB	Ori	-22.064	$< 10^{-4}$	-18.399	$< 10^{-4}$	-18.808	$< 10^{-4}$	-22.397	$< 10^{-4}$
MDLP	Ori	3.637	0.0054	10.663	$< 10^{-4}$	2.480	0.0350	3.809	0.0042

Para *Bot*, los contrastes de Shapiro–Wilk y Levene indicaron que se cumplen los supuestos paramétricos en todas las métricas evaluadas. Por lo tanto, se aplicó un **ANOVA de medidas repetidas** en *accuracy*, *f1–score*, *precision* y *recall*. En todos los casos se observaron diferencias estadísticamente significativas. Por ejemplo, en *accuracy* se obtuvo $F(3,27) = 2293.11$, $p < 0.0001$, con $\eta^2_G = 0.9798$ y $\varepsilon = 0.7407$, lo cual indica que el tipo de discretización explica cerca del 98 % de la varianza en el rendimiento.

Las comparaciones post-hoc se realizaron mediante pruebas *t* pareadas con corrección de Holm. La Tabla (4.69) resume los resultados en todas las métricas.

Tabla 4.69: Comparaciones pareadas entre discretizaciones para RL Bot en todas las métricas

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-19.55	$<!10^{-4}$	-22.39	$<!10^{-4}$	-16.86	$<!10^{-4}$	-19.20	$<!10^{-4}$
K Means-CH	MDLP	-68.81	$<!10^{-4}$	-64.26	$<!10^{-4}$	-55.35	$<!10^{-4}$	-68.32	$<!10^{-4}$
K Means-CH	Ori	-79.60	$<!10^{-4}$	-67.38	$<!10^{-4}$	-49.73	$<!10^{-4}$	-77.44	$<!10^{-4}$
K Means-DB	MDLP	-46.00	$<!10^{-4}$	-37.38	$<!10^{-4}$	-35.97	$<!10^{-4}$	-45.55	$<!10^{-4}$
K Means-DB	Ori	-37.71	$<!10^{-4}$	-35.88	$<!10^{-4}$	-32.36	$<!10^{-4}$	-37.54	$<!10^{-4}$
MDLP	Ori	10.33	$<!10^{-4}$	8.19	$<!10^{-4}$	9.07	$<!10^{-4}$	10.04	$<!10^{-4}$

K Vecinos más Cercanos

Para *All*, los contrastes de Shapiro-Wilk y Levene mostraron que los supuestos paramétricos se cumplen en *f1-score*, *precision* y *recall*, pero no en *accuracy*. En consecuencia, se aplicó un **ANOVA de medidas repetidas** a las tres primeras métricas y la **prueba de Friedman** a *accuracy*. En *accuracy* se obtuvo $Q = 30.00$ con $p < 0.0001$; en las demás métricas los resultados también fueron altamente significativos, por ejemplo en *f1–score* se registró $F(3,27) = 51496.39$, $p < 0.0001$, $\eta^2_G = 0.9996$, $\varepsilon = 0.8806$, indicando que la discretización explica prácticamente toda la varianza del rendimiento.

Las comparaciones post-hoc se corrigieron con Holm: prueba de Wilcoxon para *accuracy* y pruebas *t* pareadas para el resto. La Tabla (4.70) resume los resultados; para *accuracy* se reporta el estadístico Z de Wilcoxon y su tamaño del efecto $r = Z/\sqrt{N}$.

Tabla 4.70: Comparaciones pareadas entre discretizaciones para KNN All en todas las métricas

A	B	Accuracy†			F1-Score		Precision		Recall	
		Z	*r*	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-2.803	-0.886	0.0117	65.587	$<10^{-4}$	61.148	$<10^{-4}$	65.707	$<10^{-4}$
K Means-CH	MDLP	-2.803	-0.886	0.0117	-255.766	$<10^{-4}$	-242.491	$<10^{-4}$	-227.001	$<10^{-4}$
K Means-CH	Ori	-2.803	-0.886	0.0117	-66.875	$<10^{-4}$	-67.157	$<10^{-4}$	-69.762	$<10^{-4}$
K Means-DB	MDLP	-2.803	-0.886	0.0117	-418.779	$<10^{-4}$	-373.402	$<10^{-4}$	-414.035	$<10^{-4}$
K Means-DB	Ori	-2.803	-0.886	0.0117	-179.731	$<10^{-4}$	-158.237	$<10^{-4}$	-201.332	$<10^{-4}$
MDLP	Ori	-2.803	-0.886	0.0117	245.897	$<10^{-4}$	227.544	$<10^{-4}$	249.797	$<10^{-4}$

† Wilcoxon pareado ($N = 10$) con corrección de Holm; $r = Z/\sqrt{N}$.

Para *Top*, los contrastes de Shapiro–Wilk y Levene confirmaron que los supuestos paramétricos se cumplen en todas las métricas, por lo que se utilizó **ANOVA de medidas repetidas** para cada una. Los resultados fueron altamente significativos en las cuatro métricas, por ejemplo, en *f1–score* se obtuvo $F(3,27) = 1360.28$, $p < 0.0001$, $\eta^2_G = 0.9867$ y $\varepsilon = 0.6184$, indicando que la discretización explica cerca del 99 % de la varianza observada.

Las comparaciones post-hoc se realizaron con pruebas *t* pareadas y corrección de Holm. La Tabla (4.71) resume los resultados obtenidos para todas las métricas.

Tabla 4.71: Comparaciones pareadas entre discretizaciones para KNN Top en todas las métricas

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	18.4305	$<!10^{-4}$	17.7846	$<!10^{-4}$	16.8986	$<!10^{-4}$	18.5442	$<!10^{-4}$
K Means-CH	MDLP	-70.7415	$<!10^{-4}$	-67.4679	$<!10^{-4}$	-63.2850	$<!10^{-4}$	-69.2890	$<!10^{-4}$
K Means-CH	Ori	-17.9202	$<!10^{-4}$	-16.6258	$<!10^{-4}$	-14.6272	$<!10^{-4}$	-16.5362	$<!10^{-4}$
K Means-DB	MDLP	-52.3209	$<!10^{-4}$	-51.2804	$<!10^{-4}$	-48.7576	$<!10^{-4}$	-52.4976	$<!10^{-4}$
K Means-DB	Ori	-23.1828	$<!10^{-4}$	-21.5298	$<!10^{-4}$	-20.4188	$<!10^{-4}$	-23.1853	$<!10^{-4}$
MDLP	Ori	35.3709	$<!10^{-4}$	37.3968	$<!10^{-4}$	37.8106	$<!10^{-4}$	34.9026	$<!10^{-4}$

Para *Bot*, los contrastes de Shapiro–Wilk y Levene confirmaron que los supuestos paramétricos se cumplen en las cuatro métricas; por tanto, se aplicó **ANOVA de medidas repetidas**. Todos los análisis resultaron altamente significativos: por ejemplo, en *recall* $F = 1328.39$, $\eta^2_G = 0.9846$, $\varepsilon = 0.7626$. Estos valores indican que la discretización explica más del 98 % de la varianza en todas las métricas.

Las comparaciones post-hoc se realizaron mediante pruebas *t* pareadas con corrección de Holm. La Tabla (4.72) resume los resultados en cada métrica.

Tabla 4.72: Comparaciones pareadas entre discretizaciones para KNN Bot en todas las métricas

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	28.32	$< 10^{-4}$	25.06	$< 10^{-4}$	24.90	$< 10^{-4}$	29.03	$< 10^{-4}$
K Means-CH	MDLP	-40.41	$< 10^{-4}$	-41.99	$< 10^{-4}$	-39.09	$< 10^{-4}$	-41.20	$< 10^{-4}$
K Means-CH	Ori	-21.65	$< 10^{-4}$	-19.25	$< 10^{-4}$	-17.00	$< 10^{-4}$	-21.96	$< 10^{-4}$
K Means-DB	MDLP	-57.72	$< 10^{-4}$	-57.24	$< 10^{-4}$	-56.16	$< 10^{-4}$	-58.56	$< 10^{-4}$
K Means-DB	Ori	-34.14	$< 10^{-4}$	-29.74	$< 10^{-4}$	-28.09	$< 10^{-4}$	-34.93	$< 10^{-4}$
MDLP	Ori	12.67	$< 10^{-4}$	16.64	$< 10^{-4}$	16.15	$< 10^{-4}$	13.04	$< 10^{-4}$

Redes Neuronales

Para *All*, el cumplimiento de los supuestos paramétricos fue mixto. Las pruebas de Shapiro-Wilk y Levene confirmaron que en *precision* se cumplen los supuestos, por lo que se utilizó un **ANOVA de medidas repetidas** obteniendo ($F = 146.15$, $p < 0.0001$, $\eta^2_G = 0.817$, $\varepsilon = 0.6932$). En *recall*, *accuracy* y *f1-score* se violaron los supuestos, por lo que se empleó la **prueba de Friedman**, para *recall* se obtuvo ($Q = 27.12$, $p < 0.0001$), mientras que para las otras se obtuvieron resultados similares.

Las comparaciones post-hoc se realizaron con corrección de Holm: Wilcoxon para métricas no paramétricas y *t* pareadas para *precision*. La Tabla (4.73) resume los resultados; para **Friedman** se reporta el estadístico Z y tamaño del efecto $r = Z/\sqrt{N}$.

Tabla 4.73: Comparaciones pareadas entre discretizaciones para NN All en todas las métricas

A	B	**Accuracy**†			**F1-Score**†			**Precision**‡		**Recall**†		
		Z	*r*	*p*	*Z*	*r*	*p*	*t*	*p*	*Z*	*r*	*p*
K Means-CH	K Means-DB	-2.80	-0.89	0.012	-2.80	-0.89	0.012	-11.27	$< 10^{-4}$	-2.80	-0.89	0.012
K Means-CH	MDLP	-2.80	-0.89	0.012	-2.80	-0.89	0.012	-15.98	$< 10^{-4}$	-2.80	-0.89	0.012
K Means-CH	Ori	-2.80	-0.89	0.012	-2.80	-0.89	0.012	-24.13	$< 10^{-4}$	-2.80	-0.89	0.012
K Means-DB	MDLP	-2.80	-0.89	0.012	-2.80	-0.89	0.012	-9.16	$< 10^{-4}$	-2.80	-0.89	0.012
K Means-DB	Ori	-2.80	-0.89	0.012	-2.80	-0.89	0.012	-8.68	$< 10^{-4}$	-2.80	-0.89	0.012
MDLP	Ori	-0.86	-0.274	0.432	-0.255	-0.89	0.846	1.74	0.1163	-0.87	-0.27	0.432

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{N}$.

‡ Prueba *t* pareada con corrección de Holm.

Para *Top*, el cumplimiento de los supuestos paramétricos fue mixto. Las pruebas de Shapiro-Wilk y Levene mostraron que *accuracy* y *recall* cumplen normalidad / homocedasticidad, de modo que se aplicó **ANOVA de medidas repetidas**.

En cambio, los supuestos se violaron en *f1-score* y *precision*, por lo que se utilizó la **prueba de Friedman**.

Las comparaciones post-hoc se corrigieron con Holm: pruebas *t* pareadas para las métricas paramétricas y Wilcoxon para las no paramétricas. La Tabla (4.74) recoge los resultados.

Tabla 4.74: Comparaciones pareadas entre discretizaciones para NN Top en todas las métricas

A	B	**Accuracy**‡		**F1-Score**†			**Precision**†			**Recall**‡	
		t	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*t*	*p*
K Means-CH	K Means-DB	-3.164	0.0230	-0.255	-0.081	0.8457	-0.561	-0.177	1.0000	-3.242	0.0202
K Means-CH	MDLP	-34.379	$< 10^{-4}$	-1.478	-0.467	0.6406	-1.784	-0.564	0.3359	-34.941	$< 10^{-4}$
K Means-CH	Ori	-17.525	$< 10^{-4}$	-2.599	-0.822	0.0293	-2.803	-0.886	0.0117	-17.183	$< 10^{-4}$
K Means-DB	MDLP	-21.728	$< 10^{-4}$	-1.274	-0.403	0.6973	-1.784	-0.564	0.3359	-23.006	$< 10^{-4}$
K Means-DB	Ori	-18.406	$< 10^{-4}$	-2.701	-0.854	0.0234	-2.803	-0.886	0.0117	-18.852	$< 10^{-4}$
MDLP	Ori	1.460	0.1782	-1.070	-0.339	0.6973	-0.968	-0.306	0.7500	1.568	0.1513

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{N}$.

‡ Prueba *t* pareada con corrección de Holm.

Para *Bot*, los contrastes de Shapiro–Wilk y Levene indicaron incumplimiento de los supuestos paramétricos sólo en *accuracy*; en las demás métricas los supuestos se cumplieron.

Para *Accuracy* se aplicó la **prueba de Friedman** obteniendo $Q = 27.00$, $p < 0.0001$. Mientras que para las otras métricas se aplicó la **ANOVA de medidas repetidas**.

Las comparaciones post-hoc se corrigieron con Holm: Wilcoxon para *accuracy*; *t* pareada y *p* para las métricas paramétricas. La Tabla (4.75) resume los resultados.

Tabla 4.75: Comparaciones pareadas entre discretizaciones para NN Bot en todas las métricas

A	B	Accuracy†			F1-Score		Precision		Recall	
		Z	*r*	*p*	*t*	*p*	*t*	*p*	*t*	*p*
K Means-CH	K Means-DB	-2.803	-0.886	0.0117	-3.407	0.0156	-4.073	0.0056	-9.540	$< 10^{-4}$
K Means-CH	MDLP	-2.803	-0.886	0.0117	-12.252	$< 10^{-4}$	-15.332	$< 10^{-4}$	-25.427	$< 10^{-4}$
K Means-CH	Ori	-2.803	-0.886	0.0117	-11.960	$< 10^{-4}$	-21.454	$< 10^{-4}$	-35.743	$< 10^{-4}$
K Means-DB	MDLP	-2.803	-0.886	0.0117	-11.432	$< 10^{-4}$	-13.742	$< 10^{-4}$	-16.502	$< 10^{-4}$
K Means-DB	Ori	-2.803	-0.886	0.0117	-12.426	$< 10^{-4}$	-19.170	$< 10^{-4}$	-20.829	$< 10^{-4}$
MDLP	Ori	-0.153	-0.048	0.9219	1.922	0.0867	2.121	0.0629	0.125	0.9033

† Wilcoxon pareado con corrección de Holm; $r = Z/\sqrt{10}$.

4.6. Pruebas estadísticas de los modelos

Para analizar el efecto de los modelos sobre el desempeño de su discretización se aplicó una prueba de normalidad de Shapiro-Wilk y la prueba de homogeneidad de varianzas de Levene. Dependiendo de los supuestos se procedió a hacer pruebas paramétricas o no paramétricas. Para el caso de la prueba paramétrica, se aplica ANOVA de medidas repetidas y prueba t pareada con corrección Holm. En el caso de la prueba no paramétrica se aplica Friedman y prueba de Wilcoxon con corrección Holm. Lo anterior únicamente se aplicó para los resultados obtenidos con 3 clases en la categorización de la variable objetivo ya que, con ese número se hallaron los resultados más elevados.

Base de datos original

Para *All*, los contrastes de Shapiro-Wilk y Levene mostraron que los supuestos paramétricos se cumplen solamente en la métrica de *recall*; en *accuracy*, *f1-score* y *precision* dichos supuestos se violaron.

La métrica *recall* se le aplicó un **ANOVA de medidas repetidas**, $F(4,36) = 4227.80$, $p < 0.0001$, $\eta^2_G = 0.9966$, $\varepsilon = 0.5143$, lo que indica que el tipo de modelo explica más del 99 % de la varianza. Mientras que para las otras métricas se utilizó la **prueba de Friedman** obteniendo $Q = 40.0$, $p < 0.0001$ en las tres métricas.

Las comparaciones post-hoc se corrigieron con Holm. La Tabla (4.76) resume los resultados.

Tabla 4.76: Comparaciones pareadas de modelos con la base de datos original y All

A	B	Accuracy†			F1-Score†			Precision†			Recall	
		Z	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*t*	*p*
CART	KNN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	-108.00	$< 10^{-4}$
CART	LR	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	-173.30	$< 10^{-4}$
CART	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	-51.56	$< 10^{-4}$
CART	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	-90.58	$< 10^{-4}$
KNN	LR	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	-28.67	$< 10^{-4}$
KNN	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	62.94	$< 10^{-4}$
KNN	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	-37.20	$< 10^{-4}$
LR	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	57.36	$< 10^{-4}$
LR	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	-11.94	$< 10^{-4}$
NB	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.80	-0.886	0.0195	-51.43	$< 10^{-4}$

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.

Para el subconjunto **Top** derivado de la base de datos original, los contrastes de Shapiro–Wilk y Levene mostraron que los supuestos paramétricos se cumplen en las métricas *accuracy*, *precision* y *recall*, pero se violan en *f1-score*. Por lo tanto, se aplicaron pruebas diferentes según la métrica, por ejemplo, para *Precision* se aplicó **ANOVA de medidas repetidas** obteniendo $F(4,36) = 225.63$, $p < 0.0001$, $\eta^2_G = 0.9368$, $\varepsilon = 0.6389$, para las otras *acurracy* y *recall* se obtuvieron valores similares. Por otro lado, *F1-score* se aplicó **prueba de Friedman** obteniendo $Q(4) = 27.12$, $p < 0.0001$.

Las comparaciones post-hoc se realizaron con la corrección de Holm y los resultados se resumen en la Tabla(4.77).

Tabla 4.77: Comparaciones pareadas de modelos con la base de datos original y Top

A	B	Accuracy		F1-Score†			Precision		Recall	
		t	*p*	*Z*	*r*	*p*	*t*	*p*	*t*	*p*
CART	KNN	-8.182	0.0001	-2.701	-0.854	0.0195	-6.029	0.0006	-8.215	0.0001
CART	LR	-26.558	$< 10^{-4}$	-2.803	-0.886	0.0195	-23.705	$< 10^{-4}$	-28.531	$< 10^{-4}$
CART	NB	-9.762	$< 10^{-4}$	-2.803	-0.886	0.0195	-8.131	0.0001	-9.729	$< 10^{-4}$
CART	NN	-18.920	$< 10^{-4}$	-1.376	-0.435	0.7734	-23.073	$< 10^{-4}$	-18.724	$< 10^{-4}$
KNN	LR	-15.058	$< 10^{-4}$	-2.803	-0.886	0.0195	-14.897	$< 10^{-4}$	-16.062	$< 10^{-4}$
KNN	NB	1.923	0.0867	-0.968	-0.306	1.0000	-0.074	0.9429	1.967	0.0807
KNN	NN	-8.634	$< 10^{-4}$	-0.357	-0.113	1.0000	-11.451	$< 10^{-4}$	-8.428	0.0001
LR	NB	19.529	$< 10^{-4}$	-2.803	-0.886	0.0195	17.826	$< 10^{-4}$	21.371	$< 10^{-4}$
LR	NN	6.188	0.0003	-2.803	-0.886	0.0195	5.024	0.0014	6.175	0.0003
NB	NN	-11.696	$< 10^{-4}$	-0.459	-0.145	1.0000	-15.954	$< 10^{-4}$	-11.529	$< 10^{-4}$

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.

Para el subconjunto Bot, los contrastes de normalidad y homogeneidad de varianzas revelaron que los supuestos paramétricos solo se cumplen en la métrica *precision*; en *accuracy*, *recall* y *f1-score* dichos supuestos se violaron. En consecuencia, se utilizaron pruebas distintas según la métrica, para *Precision* se aplicó **ANOVA de medidas repetidas** se obtuvo $F(4,36) = 359.02$, $p < 0.0001$, $\eta^2_G = 0.9619$, $\varepsilon = 0.7975$, por otro lado, aplicando la **prueba de Friedman**, por ejemplo, en*Accuracy* se obtuvo $Q(4) = 39.28$, $p < 0.0001$, obteniendo valores similares para las otras métricas.

Las comparaciones post-hoc se corrigieron con Holm y los resultados se resumen en la Tabla (4.78).

Tabla 4.78: Comparaciones pareadas de modelos con la base de datos original y Bot

A	B	Accuracy†			F1-Score†			Precision		Recall†		
		Z	*r*	*p*	*Z*	*r*	*p*	*t*	*p*	*Z*	*r*	*p*
CART	KNN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-18.96	$< 10^{-4}$	-2.803	-0.886	0.0195
CART	LR	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-31.99	$< 10^{-4}$	-2.803	-0.886	0.0195
CART	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-11.29	$< 10^{-4}$	-2.803	-0.886	0.0195
CART	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-29.78	$< 10^{-4}$	-2.803	-0.886	0.0195
KNN	LR	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-5.77	0.0003	-2.803	-0.886	0.0195
KNN	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	12.19	$< 10^{-4}$	-2.803	-0.886	0.0195
KNN	NN	-2.803	-0.886	0.0195	-2.701	-0.854	0.0195	-9.95	$< 10^{-4}$	-2.803	-0.886	0.0195
LR	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	18.06	$< 10^{-4}$	-2.803	-0.886	0.0195
LR	NN	-2.701	-0.854	0.0195	-2.293	-0.725	0.0195	-7.38	0.0001	-2.701	-0.854	0.0195
NB	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-21.51	$< 10^{-4}$	-2.803	-0.886	0.0195

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.

Discretizaciónn K Means-CH

Para el conjunto *All*, los contrastes de *Shapiro-Wilk* y *Levene* mostraron que los supuestos paramétricos se cumplen únicamente en las métricas *accuracy* y *recall*; en cambio, *f1-score* y *precision* violaron dichos supuestos. En consecuencia, se aplicaron las siguientes pruebas estadísticas globales:

- *Accuracy*: ANOVA de medidas repetidas, $F(4,36) = 1947.21$, $p < 0.0001$, $\eta^2_G = 0.9948$, $\varepsilon = 0.7766$.
- *Recall*: ANOVA de medidas repetidas, $F(4,36) = 1999.18$, $p < 0.0001$, $\eta^2_G = 0.9949$, $\varepsilon = 0.7619$.
- *F1-score*: prueba de Friedman, $Q = 39.28$, $p < 0.0001$.
- *Precision*: prueba de Friedman, $Q = 38.72$, $p < 0.0001$.

Las comparaciones post-hoc se corrigieron con el método de Holm. Los resultados se resumen en la Tabla (4.79).

Tabla 4.79: Comparaciones pareadas de modelos con la discretización K Means–CH y All

A	B	Accuracy‡		F1-Score†			Precision†			Recall‡	
		t	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*t*	*p*
CART	KNN	-54.34	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-51.04	$< 10^{-4}$
CART	LR	-48.33	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-50.35	$< 10^{-4}$
CART	NB	-100.90	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-99.74	$< 10^{-4}$
CART	NN	-56.17	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-57.63	$< 10^{-4}$
KNN	LR	7.54	0.0001	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	7.39	0.0001
KNN	NB	-31.12	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-27.24	$< 10^{-4}$
KNN	NN	-7.64	0.0001	-2.803	-0.854	0.0195	-2.803	-0.886	0.0195	-7.67	0.0001
LR	NB	-62.44	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-65.20	$< 10^{-4}$
LR	NN	-15.71	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-18.72	$< 10^{-4}$
NB	NN	17.65	$< 10^{-4}$	-2.497	-0.886	0.0195	-2.497	-0.790	0.0195	18.88	$< 10^{-4}$

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.
‡ Prueba *t* pareada con corrección de Holm.

Para el subconjunto de *Top*, las pruebas de Shapiro–Wilk y Levene indicaron que los supuestos paramétricos se cumplen para las métricas *accuracy* y *recall*, mientras que se violan para *precision* y *f1-score*. Por consiguiente, se aplicaron las siguientes pruebas globales:

- *Accuracy*: ANOVA de medidas repetidas, $F(4,36) = 620.07$, $p < 0.0001$, $\eta^2_G = 0.9761$, $\varepsilon = 0.4429$.
- *Recall*: ANOVA de medidas repetidas, $F(4,36) = 648.85$, $p < 0.0001$, $\eta^2_G = 0.9783$, $\varepsilon = 0.4262$.
- *Precision*: prueba de Friedman, $Q(4) = 35.92$, $p < 0.0001$.
- *F1-score*: prueba de Friedman, $Q(4) = 35.44$, $p < 0.0001$.

Para las comparaciones post-hoc se empleó la corrección de Holm. La Tabla (4.80) resume los resultados.

Tabla 4.80: Comparaciones pareadas de modelos con la discretización K Means-CH y Top

A	**B**	**Accuracy**‡		**F1-Score**†			**Precision**†			**Recall**‡	
		t	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*t*	*p*
CART	KNN	-15.60	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-16.20	$< 10^{-4}$
CART	LR	-15.41	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-16.24	$< 10^{-4}$
CART	NB	-49.55	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-55.75	$< 10^{-4}$
CART	NN	-18.72	$< 10^{-4}$	-0.714	-0.226	0.4922	-2.803	-0.886	0.0195	-19.41	$< 10^{-4}$
KNN	LR	5.57	0.0010	-2.803	-0.886	0.0195	-2.701	-0.854	0.0195	5.59	0.0010
KNN	NB	-40.35	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-42.73	$< 10^{-4}$
KNN	NN	5.62	0.0010	-2.803	-0.886	0.0195	-1.784	-0.564	0.0840	5.62	0.0010
LR	NB	-31.24	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-31.36	$< 10^{-4}$
LR	NN	-3.16	0.0116	-1.376	-0.435	0.3867	-2.090	-0.661	0.0742	-2.92	0.0169
NB	NN	33.39	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	34.59	$< 10^{-4}$

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.
‡ Prueba *t* pareada con corrección de Holm.

Para el subconjunto *Bot*, los contrastes de normalidad y homogeneidad de varianzas confirmaron que los supuestos paramétricos se cumplen en las cuatro métricas analizadas. En consecuencia, se aplicó un **ANOVA de medidas repetidas** en cada métrica, obteniéndose por ejemplo *Recall*: $F(4,36) = 750.44$, $p < 0.0001$, $\eta^2_G = 0.9706$, $\varepsilon = 0.6007$, las demás métricas obtuvieron valores similares.

Las comparaciones post-hoc se realizaron con *t* **pareadas** y corrección de Holm. La Tabla (4.81) resume los resultados.

Tabla 4.81: Comparaciones pareadas de modelos con la discretización de K Means-CH y Bot

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
CART	KNN	-25.63	$< 10^{-4}$	-20.01	$< 10^{-4}$	-20.02	$< 10^{-4}$	-28.16	$< 10^{-4}$
CART	LR	-23.45	$< 10^{-4}$	-15.26	$< 10^{-4}$	-17.32	$< 10^{-4}$	-26.12	$< 10^{-4}$
CART	NB	-38.71	$< 10^{-4}$	-34.48	$< 10^{-4}$	-35.59	$< 10^{-4}$	-42.07	$< 10^{-4}$
CART	NN	-35.39	$< 10^{-4}$	-11.60	$< 10^{-4}$	-20.13	$< 10^{-4}$	-38.84	$< 10^{-4}$
KNN	LR	6.66	0.0002	11.02	$< 10^{-4}$	8.90	$< 10^{-4}$	6.63	0.0002
KNN	NB	-14.80	$< 10^{-4}$	-13.74	$< 10^{-4}$	-13.47	$< 10^{-4}$	-15.63	$< 10^{-4}$
KNN	NN	-2.29	0.0476	0.35	0.7351	-0.76	0.4695	-2.27	0.0493
LR	NB	-28.27	$< 10^{-4}$	-35.50	$< 10^{-4}$	-32.81	$< 10^{-4}$	-30.05	$< 10^{-4}$
LR	NN	-12.09	$< 10^{-4}$	-3.67	0.0103	-5.85	0.0005	-12.32	$< 10^{-4}$
NB	NN	14.14	$< 10^{-4}$	6.77	0.0002	9.19	$< 10^{-4}$	14.21	$< 10^{-4}$

Discretizaciónn K Means-DB

En la discretización de **K Means-DB**, las pruebas de normalidad y homogeneidad de varianzas indicaron violaciones de los supuestos paramétricos en las cuatro métricas evaluadas. En consecuencia, se utilizó la **prueba de Friedman** para los contrastes globales en cada métrica, por ejemplo, para *Accuracy* se obtuvo $Q(4) = 38.72$, $p < 0.0001$, obteniendo valores similares para las demás métricas.

Para las comparaciones post-hoc se aplicó **Wilcoxon pareado** con corrección de Holm. La Tabla (4.82) presenta los valores de los estadísticos de prueba para cada par de modelos.

Tabla 4.82: Comparaciones pareadas de modelos con la discretización K Means–DB y *All*

A	B	Accuracy			F1-Score			Precision			Recall		
		Z	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*
CART	KNN	-2.293	-0.725	0.0195	0.000	0.000	1.0000	-1.376	-0.435	0.3867	-2.446	-0.774	0.0195
CART	LR	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
CART	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
CART	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
KNN	LR	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
KNN	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
KNN	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
LR	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
LR	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
NB	NN	-2.803	-0.886	0.0195	-2.701	-0.854	0.0257	-1.325	-0.419	0.3867	-2.803	-0.886	0.0195

Las pruebas de normalidad y homogeneidad de varianzas mostraron que los supuestos paramétricos se cumplen en las métricas *accuracy* y *recall*, pero se violan en *precision* y *f1-score*. En consecuencia, se aplicaron pruebas globales diferentes según la métrica:

- *Accuracy*: ANOVA de medidas repetidas, $F(4,36) = 681.33$, $p < 0.0001$, $\eta^2_G = 0.9842$, $\varepsilon = 0.7343$.
- *Recall*: ANOVA de medidas repetidas, $F(4,36) = 750.06$, $p < 0.0001$, $\eta^2_G = 0.9865$, $\varepsilon = 0.7462$.
- *Precision*: prueba de Friedman, $Q(4) = 38.32$, $p < 0.0001$.

- *F1-score*: prueba de Friedman, $Q(4) = 32.40$, $p < 0.0001$.

Las comparaciones post-hoc se corrigieron con el método de Holm. Los resultados se resumen en la Tabla (4.83).

Tabla 4.83: Comparaciones pareadas de modelos con la discretización de K Means-DB y Top

A	**B**	**Accuracy**‡		**F1-Score**†			**Precision**†			**Recall**‡	
		t	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*t*	*p*
CART	KNN	6.6908	0.0002	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	6.7370	0.0002
CART	LR	-14.7382	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-15.6661	$< 10^{-4}$
CART	NB	-47.1946	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-53.7160	$< 10^{-4}$
CART	NN	-14.0819	$< 10^{-4}$	-0.968	-0.306	0.5508	-2.803	-0.886	0.0195	-16.0131	$< 10^{-4}$
KNN	LR	-16.9928	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-17.4953	$< 10^{-4}$
KNN	NB	-39.4039	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-41.6802	$< 10^{-4}$
KNN	NN	-18.3011	$< 10^{-4}$	-1.784	-0.564	0.2520	-2.803	-0.886	0.0195	-19.7175	$< 10^{-4}$
LR	NB	-27.2724	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-27.2404	$< 10^{-4}$
LR	NN	3.2808	0.0095	-1.223	-0.387	0.5508	-1.988	-0.629	0.0488	3.2936	0.0093
NB	NN	28.4552	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	28.8882	$< 10^{-4}$

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.
‡ Prueba *t* pareada con corrección de Holm.

Para el subconjunto **Bot**, los contrastes de normalidad y homogeneidad de varianzas mostraron que los supuestos paramétricos se cumplen en las métricas *accuracy*, *precision* y *recall*, pero se violan en *f1-score*. En consecuencia, se emplearon pruebas diferentes según la métrica, por ejemplo para la métrica *Recall* se aplicó **ANOVA de medidas repetidas**, obteniendo $F(4,36) = 978.52$, $p < 0.0001$, $\eta^2_G = 0.9854$, $\varepsilon = 0.5907$, para las métricas *accuracy* y *precision* se obtuvo valores similares, mientras que, *F1-score* se aplicó la prueba de Friedman, obteniendo $Q(4) = 35.68$, $p < 0.0001$ respectivamente.

Las comparaciones post-hoc se corrigieron con el método de Holm: *t* pareadas para las métricas paramétricas y prueba de Wilcoxon pareado para la métrica no paramétrica (*f1-score*). La Tabla (4.84) resume los resultados.

Tabla 4.84: Comparaciones pareadas de modelos con la discretización de K Means-DB y Bot

A	**B**	**Accuracy**		**F1-Score**†			**Precision**		**Recall**	
		t	*p*	*Z*	*r*	*p*	*t*	*p*	*t*	*p*
CART	KNN	-5.083	0.0007	-1.580	-0.500	0.131	-1.722	0.119	-5.107	0.0006
CART	LR	-38.172	$< 10^{-4}$	-2.803	-0.886	0.0195	-28.964	$< 10^{-4}$	-37.883	$< 10^{-4}$
CART	NB	-63.498	$< 10^{-4}$	-2.803	-0.886	0.0195	-44.604	$< 10^{-4}$	-65.629	$< 10^{-4}$
CART	NN	-42.006	$< 10^{-4}$	-2.803	-0.886	0.0195	-22.714	$< 10^{-4}$	-39.753	$< 10^{-4}$
KNN	LR	-21.526	$< 10^{-4}$	-2.803	-0.886	0.0195	-16.329	$< 10^{-4}$	-21.294	$< 10^{-4}$
KNN	NB	-34.728	$< 10^{-4}$	-2.803	-0.886	0.0195	-26.257	$< 10^{-4}$	-35.044	$< 10^{-4}$
KNN	NN	-27.863	$< 10^{-4}$	-2.803	-0.886	0.0195	-17.944	$< 10^{-4}$	-27.499	$< 10^{-4}$
LR	NB	-25.998	$< 10^{-4}$	-2.803	-0.886	0.0195	-19.237	$< 10^{-4}$	-26.146	$< 10^{-4}$
LR	NN	-9.893	$< 10^{-4}$	-2.192	-0.693	0.055	-4.647	0.0024	-9.985	$< 10^{-4}$
NB	NN	11.279	$< 10^{-4}$	-2.650	-0.838	0.0195	5.194	0.0017	11.900	$< 10^{-4}$

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.

Discretizaciónn MDLP

Para la discretización **MDLP** y el conjunto *All*, los contrastes de normalidad y homogeneidad de varianzas indicaron que **ninguna métrica** cumplía con los supuestos paramétricos. Por lo tanto, se utilizó la **prueba de Friedman** para los contrastes globales, por ejemplo, la métrica *Recall* calculó $Q(4) = 38.21$, $p < 0.0001$, para las demás fueron valores similares.

Las comparaciones post-hoc se realizaron con la prueba de **Wilcoxon pareado** y corrección de Holm. La Tabla (4.85) resume los valores estadísticos por cada par de modelos.

Tabla 4.85: Comparaciones pareadas de modelos con la discretización MDLP y *All*

A	**B**	**Accuracy**			**F1-Score**			**Precision**			**Recall**		
		Z	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*
CART	KNN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
CART	LR	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
CART	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
CART	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
KNN	LR	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
KNN	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
KNN	NN	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
LR	NB	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195
LR	NN	-0.357	-0.113	0.7695	-0.051	-0.016	1.0000	-2.701	-0.854	0.0195	-0.968	-0.306	0.6356
NB	NN	-2.803	-0.886	0.0195	-2.293	-0.725	0.0391	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195

Para el subconjunto **Top**, los contrastes de Shapiro–Wilk y Levene mostraron que los supuestos paramétricos se cumplen en las métricas *accuracy* y *recall*, pero se violan en *precision* y *f1-score*. Por lo tanto, se aplicaron las siguientes pruebas:

- *Accuracy*: ANOVA de medidas repetidas, $F(4,36) = 455.53$, $p < 0.0001$, $\eta^2_G = 0.9711$, $\varepsilon = 0.5675$.
- *Recall*: ANOVA de medidas repetidas, $F(4,36) = 497.57$, $p < 0.0001$, $\eta^2_G = 0.9747$, $\varepsilon = 0.5481$.
- *Precision*: prueba de Friedman, $Q(4) = 32.50$, $p < 0.0001$.
- *F1-score*: prueba de Friedman, $Q(4) = 36.68$, $p < 0.0001$.

Las comparaciones post-hoc se corrigieron con Holm. Para *accuracy* y *recall* se aplicaron *t* pareadas, y para *precision* y *f1-score*, la prueba de Wilcoxon pareado. La Tabla (4.86) muestra los resultados.

Tabla 4.86: Comparaciones pareadas de modelos con la discreitización MDLP y Top

A	B	Accuracy‡		F1-Score†			Precision†			Recall‡	
		t	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*t*	*p*
CART	KNN	-28.11	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-28.45	$< 10^{-4}$
CART	LR	-24.54	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-25.21	$< 10^{-4}$
CART	NB	-28.05	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	-29.56	$< 10^{-4}$
CART	NN	-18.92	$< 10^{-4}$	-0.357	-0.113	0.7695	-1.784	-0.564	0.2520	-20.38	$< 10^{-4}$
KNN	LR	10.41	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	10.65	$< 10^{-4}$
KNN	NB	15.68	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0195	16.93	$< 10^{-4}$
KNN	NN	22.92	$< 10^{-4}$	-2.803	-0.886	0.0195	-2.803	-0.886	0.0458	22.52	$< 10^{-4}$
LR	NB	-0.17	0.8721	-1.988	-0.629	0.1944	-2.497	-0.790	0.0458	-0.15	0.8871
LR	NN	5.72	0.0006	-2.803	-0.886	0.0195	-0.612	-0.193	0.6250	6.82	0.0002
NB	NN	6.85	0.0002	-2.803	-0.886	0.0195	-1.376	-0.435	0.3867	7.15	0.0002

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.
‡ Prueba *t* pareada con corrección de Holm.

Para el subconjunto **Bot** discretizado con *MDLP*, los contrastes de normalidad y homogeneidad de varianzas mostraron que los supuestos paramétricos se cumplen en *f1-score*, *precision* y *recall*, pero se violan en *accuracy*. En consecuencia, se aplicaron pruebas diferentes según la métrica, por ejemplo, para *Accuracy* se prueba de Friedman, obteniendo los valores de $Q(4) = 34.24$, $p < 0.0001$, por otro lado, a las métricas como *F1-score* se aplicó ANOVA de medidas repetidas, obteniendo $F(4,36) = 380.57$, $p < 0.0001$, $\eta^2_G = 0.9702$, $\varepsilon = 0.6694$, valores similares se obtuvieron de las otras métricas.

Para las comparaciones post-hoc se usó la corrección de Holm: *Wilcoxon* pareado para *accuracy* y *t* pareadas para las demás métricas. La Tabla (4.87) resume los resultados.

Tabla 4.87: Comparaciones pareadas de modelos con la discretización de MDLP y Bot

A	B	Accuracy†			F1-Score		Precision		Recall	
		Z	*r*	*p*	*t*	*p*	*t*	*p*	*t*	*p*
CART	KNN	-2.803	-0.886	0.0195	-28.70	$< 10^{-4}$	-28.94	$< 10^{-4}$	-30.69	$< 10^{-4}$
CART	LR	-2.803	-0.886	0.0195	-33.40	$< 10^{-4}$	-35.85	$< 10^{-4}$	-43.80	$< 10^{-4}$
CART	NB	-2.803	-0.886	0.0195	-10.55	$< 10^{-4}$	-11.92	$< 10^{-4}$	-13.95	$< 10^{-4}$
CART	NN	-2.803	-0.886	0.0195	-31.45	$< 10^{-4}$	-33.81	$< 10^{-4}$	-43.58	$< 10^{-4}$
KNN	LR	-2.701	-0.854	0.0195	5.56	0.0011	4.52	0.0043	-3.23	0.0310
KNN	NB	-2.803	-0.886	0.0195	19.85	$< 10^{-4}$	18.99	$< 10^{-4}$	20.59	$< 10^{-4}$
KNN	NN	-1.784	-0.564	0.1680	0.72	0.4886	0.73	0.4857	-2.09	0.1319
LR	NB	-2.803	-0.886	0.0195	20.04	$< 10^{-4}$	18.78	$< 10^{-4}$	29.62	$< 10^{-4}$
LR	NN	-0.459	-0.145	0.6953	-3.10	0.0256	-3.70	0.0099	0.63	0.5472
NB	NN	-2.803	-0.886	0.0195	-14.61	$< 10^{-4}$	-13.83	$< 10^{-4}$	-19.90	$< 10^{-4}$

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.

4.7. Pruebas estadísticas de los métodos de selección/reducción de características

Para analizar el efecto de la selección/reducción de carcterísticas sobre el desempeño de cada modelo se aplicó una prueba de normalidad de Shapiro-Wilk y la prueba de homogeneidad de varianzas de Levene. Dependiendo de los supuestos se procedió a hacer pruebas paramétricas

o no paramétricas. Para el caso de la prueba paramétrica, se aplica ANOVA de medidas repetidas y prueba t pareada con corrección Holm. En el caso de la prueba no paramétrica se aplica Friedman y prueba de Wilcoxon con corrección Holm. Lo anterior únicamente se aplicó para los resultados obtenidos con 3 clases en la categorización de la variable objetivo ya que, se hallaron los resultados más elevados con dicho número. Y sólo se aplicará al conjunto de datos *All*, se busca encontrar las mejores características que expliquen toda la base de datos.

Naive Bayes

Para el modelo **Naive Bayes** evaluado sobre el conjunto completo *All*,as pruebas de normalidad de Shapiro-Wilk y homocedasticidad de Levene mostraron que violaciones significativas en todas las métricas. Por lo tanto, se aplicaron pruebas no paramétricas tanto en el contraste global como en las comparaciones post-hoc.

Se aplicó la prueba de **Friedman**, encontrando diferencias estadísticamente significativas entre los métodos de selección en todas las métricas, con los siguientes resultados: $Q(3) = 30.0$, $p < 0.0001$.

Para las comparaciones post-hoc se utilizó la corrección de Holm sobre la prueba de *Wilcoxon* pareada. La Tabla (4.88) resume los resultados obtenidos.

Tabla 4.88: Comparaciones pareadas de métodos de selección en Naive Bayes

A	**B**	**Accuracy**			**F1-Score**			**Precision**			**Recall**		
		Z	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*
Gu	Kelly	-2.803	-0.886	0.0117	-2.803	-0.886	-2.803	-0.886	-2.803	-0.886	-0.886	-2.803	-0.886
Gu	PCA	-2.803	-0.886	0.0117	-2.803	-0.886	-2.803	-0.886	-2.803	-0.886	-0.886	-2.803	-0.886
Gu	RB	-2.803	-0.886	0.0117	-2.803	-0.886	-2.803	-0.886	-2.803	-0.886	-0.886	-2.803	-0.886
Kelly	PCA	-2.803	-0.886	0.0117	-2.803	-0.886	-2.803	-0.886	-2.803	-0.886	-0.886	-2.803	-0.886
Kelly	RB	-2.803	-0.886	0.0117	-2.803	-0.886	-2.803	-0.886	-2.803	-0.886	-0.886	-2.803	-0.886
PCA	RB	-2.803	-0.886	0.0117	-2.803	-0.886	-2.803	-0.886	-2.803	-0.886	-0.886	-2.803	-0.886

CART

Para el modelo **CART** evaluado sobre el conjunto completo *All*, las pruebas de normalidad de Shapiro-Wilk y homocedasticidad de Levene confirmaron que los supuestos paramétricos se cumplen en todas las métricas. Por lo tanto, se aplicaron pruebas paramétricas tanto en el contraste global como en las comparaciones post-hoc.

Se aplicó la prueba de **ANOVA de medidas repetidas**, encontrando diferencias estadísticamente significativas entre los métodos de selección en todas las métricas, con los siguientes resultados: $F(3,27) > 1580$, $p < 0.0001$, $\eta^2_G \approx 0.993$, $\varepsilon \approx 0.72$.

Para las comparaciones post-hoc se utilizó la corrección de Holm sobre la prueba de *t* pareadas. La Tabla (4.89) resume los resultados obtenidos.

Tabla 4.89: Comparaciones pareadas de métodos de selección en CART

A	B	Accuracy		F1-Score		Precision		Recall	
		t	*p*	*t*	*p*	*t*	*p*	*t*	*p*
Gu	Kelly	35.910	< 0.0001	36.027	< 0.0001	35.918	< 0.0001	35.210	< 0.0001
Gu	PCA	57.446	< 0.0001	58.489	< 0.0001	58.161	< 0.0001	57.239	< 0.0001
Gu	RB	-13.931	< 0.0001	-13.961	< 0.0001	-13.928	< 0.0001	-13.936	< 0.0001
Kelly	PCA	12.753	< 0.0001	12.740	< 0.0001	12.626	< 0.0001	12.761	< 0.0001
Kelly	RB	-36.221	< 0.0001	-36.621	< 0.0001	-36.887	< 0.0001	-36.305	< 0.0001
PCA	RB	-83.062	< 0.0001	-82.836	< 0.0001	-82.013	< 0.0001	-82.474	< 0.0001

Regresión Logística

Para el modelo **Regresión Logística** evaluado sobre el conjunto *All*, las pruebas de Shapiro–Wilk y Levene indicaron que los supuestos paramétricos se cumplen en *accuracy*, *f1-score* y *precision*, pero se violan en *recall*. En consecuencia, se aplicó un **ANOVA de medidas repetidas** en las tres primeras métricas y la prueba no paramétrica de **Friedman** en *recall*. Los contrastes globales arrojaron, por ejemplo, en la métrica de *Accuracy*: $F(3,27) = 1408.78$, $p < 0.0001$, $\eta^2_G \approx 0.979$, $\varepsilon \approx 0.910$, para *f1-score* y *precision* obtuvieron valores muy similares. Por otro lado, *Recall* obtuvo $Q(3) = 28.08$, $p < 0.0001$.

Para las comparaciones post-hoc se empleó corrección de Holm: prueba *t* pareada en las métricas paramétricas y *Wilcoxon* pareado en *recall*. La Tabla 4.90 resume los resultados.

Tabla 4.90: Comparaciones pareadas de métodos de selección en Regresión Logística

A	B	Accuracy		F1-Score		Precision		Recall†		
		t	*p*	*t*	*p*	*t*	*p*	*Z*	*r*	*p*
Gu	Kelly	69.5741	< 0.0001	81.3647	< 0.0001	62.0962	< 0.0001	-2.8031	-0.8864	0.0117
Gu	PCA	47.4055	< 0.0001	72.8098	< 0.0001	36.0491	< 0.0001	-2.8031	-0.8864	0.0117
Gu	RB	11.6464	< 0.0001	35.4502	< 0.0001	24.1088	< 0.0001	-2.8031	-0.8864	0.0117
Kelly	PCA	-1.8094	0.1038	15.0877	< 0.0001	-2.7080	0.0241	-1.4270	-0.4513	0.1602
Kelly	RB	-39.8285	< 0.0001	-24.4279	< 0.0001	-25.2640	< 0.0001	-2.8031	-0.8864	0.0117
PCA	RB	-35.1227	< 0.0001	-38.8590	< 0.0001	-18.1072	< 0.0001	-2.8031	-0.8864	0.0117

† Wilcoxon pareado con corrección de Holm ($N = 10$); $r = Z/\sqrt{10}$.

K Vecinos más Cercanos

Para el modelo **K Vecinos Más Cercanos** evaluado sobre el conjunto completo *All*, las pruebas de Shapiro–Wilk y Levene mostraron violaciones significativas de normalidad y homocedasticidad en las métricas. Por ello se emplearon métodos no paramétricos tanto en el contraste global como en las comparaciones post-hoc.

Se aplicó la prueba de **Friedman**, encontrando diferencias estadísticamente significativas entre los cuatro métodos de selección en todas las métricas: $Q(3) = 30.0$, $p < 0.0001$.

Para las comparaciones post-hoc se utilizó la corrección de Holm sobre la prueba de *Wilcoxon* pareada. La Tabla 4.91 resume los resultados.

Tabla 4.91: Comparaciones pareadas de métodos de selección en K Vecinos Más Cercanos

A	B	Accuracy			F1-Score			Precision			Recall		
		Z	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*
Gu	Kelly	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117
Gu	PCA	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117
Gu	RB	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117
Kelly	PCA	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117
Kelly	RB	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117
PCA	RB	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117	-2.803	-0.886	0.0117

Redes Neuronales

Para el modelo **Redes Neuronales** evaluado sobre el conjunto completo *All*, las pruebas de Shapiro–Wilk y Levene revelaron violaciones significativas de normalidad y homocedasticidad en todas las métricas: *accuracy*, *f1-score*, *precision* y *recall*. En consecuencia, se empleó la prueba no paramétrica de **Friedman** tanto en el contraste global como en las comparaciones post-hoc. La mayoría de las métricas daban resultados similares, por ejemplo, *Precision* dio $Q(3) = 15.48$, $p = 0.0014$.

Para las comparaciones post-hoc se utilizó la prueba de *Wilcoxon* pareada con corrección de Holm. La Tabla 4.92 resume los resultados.

Tabla 4.92: Comparaciones pareadas de métodos de selección en Redes Neuronales

A	B	Accuracy			F1-Score			Precision			Recall		
		Z	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*	*Z*	*r*	*p*
Gu	Kelly	-2.803	-0.886	0.0117	-2.497	-0.789	0.0488	-2.599	-0.822	0.0234	-2.803	-0.886	0.0117
Gu	PCA	-2.803	-0.886	0.0117	-0.153	-0.048	0.9219	-2.701	-0.854	0.0195	-2.803	-0.886	0.0117
Gu	RB	-2.701	-0.854	0.0156	-1.070	-0.339	0.8262	-2.803	-0.886	0.0117	-2.701	-0.854	0.0156
Kelly	PCA	-1.478	-0.467	0.1602	-2.803	-0.886	0.0117	-1.070	-0.339	0.6445	-1.478	-0.467	0.1602
Kelly	RB	-2.599	-0.822	0.0176	-2.497	-0.790	0.0488	-1.784	-0.564	0.2520	-2.599	-0.822	0.0176
PCA	RB	-2.599	-0.822	0.0356	-1.172	-0.371	0.8262	-0.051	-0.016	1.0000	-2.344	-0.741	0.0273

4.8. Modelado a Seleccionar

Los mejores resultados fueron obtenidos con la base de datos original y su subconjunto *Bot* el cual representa las peores 1000 firmas agrupadas por su volumen a lo largo de los 20 años, utilizando únicamente **3 clases** en la categorización de la variable objetivo **Retorno**.
El modelo KNN representó los mejores resultados en la evaluación de las métricas, sin embargo, su tiempo de ejecución es muy grande que impide que su replicabilidad sea sencilla de aplicar. En la selección de características MDL-FS obtuvo los mejores resultados y disminuye el tiempo de ejecución del modelo, esto se ve en la Tabla (4.93).

Tabla 4.93: Evaluación de All con KNN con $k = 3$

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)	K
Original	0.444 ± 0.001	0.438 ± 0.001	0.438 ± 0.001	0.444 ± 0.001	**2163**	20
K-Means(C-H)	0.420 ± 0.001	0.415 ± 0.001	0.415 ± 0.001	0.420 ± 0.001	36406	20
K-Means(D-B)	0.394 ± 0.001	0.393 ± 0.001	0.391 ± 0.001	0.394 ± 0.001	36935	20
MDLP	$\mathbf{0.513 \pm 0.001}$	$\mathbf{0.513 \pm 0.001}$	$\mathbf{0.512 \pm 0.001}$	$\mathbf{0.513 \pm 0.001}$	38648	20
Redes Bayesianas [22]	$\mathbf{0.519 \pm 0.001}$	$\mathbf{0.520 \pm 0.001}$	$\mathbf{0.519 \pm 0.001}$	$\mathbf{0.519 \pm 0.001}$	360	20
PCA	0.432 ± 0.001	0.428 ± 0.001	0.428 ± 0.001	0.432 ± 0.001	2937	20
Fts MDLP [47]	0.445 ± 0.001	0.442 ± 0.001	0.442 ± 0.001	0.445 ± 0.001	3261	20
Fts MDLP [35]	0.490 ± 0.001	0.490 ± 0.001	0.490 ± 0.001	0.489 ± 0.001	30640	20

Tabla 4.94: Evaluación de All con Naive Bayes con $k = 3$

Conjunto	Accuracy	Precision	F1-Score	Recall	E-T (S)
Original	0.430 ± 0.001	0.424 ± 0.001	0.423 ± 0.001	0.430 ± 0.001	1033
K-Means(C-H)	0.435 ± 0.001	0.437 ± 0.001	0.434 ± 0.001	0.435 ± 0.001	549
K-Means(D-B)	$\mathbf{0.446 \pm 0.001}$	0.439 ± 0.001	0.440 ± 0.001	0.446 ± 0.001	507
MDLP	0.443 ± 0.001	$\mathbf{0.450 \pm 0.001}$	$\mathbf{0.444 \pm 0.001}$	$\mathbf{0.443 \pm 0.001}$	**484**
Redes Bayesianas [22]	$\mathbf{0.493 \pm 0.001}$	$\mathbf{0.489 \pm 0.001}$	$\mathbf{0.489 \pm 0.001}$	$\mathbf{0.493 \pm 0.001}$	**91**
PCA	0.332 ± 0.001	0.111 ± 0.000	0.166 ± 0.000	0.333 ± 0.000	178
Fts MDLP [47]	0.443 ± 0.001	0.437 ± 0.001	0.437 ± 0.001	0.443 ± 0.001	124
Fts MDLP [35]	0.486 ± 0.001	0.482 ± 0.001	0.482 ± 0.001	0.486 ± 0.001	269

Por otro lado, el modelo más equilibrado en su clasificación y tiempo de ejecución fue el algoritmo de Naive Bayes, se muestra en la Tabla (4.94) que sus mejores resultados fueron con la discretización de MDLP y aplicando selección de característcas con Redes Bayesianas.

Esto sirve para saber qué enfoque utilizar para la siguiente sección donde se ve la aplicación de todo esto, pero en una firma en específico.

4.9. Aplicación de la Investigación en una Firma

En esta sección, se mostrará la eficiencia de todo este trabajo de investigación para llevarlo a cabo en las firmas que han prevalecido por bastante tiempo, como pueden ser ejemplos de la S&P 500. Los ejemplos a tomar, únicamente son una muestra, no represetan nada en especial, ya que se desconoce la naturaleza de la empresa. Los ejemplos a tomar son las firmas **10065** y **10516**.

Figura 4.1: Flujo de trabajo seguido para el análisis de clasificación de una firma.

La Figura (4.1) muestra el flujo de trabajo a seguir para el análisis de clasificación de una firma.

Una vez seguido el proceso explicado, la primera firma **10065** de la base de datos, teniendo 757 observaciones en total. Recordando que, las secciones 4.1-4.5 de Resultados (págs. 94-117), la base de datos eran más de 1.4 millones de instancias con 94 características. Esta tuvo los siguientes resultados por parte de Naive Bayes (4.95) y KNN (4.96), los cuales fueron entrenados únicamente con esas 757 observaciones de esas dos firmas utilizadas como ejemplos.

Modelo	Accuracy	Precision	Recall	F1 Score
NB MDLP	0.530 ± 0.043	0.338 ± 0.074	0.338 ± 0.065	0.331 ± 0.066
NB MDL-FS	$\mathbf{0.554 \pm 0.039}$	0.352 ± 0.072	$\mathbf{0.361 \pm 0.056}$	$\mathbf{0.349 \pm 0.057}$
NB Gu [35]	0.544 ± 0.046	$\mathbf{0.357 \pm 0.065}$	0.351 ± 0.059	0.347 ± 0.058

Tabla 4.95: Comparación del rendimiento de Naive Bayes en la firma **10065**.

Modelo	Accuracy	Precision	Recall	F1 Score
KNN MDLP	$\mathbf{0.649 \pm 0.047}$	0.269 ± 0.109	0.332 ± 0.010	0.271 ± 0.023
KNN MDL-FS	0.645 ± 0.047	0.252 ± 0.098	0.328 ± 0.012	0.264 ± 0.014
KNN Gu [35]	0.630 ± 0.049	$\mathbf{0.380 \pm 0.088}$	$\mathbf{0.351 \pm 0.027}$	$\mathbf{0.324 \pm 0.039}$

Tabla 4.96: Comparación del rendimiento de KNN en la firma **10065**.

De los anteriores resultados se aprecia que la selección de características es la metodología que mejores resultados se obtienen con la base de datos discretizada. En los valores en las métricas, *Accuracy* es la más alta, sin embargo, tiene valores bajos en las demás métricas, donde los falsos positivos y falsos negativos son altos, siendo posiblemente las pocas instancias por firma una posible causa de estos valores bajos.

Aplicando el mismo flujo de trabajo a la firma **10516** de 757 observaciones, se obtuvieron los siguientes resultados por parte de Naive Bayes (4.97) y KNN (4.98). Se halló un mejor equilibrio entre los valores de las métricas de evaluación, aunque *Accuracy* sigue siendo la métrica más alta, las otras no están tan alejadas de su valor, además, la selección de características por parte de MDL-FS para esta firma fue la mejor opción, dando los mejores resultados en NB y KNN.

Modelo	Accuracy	Precision	Recall	F1 Score
NB MDLP	0.398 ± 0.051	0.389 ± 0.054	0.384 ± 0.055	0.376 ± 0.054
NB MDL-FS	$\mathbf{0.432 \pm 0.055}$	$\mathbf{0.414 \pm 0.051}$	$\mathbf{0.411 \pm 0.046}$	$\mathbf{0.405 \pm 0.047}$
NB Gu [35]	0.419 ± 0.047	0.396 ± 0.043	0.401 ± 0.038	0.393 ± 0.041

Tabla 4.97: Comparación del rendimiento de Naive Bayes en la firma **10516**.

Modelo	Accuracy	Precision	Recall	F1 Score
KNN MDLP	0.404 ± 0.050	0.377 ± 0.049	0.375 ± 0.040	0.364 ± 0.044
KNN MDL-FS	$\mathbf{0.409 \pm 0.057}$	$\mathbf{0.385 \pm 0.055}$	$\mathbf{0.387 \pm 0.051}$	$\mathbf{0.378 \pm 0.053}$
KNN Gu [35]	0.376 ± 0.061	0.343 ± 0.057	0.343 ± 0.038	0.322 ± 0.045

Tabla 4.98: Comparación del rendimiento de KNN en la firma **10516**.

4.10. Discusión

Los resultados obtenidos a lo largo de esta investigación reflejan la complejidad del problema de clasificación de activos financieros, especialmente al trabajar con bases de datos de gran

volumen y alta dimensionalidad. La base original, compuesta por más de 1.4 millones de registros y 94 características numéricas, requirió una estructuración y segmentación para permitir su análisis eficiente. Esta necesidad motivó la creación de dos subconjuntos diferenciados como lo hizo Gu et Al. [36]: uno con las 1000 firmas de mayor rendimiento histórico y otro con las 1000 de menor rendimiento, ambos agrupados según el tamaño de la empresa y con 240,000 instancias cada uno.

La decisión de transformar la variable objetivo "Retorno" en versiones categorizadas de tres y cinco clases permitió replantear el problema como una tarea de clasificación, evitando así los desafíos del modelado directo de series continuas, como el sobreajuste en modelos de regresión. Los resultados indicaron que la configuración de tres clases proporcionó mejores resultados en la mayoría de los clasificadores, debido a una mayor densidad de ejemplos por clase y menor dispersión.

En cuanto a la transformación de las características, se utilizaron dos discretizadores con enfoques distintos: **MDLP** (supervisado) y **K-means** (no supervisado), evaluando este último mediante los índices de Davies-Bouldin (3.11) y Calinski-Harabasz (3.9) para seleccionar el valor óptimo de k. La elección de discretizar los datos se justificó en función del impacto positivo que esta transformación tiene sobre modelos sensibles a la distribución de datos, como Naive Bayes y CART. En particular, MDLP (3.13 y 3.15) mostró ser más efectivo para estos modelos al preservar las relaciones entre las variables discretizadas y la variable objetivo, mientras que K-Means en la sección de metodología se mostró que muchas características tuvieron un alto porcentaje de los datos en una sola etiqueta, haciendo que muchas de estas fueron una posible columna constante lo cual podría eliminar información relevante para los modelos de clasificación, arrojando valores bajos en los rendimientos de las métricas.

La combinación de discretización y selección de características también evidenció efectos relevantes. La aplicación de MDL-FS y de criterios teóricos como los propuestos por Gu et Al. [35] y Kelly et Al. [47] redujo significativamente la dimensionalidad del conjunto de datos, lo que se tradujo en tiempos de ejecución considerablemente menores sin comprometer la calidad de las métricas de clasificación, y en varios casos, mejorando la calidad de las métricas. Esta mejora fue especialmente visible en modelos como Naive Bayes, KNN y Regresión Logística, cuyos tiempos de entrenamiento son sensibles al número de características.

En contraste, la reducción de dimensionalidad mediante **PCA** no produjo mejoras consistentes en las métricas. Aunque sí permitió acelerar los tiempos de ejecución con el costo de perder de información en los datos y que la transformación de las características a un nuevo espacio proyectado dificultó la captura de relaciones relevantes para algunos modelos. Este fenómeno fue más notorio en clasificadores como Naive Bayes, CART y KNN, que dependen directamente de la representación espacial original de los datos.

Uno de los hallazgos más relevantes fue la variabilidad del desempeño entre modelos. El algoritmo **KNN** alcanzó los mayores valores en las métricas de evaluación bajo ciertas combinaciones de discretización y selección, particularmente con MDLP. Esto puede explicarse por la naturaleza del modelo: al ser un clasificador basado en distancias. La discretización mediante MDLP ayudó a que las clases quedaran mejor definidas en el espacio de características, lo que permitió una clasificación más precisa. No obstante, este modelo presentó un tiempo de ejecución considerablemente alto, ya que su fase de predicción implica calcular distancias para cada instancia del conjunto de entrenamiento, lo cual es costoso computacionalmente, en especial con

grandes volúmenes de datos.

Por otro lado, **Naive Bayes** ofreció un equilibrio ideal entre precisión y eficiencia, confirmando su idoneidad cuando los datos están discretizados adecuadamente. Este modelo se basa en la probabilidad condicional de cada clase dado cada característica y asume independencia entre las características, por lo que discretizaciones supervisadas como MDLP permiten una mejor partición del espacio de probabilidad. Además, su simplicidad lo hace altamente eficiente en cuanto a tiempo de cómputo, convirtiéndolo en una opción viable para bases de datos de gran escala como la utilizada en este trabajo.

En contraparte, algunos modelos presentaron un desempeño considerablemente más bajo. El algoritmo **CART**, modelo de árboles de decisión, mostró limitaciones tanto en precisión como en estabilidad. Esto puede atribuirse a su sensibilidad ante discretizaciones con demasiadas clases poco balanceadas o características con escasa relevancia informativa, lo que genera árboles demasiado específicos o, por el contrario, reglas poco generalizables. Además, la alta dimensionalidad inicial pudo dificultar su capacidad para encontrar divisiones informativas sin una previa reducción o selección adecuada.

En el caso de la **Regresión Logística**, su comportamiento fue intermedio en comparación con los demás modelos. Aunque no alcanzó los mejores resultados en ninguna de las métricas, sí mostró un desempeño relativamente estable en múltiples configuraciones. Este modelo se vio beneficiado especialmente por la selección de características basada en teoría como Gu et al. [36] y MDL-FS, ya que al reducir el número de características irrelevantes, mejoró la capacidad predictiva del modelo sin comprometer la interpretabilidad, lo mismo con la reducción de dimensionalidad con PCA, sólo que con este método los tiempos de ejecución fueron muy bajos comparados con las otras configuraciones. Su bajo tiempo de ejecución y sus buenos resultados en las métricas frente a discretizaciones lo convierten en una opción viable para clasificar.

El modelo de **Redes Neuronales (MLP)** no logró consolidarse como el clasificador ideal para esta tarea. Si bien alcanzó métricas aceptables, su rendimiento no superó significativamente al de otros modelos más simples, y su tiempo de ejecución fue mayor por su exhaustiva búsqueda del número de capas escondidas. Esto podría explicarse por una combinación de factores: la cantidad de capas y neuronas utilizada, el tamaño del conjunto de entrenamiento y, sobre todo, la sensibilidad de este tipo de modelos al preprocesamiento y escalado de los datos.

Desde un punto de vista estadístico, se aplicaron pruebas formales para sustentar las diferencias observadas entre métodos y configuraciones. Previamente se verificaron los supuestos de normalidad y homogeneidad de varianzas en las distribuciones de las métricas, lo que guió la elección de pruebas paramétricas o no paramétricas. En varios casos los supuestos no se cumplieron, se optó por pruebas no paramétricas: se realizó un ANOVA o Friedman para comparar simultáneamente los métodos, seguido de comparaciones post hoc mediante pruebas de Wilcoxon pareadas o pruebas t pareadas según correspondiera al tipo de métrica. En particular, para modelos como Naive Bayes, Regresión Logística y KNN se hallaron diferencias estadísticamente significativas entre configuraciones, identificando así cuáles combinaciones de discretización y selección condujeron a mejoras reales en las métricas. Por el contrario, en CART y Redes Neuronales muchas de las variaciones entre configuraciones no resultaron significativas estadísticamente, lo que sugiere que ninguna configuración particular logró mejorar sustancialmente su desempeño más allá de la variabilidad inherente. Esto último resulta consistente con los bajos incrementos observados en dichos modelos: sus mejoras con preprocesamiento fueron limitadas

y no lo suficientemente consistentes como para declararse superiores con un nivel de confianza alto.

Tanto desde una perspectiva técnica como interpretativa, estos hallazgos ofrecen información valiosa sobre qué técnicas brindan el mejor rendimiento y bajo qué condiciones.

En resumen, se confirmó que la solución con mejores rendimientos en las métricas de evaluación no radica únicamente en elegir el algoritmo de clasificación más sofisticado, sino en un diseño metodológico integral que abarca un adecuado preprocesamiento. La combinación de discretización supervisada MDLP y selección de características basada en criterios teóricos o en redes bayesianas demostró ser la más efectiva para maximizar las métricas de clasificación. Bajo estas condiciones, algoritmos simples como Naive Bayes alcanzan un rendimiento destacado (muy próximo al de KNN en F1-score y con mejor precision/recall balanceados), a la vez que mantienen costos computacionales bajos, una ventaja crucial para bases de datos de gran escala. El método KNN, si bien lideró en eficacia con dichas transformaciones, conlleva un costo en tiempo que podría limitar su aplicabilidad en la práctica; no obstante, en escenarios donde la precisión sea prioritaria y el tiempo de respuesta no sea crítico, KNN con MDLP será el candidato de mejor desempeño. Por otro lado, el pobre resultado de CART sugiere que los árboles de decisión simples no son adecuados para este dominio con tantas características continuas; podrían requerirse modelos de árbol más complejos para obtener mejoras sustanciales. En cuanto a las redes neuronales, su desempeño similar al de modelos más simples implica que, para problemas de clasificación de retornos financieros con este conjunto de datos y transformaciones, la complejidad adicional de una red profunda no garantiza beneficios proporcionales. Esto destaca la importancia de considerar la interpretabilidad y la eficiencia: modelos como Regresión Logística o Naive Bayes, apoyados por discretización y selección de variables, ofrecen soluciones más transparentes y fáciles de implementar, con rendimientos altos que en muchos casos no difieren significativamente de los obtenidos por una red neuronal.

Un aspecto clave que ayuda a explicar por qué los resultados globales no alcanzaron niveles altos de desempeño radica en la complejidad o la naturaleza de la base de datos utilizada. Al trabajar con 94 características y más de 1.4 millones de instancias, el problema consigue una naturaleza de alta dimensionalidad que incrementa la dificultad de los clasificadores para identificar patrones. Esta situación se intensifica al considerar la categorización de la variable objetivo en cinco clases, ya que la distribución de ejemplos por categoría se vuelve mayor. Los modelos tienden a presentar menor capacidad de generalización y métricas de evaluación más bajas que en el caso de la categorización en tres clases, donde la mayor concentración de datos por clase favorece la detección de relaciones más estables. Tanto la alta dimensionalidad como la complejidad intrínseca de los retornos financieros contribuyen a que el desempeño de los clasificadores no sea comparable al de dominios con menor ruido y por ende, no sea óptimo para una aplicación de la vida real. Sin embargo, estos resultados también se observan en los artículos reportados en el capítulo del Marco Teórico 2.1 (págs. 25-29), donde resuelven el mismo problema o muy parecido, pero usando predicción-regresión. La mayoría de los resultados reportados de esos artículos están por debajo de la consideración de ser buenos predictores. Las métricas reportadas en ellos (*Mean Squared Error, Mean Absolute Error,* R^2) dan resultados muy bajos e inclusive negativos, teniendo la probabilidad de que los modelos se hayan sobreajustado.

Finalmente, al aplicar el proceso de trabajo completo a un conjunto de firmas que han permanecido en el índice S&P 500 durante periodos extensos, se observaron resultados similares a

los obtenidos con los subconjuntos “Top” y “Bot”. Esto sugiere que la metodología propuesta puede generalizarse a otros escenarios históricos del mercado financiero.

En resumen, los resultados obtenidos no son únicamente consecuencia de los algoritmos aplicados, sino del diseño metodológico basado en discretización y selección de características. Este enfoque fue esencial para manejar la complejidad del problema y maximizar la interpretabilidad y desempeño de los modelos.

Capítulo 5

Conclusiones

En esta investigación se abordó el problema de la clasificación supervisada de activos financieros mediante la transformación de características continuas a discretas y la posterior evaluación de distintos algoritmos de aprendizaje supervisado. Se propusieron y analizaron dos categorizaciones de la variable objetivo "Retorno", siendo la configuración de tres clases la más adecuada, ya que permitió una mejor distribución de instancias y un mayor rendimiento de los modelos.

Se aplicaron dos métodos de discretización de características: **K-means** y **MDLP**. Para K-means, se determinaron los valores óptimos de k utilizando el índice de Davies-Bouldin y el índice de Calinski-Harabasz. Posteriormente, se evaluaron cinco algoritmos de clasificación: **Regresión Logística**, **Redes Neuronales**, **K-Vecinos Más Cercanos (KNN)**, **Naive Bayes** y **CART**.

Los resultados muestran que:

1. El mejor desempeño en términos de métricas de evaluación (*accuracy, F1-score, precision y recall*) fue alcanzado por el algoritmo **KNN** con discretización supervisada MDLP. Sin embargo, este modelo presentó el mayor tiempo de ejecución.

2. El clasificador **Naive Bayes** en conjunto con la discretización de MDLP, ofreció un equilibrio notable entre desempeño y eficiencia computacional, destacándose como una opción viable cuando se requiere rapidez sin sacrificar demasiada precisión.

3. La reducción del conjunto de datos a las 1000 firmas con peor rendimiento histórico mostró resultados más favorables que el conjunto completo, con mejoras en métricas y tiempos de ejecución. Siendo una opción viable para aplicaciones reales que expliquen el mercado financiero sin tener en cuenta todas las firmas y disminuyendo el tiempo de ejecución.

4. La aplicación de reducción de dimensionalidad mediante **PCA** no resultó eficaz para la mayoría de los modelos, ya que afectó negativamente las métricas de evaluación, salvo en ciertos casos como Regresión Logística y Redes Neuronales.

5. En cambio, la selección de características mediante métodos teóricos (Gu, 2021 [35]) y redes bayesianas (MDL-FS [22]) fue altamente efectiva, reduciendo significativamente los tiempos de ejecución y, en muchos casos, mejorando el desempeño de los modelos.

Además, se aplicó todo el proceso propuesto al subconjunto de firmas que han prevalecido desde la creación del índice S&P 500 hasta la fecha actual, encontrando resultados muy similares a los obtenidos con el conjunto original de los últimos 20 años. Con la investigación realizada fue posible responder las preguntas planteadas en un inicio.

La categorización del retorno, al agruparlo en tres clases balanceadas, simplificó la detección de patrones y permitió aplicar modelos de clasificación con métricas más estables y precisas, al tiempo que evitó el sesgo derivado de clases poco representadas. Este preprocesamiento se complementó con la discretización de las variables predictoras mediante K-means y MDLP. Mientras K-means ofreció rapidez y versatilidad, MDLP capturó mejor la estructura de los datos y rindió especialmente bien en modelos probabilísticos como Naive Bayes y en métodos basados en vecinos como KNN.

La combinación de estas transformaciones se tradujo en mejoras notables de *accuracy*, *precision*, *recall* y *F1-score* en la mayoría de los casos; el aumento de rendimiento fue más evidente en Naive Bayes y KNN, aunque este último implicó un mayor costo computacional. Optar por un enfoque de clasificación en lugar de regresión también contribuyó a reducir el sobreajuste y a disponer de métricas de evaluación más directas. Finalmente, la selección de características basada en redes bayesianas y teoría resultaron más eficiente que PCA, pues preservaron las relaciones entre variables y mantuvo, o incluso mejoró, el desempeño del modelo con un menor esfuerzo computacional.

Los resultados obtenidos respaldan las hipótesis planteadas en esta investigación. En particular, la aplicación de discretizadores como K-means y MDLP permitió mejorar el desempeño de los modelos de clasificación en comparación con características sin transformar. Asimismo, los métodos de selección de características, especialmente los basados en teoría y redes bayesianas, redujeron de forma significativa el tiempo de ejecución sin comprometer e incluso mejorando en varios casos la calidad de las métricas de evaluación.

Cabe destacar que los resultados obtenidos, aunque consistentes, se mantuvieron en niveles moderados-bajos de desempeño si se comparan con lo esperado de un buen clasificador. Una de las principales razones es la naturaleza de la base de datos, caracterizada por su alta dimensionalidad (94 características) y la complejidad misma de los retornos financieros. Estos factores dificultan la identificación de patrones estables. Además, la categorización en cinco clases resultó menos efectiva que en tres, pues la dispersión de instancias por categoría incrementó el desbalance y redujo la densidad de datos disponibles para el entrenamiento, afectando negativamente la capacidad de generalización de los modelos. En cambio, la clasificación en tres clases favoreció una distribución más balanceada, lo que explicó el mejor rendimiento obtenido bajo esta configuración.

En conjunto, los resultados obtenidos permiten reafirmar las contribuciones planteadas inicialmente, no sólo en el ámbito metodológico, sino también en términos de su aplicabilidad económica y práctica. Se comprobó que reformular el problema como una tarea de clasificación, mediante la categorización del retorno, facilita el uso de modelos y métricas más interpretables, lo cual mejora la toma de decisiones en escenarios financieros reales. Además, la combinación de discretización supervisada (MDLP) con selección de características basada en conocimiento teórico o en estructuras aprendidas con redes bayesianas permitió alcanzar niveles de rendimiento comparables, e incluso superiores, a los de modelos más complejos como las redes neuronales, pero con una reducción significativa en los tiempos de ejecución.

Esta eficiencia metodológica no sólo representa una ventaja computacional, además, ofrece una oportunidad de implementación práctica para instituciones financieras que requieren soluciones rápidas, escalables y transparentes. La metodología propuesta, al haber sido validada sobre una base de datos real y replicada en diferentes subconjuntos (Top, Bot y firmas históricas del S&P 500), demuestra su adaptabilidad y potencial de generalización hacia otros contextos financieros. Además de contribuir al cuerpo teórico de la inteligencia artificial aplicada a finanzas, también ofrece herramientas concretas, validadas y eficientes para el diseño de estrategias de inversión en entornos de alta dimensionalidad y complejidad estructural.

Este trabajo puede ampliarse en diversas direcciones. La investigación se limitó a dos métodos de discretización y cinco clasificadores supervisados. Futuros trabajos relacionados con esta metodología podrían explorar discretizadores como **ChiMerge** o **CAIM**, así como métodos de clasificación más recientes, con el fin de confirmar y ampliar los hallazgos presentados. Por un lado, conviene evaluar discretizadores híbridos o basados en heurísticas; por otro, estudiar cómo técnicas de *AutoML* o aprendizaje profundo pueden integrarse con una etapa previa de discretización para tareas de predicción en mercados financieros. Otra línea de trabajo es analizar las características de los activos mediante agrupación en clústeres, por ejemplo, según su periodicidad de reporte (mensual, bimestral, trimestral, anual) o la entidad que genera las características (Compustat o CRSP). Finalmente, aplicar este enfoque a otros mercados o clases de activos no considerados en esta tesis ampliaría la aplicabilidad de los resultados.

Como propuesta de continuidad en el doctorado del IIIA, se plantea desarrollar un modelo difusivo para generar datos sintéticos que simulen la distribución del mercado financiero y de sus activos; posteriormente, incorporar estos datos a un *transformer* tabular con el objetivo de entrenar un clasificador supervisado que permita predecir el comportamiento de los activos analizados en esta investigación.

Esta tesis ofrece una propuesta integral para la clasificación de retornos financieros mediante la discretización y selección de características, aplicadas a algoritmos de aprendizaje supervisado. Los hallazgos evidencian que la transformación previa de las características, junto con su selección, mejora la eficiencia y la precisión del análisis de activos financieros, constituyendo una herramienta útil para el análisis cuantitativo y cualitativo en contextos de alta dimensionalidad.

Bibliografía

[1] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015.

[2] Peter Auer, Robert C. Holte y Wolfgang Maass. "Theory and applications of agnostic PAC-learning with small decision trees". En: *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*. ICML'95. Tahoe City, California, USA: Morgan Kaufmann Publishers Inc., 1995, 21–29. ISBN: 1558603778.

[3] Matteo Bagnara. "Asset Pricing and Machine Learning: A critical review". En: *Journal of Economic Surveys* 38.1 (2024), págs. 27-56. DOI: https://doi.org/10.1111/joes.12532.

[4] Ravi Bansal y Amir Yaron. "Risks for the Long Run: A Potential Resolution of Asset Pricing Puzzles". En: *The Journal of Finance* 59.4 (2004), págs. 1481-1509. DOI: https://doi.org/10.1111/j.1540-6261.2004.00670.x.

[5] Rolf W. Banz. "The relationship between return and market value of common stocks". En: *Journal of Financial Economics* 9.1 (1981), págs. 3-18. ISSN: 0304-405X. DOI: https://doi.org/10.1016/0304-405X(81)90018-0.

[6] BBVA. *ROI: ¿Qué es el retorno de la inversión y cuál es su fórmula?* https://www.bbva.com/es/salud-financiera/roi-que-es-el-retorno-de-la-inversion-y-cual-es-su-formula/. [Accedido: 09-01-2025]. s.f.

[7] Stefano Beretta et al. "Learning the Structure of Bayesian Networks: A Quantitative Assessment of the Effect of Different Algorithmic Schemes". En: *Complexity* 2018.1 (2018), pág. 1591878.

[8] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738.

[9] Fisher Black, Michael C. Jensen y Myron Scholes. "The Capital Asset Pricing Model: Some Empirical Tests". En: *Capital Markets: Asset Pricing & Valuation* (1972).

[10] Z. Bodie, A. Kane y A.J. Marcus. *Principios de inversiones*. McGraw-Hill Interamericana de España S.L., 2004. ISBN: 978-84-481-4075-5. URL: https://books.google.com.mx/books?id=uIp2PgAACAAJ.

[11] L. Breiman et al. *Classification and Regression Trees*. Wadsworth International Group, 1984.

[12] Morton B. Brown y Alan B. Forsythe. "The ANOVA and multiple comparisons for data with heterogeneous variances". En: *Biometrics* 30.4 (1974), págs. 719-724. DOI: 10.2307/2529238.

[13] Tadeusz Caliński y Harabasz JA. "A Dendrite Method for Cluster Analysis". En: *Communications in Statistics - Theory and Methods* 3 (ene. de 1974), págs. 1-27. DOI: 10.1080/03610927408827101.

[14] J. Catlett. "On changing continuous attributes into ordered discrete attributes". En: *Proceedings of the 5th European Conference on European Working Session on Learning*. EWSL'91. Porto, Portugal: Springer-Verlag, 1991, 164–178. ISBN: 354053816X. DOI: 10.1007/BFb0017012. URL: https://doi.org/10.1007/BFb0017012.

[15] OpenAI ChatGPT. *Asistencia con el análisis de datos financieros*. Explicación sobre el cálculo e interpretación de las métricas baspread e idiovol para análisis de activos financieros. 2024. URL: https://openai.com/chatgpt.

[16] D.M. Chickering, C. Meek y R. Rounthwaite. "Efficient determination of dynamic split points in a decision tree". En: *Proceedings 2001 IEEE International Conference on Data Mining*. 2001, págs. 91-98. DOI: 10.1109/ICDM.2001.989505.

[17] Guillaume Coqueret. "Machine Learning in Finance: From Theory to Practice". En: *Quantitative Finance* 21.1 (2021), págs. 9-10. DOI: 10.1080/14697688.2020.1828609.

[18] D. R. Cox. "The Regression Analysis of Binary Sequences". En: *Journal of the Royal Statistical Society. Series B (Methodological)* 20.2 (1958), págs. 215-242. ISSN: 00359246. (Visitado 30-10-2024).

[19] David L. Davies y Donald W. Bouldin. "A Cluster Separation Measure". En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.2 (1979), págs. 224-227. DOI: 10.1109/TPAMI.1979.4766909.

[20] Aaron Defazio, Francis R. Bach y Simon Lacoste-Julien. "SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives". En: *CoRR* abs/1407.0202 (2014). arXiv: 1407.0202.

[21] James Dougherty, Ron Kohavi y Mehran Sahami. "Supervised and Unsupervised Discretization of Continuous Features". En: *Machine Learning Proceedings 1995*. Ed. por Armand Prieditis y Stuart Russell. San Francisco (CA): Morgan Kaufmann, 1995, págs. 194-202. ISBN: 978-1-55860-377-6. DOI: https://doi.org/10.1016/B978-1-55860-377-6.50032-3.

[22] Madalina Drugan y Marco Wiering. "Feature selection for Bayesian network classifiers using the MDL-FS score". En: *International Journal of Approximate Reasoning* 51 (jul. de 2010), págs. 695-717. DOI: 10.1016/j.ijar.2010.02.001.

[23] Eugene F. Fama y Kenneth R. French. "A five-factor asset pricing model". En: *Journal of Financial Economics* 116.1 (2015), págs. 1-22. ISSN: 0304-405X. DOI: https://doi.org/10.1016/j.jfineco.2014.10.010.

[24] Eugene F. Fama y Kenneth R. French. "Common risk factors in the returns on stocks and bonds". En: *Journal of Financial Economics* 33.1 (1993), págs. 3-56.

[25] Eugene F. Fama y Kenneth R. French. "The Cross-Section of Expected Stock Returns". En: *The Journal of Finance* 47.2 (1992), págs. 427-465. DOI: https://doi.org/10.1111/j.1540-6261.1992.tb04398.x.

[26] Eugene F. Fama y James D. MacBeth. "Risk, Return, and Equilibrium: Empirical Tests". En: *Journal of Political Economy* 81.3 (1973), págs. 607-636. DOI: 10.1086/260061.

[27] Usama M. Fayyad y Keki B. Irani. "Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning". En: *International Joint Conference on Artificial Intelligence*. 1993.

[28] Milton Friedman. "The use of ranks to avoid the assumption of normality implicit in the analysis of variance". En: *Journal of the American Statistical Association* 32.200 (1937), págs. 675-701. DOI: 10.1080/01621459.1937.10503522.

[29] Milton Friedman y L. J. Savage. "The Utility Analysis of Choices Involving Risk". En: *Journal of Political Economy* 56.4 (1948), págs. 279-304. DOI: 10.1086/256692.

[30] Salvador García et al. "A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning". En: *IEEE Transactions on Knowledge and Data Engineering* 25.4 (2013), págs. 734-750. DOI: 10.1109/TKDE.2012.35.

[31] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 2nd. Sebastopol, CA: O'Reilly Media, 2019. ISBN: 978-1492032649.

[32] Stefano Giglio, Bryan Kelly y Dacheng Xiu. "Factor Models, Machine Learning, and Asset Pricing". En: *Annual Review of Financial Economics* 14.1 (2022), págs. 337-368. DOI: 10.1146/annurev-financial.

[33] Gene H. Golub y Charles F. Van Loan. *Matrix Computations*. Third. The Johns Hopkins University Press, 1996.

[34] Jeremiah Green, John R. M. Hand y X. Frank Zhang. "The Characteristics that Provide Independent Information about Average U.S. Monthly Stock Returns". En: *The Review of Financial Studies* 30.12 (2017), págs. 4389-4436.

[35] Shihao Gu, Bryan Kelly y Dacheng Xiu. "Autoencoder asset pricing models". En: *Journal of Econometrics* 222.1, Part B (2021). Annals Issue: Financial Econometrics in the Age of the Digital Economy, págs. 429-450. ISSN: 0304-4076. DOI: https://doi.org/10.1016/j.jeconom.2020.07.009.

[36] Shihao Gu, Bryan Kelly y Dacheng Xiu. "Empirical Asset Pricing via Machine Learning". En: *The Review of Financial Studies* 33.5 (feb. de 2020), págs. 2223-2273. ISSN: 0893-9454. DOI: 10.1093/rfs/hhaa009.

[37] Charles R. Harris et al. "Array programming with NumPy". En: *Nature* 585.7825 (sep. de 2020), págs. 357-362. DOI: 10.1038/s41586-020-2649-2.

[38] Trevor Hastie, Robert Tibshirani y Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. 2.ª ed. Springer, 2009.

[39] Harold Hotelling. "Analysis of a complex of statistical variables into principal components." En: *Journal of Educational Psychology* 24 (1933), págs. 498-520.

[40] J. D. Hunter. "Matplotlib: A 2D graphics environment". En: *Computing in Science & Engineering* 9.3 (2007), págs. 90-95. DOI: 10.1109/MCSE.2007.55.

[41] Gareth James et al. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013.

[42] Barasimhan Jegadeesh y Sheridan Titman. "Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency". En: *The Journal of Finance* 48.1 (1993), págs. 65-91. DOI: https://doi.org/10.1111/j.1540-6261.1993.tb04702.x.

[43] Ruoming Jin, Yuri Breitbart y Chibuike Muoh. "Data Discretization Unification". En: *Knowledge and Information Systems - KAIS* 19 (nov. de 2007), págs. 183-192. DOI: 10.1109/ICDM.2007.35.

[44] John D. Kelleher, Brian Mac Namee y Aoife D'Arcy. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies*. The MIT Press, 2015. ISBN: 0262029448.

[45] Brian Kelly y Seth Pruitt. "Market Expectations in the Cross-Section of Present Values". En: *The Journal of Finance* 68.5 (2013), págs. 1721-1756. DOI: https://doi.org/10.1111/jofi.12060.

[46] Brian Kelly y Seth Pruitt. "The three-pass regression filter: A new approach to forecasting using many predictors". En: *Journal of Econometrics* 186.2 (2015). High Dimensional Problems in Econometrics, págs. 294-316. ISSN: 0304-4076. DOI: https://doi.org/10.1016/j.jeconom.2015.02.011.

[47] Bryan T. Kelly, Seth Pruitt y Yinan Su. "Characteristics are covariances: A unified model of risk and return". En: *Journal of Financial Economics* 134.3 (2019), págs. 501-524.

[48] Diederik P. Kingma y Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[49] Neville Kenneth Kitson et al. "A survey of Bayesian Network structure learning". En: *Artificial Intelligence Review* 56.8 (2023), págs. 8721-8814. ISSN: 1573-7462. DOI: 10.1007/s10462-022-10351-w.

[50] Ron Kohavi y Mehran Sahami. "Error-based and entropy-based discretization of continuous features". En: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD'96. Portland, Oregon: AAAI Press, 1996, 114–119.

[51] Serhiy Kozak, Stefan Nagel y Shrihari Santosh. "Shrinking the cross-section". En: *Journal of Financial Economics* 135.2 (2020), págs. 271-292. ISSN: 0304-405X. DOI: https://doi.org/10.1016/j.jfineco.2019.06.008.

[52] John Lintner. "The Valuation of Risk Assets and the Selection of Risky Investments in Stock Portfolios and Capital Budgets". En: *The Review of Economics and Statistics* 47.1 (1965), págs. 13-37. ISSN: 00346535, 15309142. (Visitado 25-09-2024).

[53] Huan Liu et al. "Discretization: An Enabling Technique". En: *Data Min. Knowl. Discov.* 6 (oct. de 2002), págs. 393-423. DOI: 10.1023/A:1016304305535.

[54] J. MacQueen. *Some methods for classification and analysis of multivariate observations*. English. Proc. 5th Berkeley Symp. Math. Stat. Probab., Univ. Calif. 1965/66, 1, 281-297 (1967). 1967.

[55] Harry Markowitz. "Portfolio Selection". En: *Journal of Finance* 7.1 (1952), págs. 77-91. DOI: j.1540-6261.1952.tb01525..

[56] Harry Markowitz. "The Utility of Wealth". En: *Journal of Political Economy* 60.2 (1952), págs. 151-158. ISSN: 00223808, 1537534X. (Visitado 25-09-2024).

[57] Stephen Marsland. *Machine Learning - An Algorithmic Perspective.* Chapman and Hall / CRC machine learning and pattern recognition series. CRC Press, 2009, págs. I-XVI, 1-390. ISBN: 978-1-4200-6718-7.

[58] Wes McKinney et al. "Data structures for statistical computing in python". En: *Proceedings of the 9th Python in Science Conference*. Vol. 445. Austin, TX. 2010, págs. 51-56.

[59] Robert C. Merton. "An Intertemporal Capital Asset Pricing Model". En: *Econometrica* 41.5 (1973), págs. 867-887. ISSN: 00129682, 14680262.

[60] Robert C. Merton. "On estimating the expected return on the market: An exploratory investigation". En: *Journal of Financial Economics* 8.4 (1980), págs. 323-361. ISSN: 0304-405X. DOI: https://doi.org/10.1016/0304-405X(80)90007-0.

[61] Tom M Mitchell. *Machine learning*. Vol. 1. 9. McGraw-hill New York, 1997.

[62] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN: 0262018020.

[63] Vinod Nair y Geoffrey E. Hinton. "Rectified linear units improve restricted boltzmann machines". En: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. ICML'10. Haifa, Israel: Omnipress, 2010, 807–814. ISBN: 9781605589077.

[64] Lubos Pastor. "Portfolio Selection and Asset Pricing Models". En: *Journal of Finance* 55.1 (2000), págs. 179-223.

[65] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". En: *Journal of Machine Learning Research* 12 (2011), págs. 2825-2830.

[66] Bernhard Pfahringer. "Compression-based discretization of continuous attributes". En: *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*. ICML'95. Tahoe City, California, USA: Morgan Kaufmann Publishers Inc., 1995, 456–463. ISBN: 1558603778.

[67] Marcos Lopez de Prado. *Advances in Financial Machine Learning*. 1st. Wiley Publishing, 2018. ISBN: 1119482089.

[68] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2021.

[69] Ban Rosenberg, Kenneth Reid y Ronald Lanstein. "Persuasive Evidence of Market Inefficiency (Spring 1985)". En: *The Best of The Journal of Portfolio Management*. Princeton: Princeton University Press, 1998, págs. 48-55. ISBN: 9781400829408. DOI: doi: 10.1515/9781400829408-007.

[70] Stephen A. Ross. “The arbitrage theory of capital asset pricing”. En: *Journal of Economic Theory* 13.3 (1976), págs. 341-360.

[71] Stuart Russell y Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3.ª ed. Prentice Hall, 2010.

[72] S. S. Shapiro y M. B. Wilk. “An analysis of variance test for normality (complete samples)”. En: *Biometrika* 52.3-4 (1965), págs. 591-611. DOI: 10.1093/biomet/52.3-4.591.

[73] William F. Sharpe. “Capital Asset Prices: A Theory Of Market Equilibrium Under Conditions Of Risk”. En: *Journal of Finance* 19.3 (1964), págs. 425-442. DOI: j.1540-6261.1964.tb02865..

[74] C. Shearer. “The CRISP-DM Model: The New Blueprint for Data Mining.” En: *Journal of Data Warehousing, 5, 13-22.* (2000).

[75] David J. Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures*. 2nd. Boca Raton, FL: CRC Press, 1997. ISBN: 9780849339734.

[76] Michael E. Tipping y Christopher M. Bishop. “Mixtures of Probabilistic Principal Component Analyzers”. En: *Neural Computation* 11.2 (feb. de 1999), págs. 443-482. ISSN: 0899-7667. DOI: 10.1162/089976699300016728.

[77] Guido Van Rossum y Fred L. Drake. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN: 1441412697.

[78] P. Virtanen et al. *SciPy library: scipy.stats.wilcoxon — Wilcoxon signed-rank test*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wilcoxon.html. Accessed June 17, 2025. 2023.

[79] Michael L. Waskom. “seaborn: statistical data visualization”. En: *Journal of Open Source Software* 6.60 (2021), pág. 3021. DOI: 10.21105/joss.03021.

[80] Wharton Research Data Services. *Wharton Research Data Services (WRDS)*. https://wrds-www.wharton.upenn.edu/. University of Pennsylvania, The Wharton School. Accessed: 2023-12-01. 2023.

Apéndice A

Anexos

Tabla A.1: Lista de las características que se usaron para el estudio empírico. La información fue colectada de [34]

Acronym & Data Source	**Author(s)**	**Date, Journal & Frequency**	**Definition of the characteristic-based anomaly variable**
absacc / Compustat	Bandyopadhyay, Huang, and Wirjanto	2010, WP / Annual	Absolute value of acc
acc / Compustat	Sloan	1996, TAR / Annual	Annual income before extraordinary items (ib) minus operating cash flows (oancf) divided by average total assets (at); if oancf is missing then set to change in act - change in che - change in lct + change in dlc + change in txp-dp
aeavol / Compustat + CRSP	Lerman, Livnat, and Mendenhall	2008, WP / Quarterly	Average daily trading volume (vol) for 3 days around earnings announcement minus average daily volume for 1-month ending 2 weeks before earnings announcement divided by 1-month average daily volume. Earnings announcement day from Compustat quarterly (rdq)
age / Compustat	Jiang, Lee, and Zhang	2005, RAS / Annual	Number of years since first Compustat coverage
agr / Compustat	Cooper, Gulen, and Schill	2008, JF / Annual	Annual percent change in total assets (at)
baspread / CRSP	Amihud and Mendelson	1989, JF / Monthly	Monthly average of daily bid-ask spread divided by average of daily spread

beta / CRSP	Fama and MacBeth	1973, JPE / Monthly	Estimated market beta from weekly returns and equal weighted market returns for 3 years ending month t-1 with at least 52 weeks of returns
betasq / CRSP	Fama and MacBeth	1973, JPE / Monthly	Market beta squared
bm / Compustat + CRSP	Rosenberg, Reid, and Lanstein	1985, JPM / Annual	Book value of equity (ceq) divided by end of fiscal year-end market capitalization
bm_ia / Compustat + CRSP	Asness, Porter, and Stevens	2000, WP / Annual	Industry adjusted book-to-market ratio
cash / Compustat	Palazzo	2012, JFE / Quarterly	Cash and cash equivalents divided by average total assets
cashdebt / Compustat	Ou and Penman	1989, JAE / Annual	Earnings before depreciation and extraordinary items (ib+dp) divided by avg. total liabilities (lt)
cashpr / Compustat	Chandrashekar and Rao	2009, WP / Annual	Fiscal year-end market capitalization plus long-term debt (dltt) minus total assets (at) divided by cash and cash equivalents (che)
cfp / Compustat	Desai, Rajgopal, and Venkatachalam	2004, TAR / Annual	Operating cash flows divided by fiscal-year-end market capitalization
cfp_ia / Compustat	Asness, Porter, and Stevens	2000, WP / Annual	Industry adjusted cfp
chatoia / Compustat	Soliman	2008, TAR / Annual	2-digit SIC - fiscal-year mean-adjusted change in sales (saleq) divided by average total assets (at)
chcsho / Compustat	Pontiff and Woodgate	2008, JF / Annual	Annual percent change in shares outstanding (csho)
chempia / Compustat	Asness, Porter, and Stevens	1994, WP / Annual	Industry-adjusted change in number of employees
chinv / Compustat	Thomas and Zhang	2002, RAS / Annual	Change in inventory (inv) scaled by average total assets (at)
chmom / CRSP	Gettleman and Marks	2006, WP / Monthly	Cumulative returns from months t-6 to t-1 minus months t-12 to t-7
chpmia / Compustat	Soliman	2008, TAR / Annual	2-digit SIC - fiscal-year mean adjusted change in income before extraordinary items (ib) divided by sales (sale)
chtx / Compustat	Thomas and Zhang	2011, JAR / Quarterly	Percent change in total taxes (txtq) from quarter-4 to quarter-1

cinvest / Compustat	Titman, Wei, and Xie	2004, JFQA / Quarterly	Change over one quarter in net PP&E (ppentq) divided by sales (saleq). Average of this variable for prior 3 quarters; if saleq = 0, then scale by 0.01
convind / Compustat	Valta	2016, JFQA / Annual	An indicator equal to 1 if company has convertible debt obligations
currat / Compustat	Ou and Penman	1989, JAE / Annual	Current assets / current liabilities
depr / Compustat	Holthausen and Larcker	1992, JAE / Annual	Depreciation divided by PP&E
divi / CRSP	Michaely, Thaler, and Womack	1995, JF / Monthly	An indicator variable equal to 1 if company pays dividends but did not cut them
divo / Compustat	Michaely, Thaler, and Womack	1995, JF / Annual	An indicator variable equal to 1 if company does not pay dividends but did in prior year
dolvol / CRSP	Chordia, Subrahmanyam, and Anshuman	2001, JFE / Monthly	Natural log of trading volume scaled by share from month t-2
dy / Compustat	Litzenberger and Ramaswamy	1982, JF / Annual	Total dividends (dvt) divided by market capitalization at fiscal year-end
ear / Compustat + CRSP	Kishore et al.	2008, WP / Quarterly	Sum of daily returns in three days around earnings announcement. Earnings announcement from Compustat quarterly file (rdq)
egr / Compustat	Richardson et al.	2005, JAE / Annual	Annual percent change in book equity (ceq)
ep / Compustat	Basu	1977, JF / Annual	Annual income before extraordinary items (ib) divided by end of fiscal-year-end market cap
gma / Compustat	Novy-Marx	2013, JFE / Annual	Revenues (revt) minus cost of goods sold (cogs) divided by lagged total assets (at)
grCAPX / Compustat	Anderson and Garcia-Feijoo	2006, JF / Annual	Percent change in capital expenditures from year t-2 to year t
grltnoa / Compustat	Fairfield, Whisenant, and Yohn	2003, TAR / Annual	Growth in long-term net operating assets
herf / Compustat	Hou and Robinson	2006, JF / Annual	2-digit SIC - fiscal-year sales concentration (sum of squared percent of sales in industry for each company)

hire / Compustat	Bazdrech, Belo, and Lin	2014, JPE / Annual	Percent change in number of employees (emp)
idiovol / CRSP	Ali, Hwang, and Trombley	2003, JFE / Monthly	Standard deviation of residuals of weekly returns on weekly equal weighted market returns for 3 years prior to month t-1
ill / CRSP	Amihud	2002, JFM / Monthly	Average of daily absolute returns scaled by average dollar volume
indmom / CRSP	Moskowitz and Grinblatt	1999, JF / Monthly	Equal weighted average industry 12-month returns
invest / Compustat	Chen and Zhang	2010, JF / Annual	Annual change in gross property, plant, and equipment (ppent) + annual change in inventories (invt) all scaled by lagged total assets
lev / Compustat	Bhandari	1988, JF / Annual	Total liabilities (lt) divided by fiscal-year-end market capitalization
lgr / Compustat	Richardson et al.	2005, JAE / Annual	Annual percent change in total liabilities (lt)
maxret / CRSP	Bali, Cackci, and Whitelaw	2011, JFE / Monthly	Maximum daily return from returns during calendar month t-1
mom1m / CRSP	Jegadeesh	1990, JF / Monthly	1-month cumulative returns ending one month before month end
mom6m / CRSP	Jegadeesh and Titman	1993, JF / Monthly	6-month cumulative returns ending t-1
mom12m / CRSP	Jegadeesh	1990, JF / Monthly	12-month cumulative returns ending one month before month end
mom36m / CRSP	Jegadeesh and Titman	1993, JF / Monthly	36-month cumulative returns ending t-1
ms / Compustat	Mohanram	2005, RAS / Quarterly	Sum of 8 indicator variables for fundamental performance
mve1 / CRSP	Banz	1981, JFE / Monthly	Natural log of market capitalization at end of month t-1 (Size)
mve_ia / Compustat	Asness, Porter, and Stevens	2000, WP / Annual	2-digit SIC industry-adjusted fiscal-year-end market capitalization
nincr / Compustat	Barth, Elliott, and Finn	1999, JAR / Quarterly	Number of consecutive quarters (up to eight quarters) with an increase in earnings (ibq) over same quarter in the prior year
operprof / Compustat	Fama and French	2015, JFE / Annual	Revenue minus cost of goods sold - SG&A expense - interest expense divided by lagged common shareholders' equity
orgcap / Compustat	Eisfeldt and Papanikolaou	2013, JF / Annual	Capitalized SG&A expenses

pchcapx_ia / Compustat	Abarbanell and Bushee	1998, TAR / Annual	2-digit SIC - fiscal-year mean-adjusted percent change in capital expenditures (capx)
pchcurrat / Compustat	Ou and Penman	1989, JAE / Annual	Percent change in current ratio (currat)
pchdepr / Compustat	Holthausen and Larcker	1992, JAE / Annual	Percent change in depreciation (dp)
pchgm _pchsale / Compustat	Abarbanell and Bushee	1998, TAR / Annual	Percent change in gross margin (sale - cogs) minus percent change in sales (sale)
pchquick / Compustat	Ou and Penman	1989, JAE / Annual	Percent change in quick ratio
pchsale _pchinvt / Compustat	Abarbanell and Bushee	1998, TAR / Annual	Annual percent change in sales (sale) minus annual percent change in inventory (invt)
pchsale _pchrect / Compustat	Abarbanell and Bushee	1998, TAR / Annual	Annual percent change in sales (sale) minus annual percent change in A/R
pchsale _pchxsga / Compustat	Abarbanell and Bushee	1998, TAR / Annual	Annual percent change in sales (sale) minus annual percent change in SG&A
pchsaleinv / Compustat	OU & Penman	1989, JAE / Annual	Annual percent change in sales to inventory
pctacc / Compustat	Hafzalla, Lundholm, and Van Winkle	2011, TAR / Annual	Same as acc except that the numerator is divided by the absolute value of it; if ib is 0 then set to 0.01 for denominator
pricedelay / CRSP	Hou and Moskowitz	2005, RFS / Monthly	The proportion of variation in weekly returns for 36 months ending in month t explained by 4 lags of weekly market returns incremental to contemporaneous market return
ps / Compustat	Piotroski	2000, JAR / Annual	Sum of 9 indicator variables to form fundamental health score
quick / Compustat	Ou and Penman	1989, JAE / Annual	(current assets - inventory) / current liabilities
rd / Compustat	Eberhart, Maxwell, and Siddique	2004, JF / Annual	An indicator variable equal to 1 if R&D expense as a percent of net sales has an increase greater than 5%

rd_mve / Compustat	Guo, Lev, and Shi	2006, JBFA / Annual	R&D expense divided by end-of-fiscal-year market capitalization
rd_sale / Compustat	Guo, Lev, and Shi	2006, JBFA / Annual	R&D expense divided by sales (sale)
realestate / Compustat	Tuzel	2010, RFS / Annual	Buildings and capitalized leases divided by gross PP&E
retvol / CRSP	Ang et al.	2006, JF / Monthly	Standard deviation of daily returns from month t-1
roaq / Compustat	Balakrishnan, Bartov, and Faurel	2010, JAE / Quarterly	Income before extraordinary items (ibq) divided by one quarter lagged total assets (atq)
roavol / Compustat	Francis et al.	2004, TAR / Quarterly	Standard deviation for 16 quarters of income before extraordinary items divided by average total assets
roeq / Compustat	Hou, Xue, and Zhang	2015, RFS / Quarterly	Earnings before extraordinary items (ib) divided by end-of-fiscal-year common equity (ceq-t+che)
roic / Compustat	Brown and Rowe	2007, WP / Annual	Annual earnings before interest and taxes (ebit) minus nonoperating income (nopi) divided by non-cash enterprise value (ceq-t+lt-che)
rsup / Compustat	Kama	2009, JBFA / Quarterly	Sales from quarter t minus sales from quarter t-4 (saleq) divided by fiscal-quarter-end market capitalization (cshoq * prccq)
salecash / Compustat	Ou and Penman	1989, JAE / Annual	Annual sales divided by cash and cash equivalents
saleinv / Compustat	Ou and Penman	1989, JAE / Annual	Annual sales divided by total inventory
salerec / Compustat	Ou and Penman	1989, JAE / Annual	Annual sales divided by accounts receivable
secured / Compustat	Valta	2016, JFQA / Annual	Total liability scaled by secured debt obligations
securedind / Compustat	Valta	2016, JFQA / Annual	Secured debt indicator
sgr / Compustat	Lakonishok, Shleifer, and Vishny	1994, JF / Annual	Annual percent change in sales (sale)

sin / Compustat	Hong	2009, JFE / Annual	An indicator variable equal to 1 if a company's primary industry classification is in sin industries (alcohol, tobacco, gambling, etc.)
sp / Compustat	Barbee, Mukherji, and Raines	1996, FAJ / Annual	Annual revenue (sale) divided by fiscal-year-end market capitalization
std_dolvol / CRSP	Chordia, Subrahmanyam, and Anshuman	2001, JFE / Monthly	Monthly standard deviation of daily dollar trading volume
std_turn / CRSP	Chordia, Subrahmanyam, and Anshuman	2001, JFE / Monthly	Monthly standard deviation of daily share turnover
stdacc / Compustat	Bandyopadhyay, Huang, and Wirjanto	2010, WP / Quarterly	Standard deviation for 16 quarters of accruals (acc measured with quarterly Compustat) scaled by sales; if saleq = 0, then scale by 0.01
stdcf / Compustat	Huang	2009, JEF / Quarterly	Standard deviation for 16 quarters of cash flows divided by sales (saleq); if saleq = 0, then scale by 0.01. Cash flows defined as ibq minus quarterly accruals
tang / Compustat	Almeida and Campello	2007, RFS / Annual	Cash holdings + 0.715 * receivables + 0.547 * inventory + 0.535 * PPE / total assets
tb / Compustat	Lev and Nissim	2004, TAR / Annual	Tax income, calculated from current tax expense divided by maximum federal tax rate, divided by income before extraordinary items
turn / CRSP	Datar, Naik, and Radcliffe	1998, JFM / Monthly	Average monthly trading volume for most recent 3 months scaled by number of shares outstanding in current month
zerotrade / CRSP	Liu	2006, JFE / Monthly	Turnover weighted number of zero trading days for most recent 1 month