Universidad Veracruzana

Algoritmo robusto de identificación de cúmulos en bases de datos para la obtención de variables escondidas a través de redes bayesianas.

TESIS

Presenta:

L.I. Niels Martínez Guevara

Para obtener el grado de:

Maestro en inteligencia artificial.

Director de tesis:

Dr. Nicandro cruz Ramírez

Xalapa, Ver - 24 de enero de 2018

Dedicatoria

En primer lugar a Dios, a quién dedico todo mi trabajo y esfuerzo.

A mis padres quienes me apoyaron y me instruyeron con sus palabras de aliento y orientación para proseguir en mi vida académica y desarrollarme como individuo.

A mi hermano, quién siempre me ha ayudado en momentos de dificultad.

A mis amigos con quienes he compartido múltiples risas y en ellos he encontrado apoyo incondicional.

A Daniel Garcia(q.d.p) y Joaquín Ramos(q.d.p) qué a través de sus vidas me mostraron una perspectiva distinta del mundo.

A Carolina por todo tu amor y apoyo.

Agradecimientos

En primer lugar a Dios, por darme fortaleza y sabiduría para llevar a cabo este gran hito en mi vida.

Agradezco a CONACYT por su apoyo en el cumplimiento de esta gran meta académica en mi vida.

Al CIIA por brindarme todas las herramientas para alcanzar estas metas.

Al Dr. Nicandro Cruz Ramírez, por inspirarme a través de su vida a tomar el camino de la investigación y docencia.

A todos mis mentores, que durante este programa de estudios me retaron y apoyaron para crecer académicamente.

Resumen

En este trabajo se expondrá una variante del algoritmo "Selección de Modelo a Través de la Maximización de la Esperanza"(o MS-EM por sus siglas en inglés). Este algoritmo es una propuesta para obtener una variable oculta en una base de datos a través de modelos bayesianos, la variante propuesta a lo largo de este documento se refiere a la obtención de cúmulos, utilizando una versión robusta del algoritmo de maximización de la esperanza, que consiste en la modificación de su estado inicial para obtener un mejor ajuste en los cúmulos.

Durante este documento utilizaremos la palabra de "variable oculta" o "latente" a una variable X que no se encuentra presente en una base de datos. La peculiaridad de este algoritmo es que propone encontrar una variable escondida y una red bayesiana que explica la presencia de la variable. A diferencia de otros algoritmos de identificación de cúmulos MS-EM basa su función de maximización de la verosimilitud de una red bayesiana en respecto a los datos. En este trabajo se pretende atacar dos problemáticas, la primera es el cálculo de una variable latente y la segunda es determinar el número de posibles valores que puede tomar dicha variable. Para resolver esto se realizó una versión robusta del algoritmo de maximización de la esperanza (REM por sus siglas en ingles), este algoritmo que se propone en la literatura es una solución flexible para encontrar el número de cúmulos (o clusters) que puede tener cada variable.

Índice general

Índice de figuras

Capítulo 1

Introducción

Dentro del área de aprendizaje automático existen tres vertientes las cuales son el aprendizaje supervisado, no supervisado y aprendizaje por refuerzo. Para este trabajo nos enfocaremos en los dos primeros casos. Primeramente en el aprendizaje supervisado para la construcción de una red bayesiana sin restricciones y en el aprendizaje no supervisado (o detección de cúmulos) para calcular una variable no observada (o latente) en la base de datos. Las redes bayesianas nos proporcionan información de cómo se comportan los datos entre si, es decir, las relaciones entre las variables de una base de datos.

Supongamos que tenemos una variable latente que tiene una influencia grande sobre un conjunto de variables de una base de datos ¿Será posible obtener sus valores? Por ejemplo, supongamos unos datos que reflejan los tipos de compradores de una agencia de automóviles y dentro de los atributos se encuentra ausente la variable estatus socio-económico, sin embargo, este valor se puede obtener a partir del modelo del vehículo y los años de financiamiento ya que a un mayor precio y menor tiempo nos indica que la persona tiene un estatus alto, en caso contrario si el precio del vehículo es menor y mayor el tiempo de financiamiento nos indicará que su estatus es bajo.

Este es un ejemplo de las dependencias que pueden tener las variables y que para una determinada combinación de valores se puede agrupar en una variable que tenga sentido en el caso de estudio, sin embargo, encontrar estas relaciones no es nada sencillo y definir los valores de una variable latente suele ser muy costoso y en muchas ocasiones no se puede tener la certeza de cuál es la variable que se está calculando [10].

Algunas de las técnicas más fiables es que un experto pueda modelar una red bayesiana a partir de los datos de un fenómeno, incluyendo la variable latente y su posible relación con las otras variables, pero desgraciadamente esta opción se ha vuelto poco viable ya que el proceso de construcción de una red bayesiana a partir de conocimiento del experto es una labor larga y costosa. Es por ello que se han implementado algoritmos de búsqueda que nos arrojan un óptimo local como Hill-climbing, EDAs, K2, algoritmos genéticos, TAN o en otros casos estructuras establecidas como el caso de Naive bayes.

Para la estimación de los valores de una variable latente se ocupan distintos algoritmos de clustering (detección de cúmulos) con diferentes métricas, como EM, K-means, canopy, cobweb, farthestFirst. El número de agrupaciones podrá ser flexible o estático, dependiendo el algoritmo que se utilice.

En este trabajo se propone un algoritmo capaz de encontrar una variable oculta a través de una red bayesiana (o probabilísitca) que explique la relación entre las variables observadas y la obtenida (o latente) a partir de una base de datos, ésto a través de un algoritmo de clustering (EM) para obtener los valores faltantes de una variable y otro de búsqueda para encontrar la red bayesiana que mejor explique los datos.

1.1. Planteamiento del problema

En este documento se plantea encontrar una alternativa para solucionar dos problemáticas que se presentan durante el proceso de obtención de cúmulos o clustering. La primera es ¿Cuántos valores puede obtener una variable latente? , en el caso especial de EM una de sus virtudes también se puede convertir en un gran problema, al ser un algoritmo de soft-clustering su número de cúmulos es dinámico y en su versión tradicional este valor se obtiene aleatoriamente en un rango de valores con una distribución normal entre los valores de $2 \leq c \leq n$ donde c es el número de clusters y n el número de túplas o casos de la base de datos, permitiendo en una primera instancia el caso de cúmulos duplicados.

A partir de una primera experimentación aleatoria el algoritmo ajustará sus valores para poder encontrar los valores de la variable, al ser c un valor aleatorio dentro de un rango, los resultados del algoritmo serán variables y será difícil poder determinar el número de posibles valores que puede tomar dicha variable ya que nos presenta en primer instancia dos panoramas, el primero, siendo la variable latente una variable con dos posibles valores o el otro extremo siendo cada configuración de valores como perteneciente a un clúster. Ésto puede presentarse como un gran problema ya que una vez definido el número inicial de clusters se podrá reducir, sin embargo no se podrá aumentar, alterando el comportamiento del algoritmo y pudiendo generar ruido en su ejecución de este [2].

La segunda es ¿Qué relación tiene la variable latente con la base de datos? Para esta segunda problemática la literatura nos ofrece alternativas como el algoritmo selección del modelo de acuerdo a la maximización de la esperanza (o MS-EM por sus siglas en inglés) el cual es una modificación del algoritmo EM cuya función de verosimilitud (Likelihood) se basan en una red bayesiana que va cambiando conforme se vayan modificando los datos.

Existen diferentes métodos para obtener el número de clusters, uno de ellos es mediante la técnica de k-folds, la cual consiste en dividir la base de datos en k secciones y aplicar EM en cada una de ellas y si el promedio de cada loglikelihood incrementa entonces el número de cluster incrementa en uno, pero este método a pesar de ser efectivo implica un pre-procesamiento que puede llegar a ser muy costoso en conjuntos de datos muy grandes.[2]

1.2. Objetivos

1.2.1. Objetivo general

Desarrollar una versión robusta del algoritmo MS-EM para la obtención de valores de una variable escondida y ver la relación de dicha variable con la base de datos de la que se obtuvo.

1.2.2. Objetivos particulares

- Realizar un análisis del algoritmo MS-EM propuesto en la literatura.
- Desarrollar el algoritmo adaptativo MS-REM.
- Realizar un análisis comparativo entre el algoritmo propuesto en la literatura (MS-EM) con el que se presenta en este documento (MS-REM).
- Analizar los resultados con cada algoritmo respecto MDL y número de clusters.

1.3. Hipótesis

Es posible diseñar un algoritmo robusto para la obtención de los valores de una variable oculta en una base de datos. Basado en técnicas de soft-clustering mediante la búsqueda de una red bayesiana que muestre la relación de la variable con los datos observables.

1.4. Fundamentación

La propuesta de MS-REM es que se puedan obtener los números de clusters de más a menos ajustando su número hasta que sea el adecuado, mientras encuentra una red que maximice el valor de MDL, implementando un algoritmo adaptativo como lo es REM (Maximización de la Esperanza Robusta) el cual va actualizando sus valores y el número de clusters dependiendo de las iteraciones del algoritmo, ajustando sus probabilidades para tener un mejor Log-Likelihood. Por otro lado el algoritmo MS-EM toma la capacidad ver la relación de la variable con el modelo propuesto. De esta forma se pretende implementar una técnica de la obtención del número de clusters y su modelo.

1.5. Metodología

1. Amplia revisión a la literatura referente a la obtención de variables ocultas, EM y métricas de comparación de modelos y conceptos básicos de estadística para la comprensión del tema a desarrollar.

2. Implementación de la versión tradicional del algoritmo EM.

3. Reproducción del algoritmo MS-EM para observar su comportamiento en diferentes conjuntos de datos.

4. Implementación del algoritmo REM, y ver sus diferencias con respecto a la versión original.

5. Diseño e implementación del algoritmo MS-REM.

6. Analizar el comportamiento del algoritmo a través de pruebas de rendimiento con diferentes conjuntos de datos.

7. Realizar un análisis comparativo entre ambos algoritmos con diferentes bases de datos y registrar los cambios significativos entre la versión original y la propuesta.

1.6. Estructura del documento

A continuación se presenta una breve explicación de los temas que conforman este documento:

- **Capítulo 1:** Se introduce al tema central de este documento y se define la hipótesis, los objetivos y la metodología a seguir para llevar a cabo este trabajo de investigación.
- **Capítulo 2:** Se presentan los conceptos esenciales para desarrollar la propuesta, así como los fundamentos teóricos que la sustentan.
- **Capítulo 3:** Se describe el algoritmo MS-EM propuesto por la literatura, así como un análisis de su funcionamiento y su arquitectura.
- **Capítulo 4:** Se describe el algoritmo MS-REM el cual es la propuesta de este documento, así como un análisis de su funcionamiento y su arquitectura.
- **Capítulo 5:** Se detalla los experimentos que se realizaron para comparar los dos algoritmos, en la fase de experimentación se proponen dos tipos de experimentos los primeros son sugeridos por la literatura y los segundos están enfocados a analizar el comportamiento de los algoritmos.

Capítulo 2

Marco téorico y estado del arte

En el siguiente capítulo se desarrollará el conjunto de saberes en los que se sustenta esta propuesta, así como otras alternativas que se han desarrollado para solucionar esta problemática. En primera instancia se describirá el concepto de variable latente, así como su importancia y cómo se obtienen estos valores, el concepto de clusterización y de cómo es posible utilizar estos algoritmos como EM y k-means para obtener variables ocultas y se tocarán algunos conceptos básicos relacionados con dichos temas como MDL, log-likelihood, max-likelihood y distancia euclidiana.

2.1. Variables latentes

Obtener una red bayesiana de un experto puede ser un proceso largo y muy costoso [3], debido a ello surge el aprendizaje automático, algoritmos basados en la probabilidad de Bayes que nos permiten obtener una red a partir de una base de datos, dentro de este proceso de aprendizaje nos encontraremos con cuatro escenarios principales en el cuadro 2.1 [4]:

	Se conoce la estructura de los datos	**No se conoce la red bayesiana**
Datos Completos	Estimación paramétrica estadística.	Optimización discreta sobre la estructura.
Datos Incompletos	Optimización de parámetros.	Optimización de parámetros y estructura.

Cuadro 2.1: Los problemas dentro del aprendizaje.

Dentro de la literatura y teniendo en cuenta los escenarios del cuadro 2.1, el algoritmo MS-EM (Selección de modelo a partir de la maximización de la esperanza) parte del supuesto de que nuestra base de datos se encuentra incompleta y que desconocemos el modelo (red bayesiana) que generó dichos datos, por lo que se encuentra en el peor escenario del proceso de aprendizaje [5].

Para obtener estos valores partimos de la idea que la variable latente esta relacionada con los datos observables, por lo que puede incluir uno o más arcos con ellas, por lo que a partir de la información de la base de datos es que podremos calcular los valores. Sin embargo al desconocer que variable estamos buscando ¿La variable calculada tiene relación con todas las variables o sólo con algunas? Y ¿Con cuáles tiene relación? Es por eso que surge el algoritmo MS-EM ya que no sólo calcula la variable latente, sino que le da un sentido y relación con las otras variables de la base de datos [5].

Para ilustrar lo anterior consideremos el siguiente ejemplo: Supongamos que nos interesa realizar un análisis de marketing y debemos analizar la relación entre dos variables: Publicidad y Ventas, si nos interesa relacionar Ventas con Publicidad tendríamos los siguientes casos $Ventas = True$ y $Publicidad = True$ o $Ventas = True$ y $Publicidad = False$ por lo que deberíamos calcular las siguientes probabilidades: $P(ven|\widehat{pub})$ y $P(ven|\widehat{\bar{pub}})$ para calcular dichas probabilidades es necesario obtenerlas de un experimento aleatorio, en caso de que $P(ven|\widehat{pub}) = P(ven|\widehat{\bar{pub}}) = P(ven)$ puede concluirse que la variable Ventas no esta influenciada por la variable Publicidad, pero en caso contrario es decir $P(ven|\widehat{pub})! = P(ven|\widehat{\bar{pub}})! = P(ven)$ podremos concluir que existe una relación entre dichas variables, por lo general saber que X causa Y nos permite igualar $P(y|\widehat{x})$ con $P(y|x)$ donde $\widehat{x}$ determina los valores que puede tomar X para x. Con lo anterior se ilustra una relación entre variables en una red bayesiana y con ello podemos ver que la variable X afecta a la variable Y, entonces ¿Si solo tuviéramos una variable sería posible obtener los valores de la otra? Mediante técnicas de obtención de variables latentes es posible [6].

Tomando en cuenta lo anterior y suponiendo que conocemos la estructura de los datos, es decir la red bayesiana, lo que se debe hacer es calcular variable escondida y a través de ella buscamos reducir la complejidad de la red (véase figura 2.1 [5]) u obtener una red que explique el fenómeno observado (véase figura 2.2[6]) En esta segunda también nos ayuda a dar un sentido a una base de datos a partir de la obtención de cúmulos a partir de las características sus características [7].

Sea cual sea el caso, se procederá a insertar la nueva variable en la red, la literatura recomienda que a la variable escondida H se le asigne un valor binario y sus valores se asignen de acuerdo a la maximización de la probabilidad de que $h = 1$ ó $h = 0$ con base en un experimento aleatorio [6]. Sin embargo, ¿Cómo saber que sólo dos valores nos ofrecen una buena representación de la variable? Es por ello que se han implementado técnicas de aprendizaje no supervisado para obtener estos valores [2].

Figura 2.1: Reducción de la complejidad de la red bayesiana, al tener menos arcos la complejidad de la red es menor.

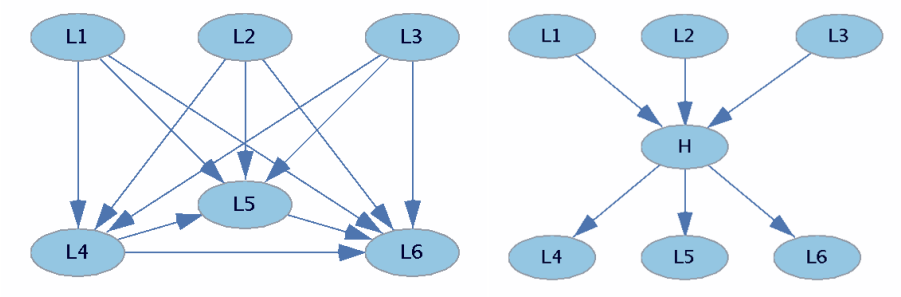
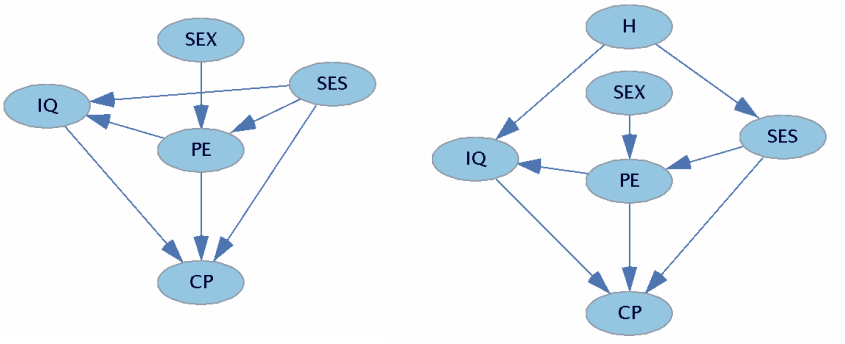


Figura 2.2: El uso de variables escondidas puede ayudar a explicar el fenómeno.



2.2. Obtención de cúmulos.

Cuando nos referimos al proceso de obtención de cúmulos o clustering, nos referimos a la agrupación de los datos a partir de sus características. Estas características se pueden obtener de la proximidad que existe en los datos es decir ¿Qué tan similares son entre sí? O mediante una probabilidad, es decir ¿Qué tan probable es que este dato pertenezca a un determinado cúmulo? Esta función dentro de cualquier algoritmo de clustering se le llamará verosimilitud (o likelihood) y su función es encontrar patrones dentro de los datos y poder categorizarlos en agrupaciones (clusters). Un clúster es un conjunto de túplas (o casos) que tienen en común ciertas características.

Existen dos vertientes dentro del aprendizaje no supervisado: La obtención dinámica de cúmulos (soft-clustering) y la obtención estática de cúmulos (hard-clustering). Esto hace referencia a cómo el algoritmo va a poder identificar el número de clusters que va a encontrar en una base de datos. Si se utiliza un algoritmo de identificación dinámica, el algoritmo se encargará de identificar los clusters a partir de las túplas de la base de datos. En cambio, en un algoritmo de identificación estática, el usuario será el encargado de proporcionar tal información [3]. En esta sección analizaremos dos algoritmos, uno de cada categoría, con la finalidad de analizar su funcionamiento y así destacar sus virtudes e inconvenientes.

2.3. K-means

Este algoritmo de Hard-clustering es uno de los algoritmos más representativos del aprendizaje no supervisado, es un algoritmo iterativo que utiliza la siguiente secuencia de pasos:

- ¿Cuántos clusters se van a encontrar?
- Aleatoriamente se determina el centro de cada clúster.
- Se asigna cada caso de acuerdo a la cercanía con el centro de cada clúster.
- Se encuentra un nuevo centro de cluster de acuerdo los datos.
- Se reubica el centro.
- Se repite el proceso hasta que termina.

De esta forma en cada iteración el algoritmo se va adaptando para encontrar los centroides de cada clúster a partir de los datos, sin embargo, una de las grandes desventajas es que el usuario deberá introducir el número de clusters que desea encontrar y asume clusters esféricos. De esta forma el funcionamiento del algoritmo recae en la configuración del usuario, teniendo así un comportamiento uniforme. [8]

2.4. EM.

El algoritmo de maximización de la esperanza o EM por sus siglas en inglés (expectation maximization) es un algoritmo de soft-clustering iterativo que parte de una configuración de parámetros aleatorios referentes al número de clusters y con cada iteración se producirá un ajuste en estos hasta que se encuentre sus valores óptimos (o sus medias). La versión tradicional de este algoritmo parte con la idea de que el número inicial de clusters es un valor aleatorio mayor a dos y menor al número de casos de la base de datos. Este algoritmo se divide en dos pasos [9]:

- **Esperanza (expectation):** Este paso tiene que ver con la parte probabilística del algoritmo, es decir, cuál es la probabilidad de que un conjunto de datos

pertenezca a un clúster. EM utiliza el teorema de Bayes para calcular dicha probabilidad, como el caso del algoritmo GNB (Gaussian Naive Bayes), sin embargo, es en este apartado donde se introduce el concepto de Maximum Likelihood que se desarrollará más adelante. [5]

- **Maximización (Maximization):** Este paso es el ajuste de parámetros, una vez establecida la esperanza, por lo que realiza el proceso de calcular los nuevos parámetros a partir de los nuevos datos y con ello llevar al algoritmo a la convergencia.[5]

El algoritmo EM se puede describir en la siguiente secuencia de pasos donde ε es un margen de error, S el número de iteración y z las medias de cada clúster:

1. Calcular de forma aleatoria el número de clusters en el siguiente rango $2 \leq c \leq n$ y $\varepsilon > 0$ inicializar $\widehat{z}^{(0)} = (\widehat{z}_1^{(0)}, ..., \widehat{z}_c^{(0)})$ y $S = 1$
2. Calcular parámetros del algoritmo con $\widehat{z}^{(s)}$.
3. Actualizar $\widehat{z}^{(s)}$ con los nuevos parámetros.
4. comparar $\widehat{z}^{(s)}$ con $\widehat{z}^{(s-1)}$ si $\left\| \widehat{z}^{(s)} - \widehat{z}^{(s-1)} \right\| < \varepsilon$ entonces finaliza, en caso contrario regresa al paso 2.

Cuando analizamos el comportamiento del algoritmo EM con sus dos pasos podemos darnos cuenta que EM realmente es un algoritmo de búsqueda estocástico dado que las variables c y $\widehat{z}^{(0)}$ se calculan de forma aleatoria y es por su naturaleza que se ha utilizado para la clusterización. Sin embargo, podemos ver en la literatura que hay diferentes propuestas de este algoritmo para la obtención de redes bayesianas como es el caso de SEM (Structural EM)[10].

EM, a diferencia de K-means, ofrece un panorama flexible en cuanto a la obtención del número de clusters. Sin embargo esto puede llegar a ser una desventaja si se toma en cuenta que parte de un número aleatorio, es por eso que se han publicado algoritmos que intentan atacar esta debilidad en el algoritmo, una de ellas es EM robusto o REM por sus siglas en inglés (Robust expectation maximization), el cual

siempre inicia en el peor de los casos, es decir, las medias de los clusters inicialmente son iguales, posteriormente en cada iteración va a ir actualizando las medias de los clusters para ajustar los tamaños de cada clúster [2].

2.5. Modelos gráficos.

Los modelos gráficos son modelos estadísticos representados mediante un grafo que muestra relaciones condicionales entre sus variables, en los que destacan las redes bayesianas, los campos aleatorios de Markov, entre otros. [11] Un ejemplo de estos modelos se puede observar en la figura 2.3, donde se muestran dos variables de un conjunto de datos **A** y **B**. La variable **A** tiene una relación causal sobre la variable **B** representada por un arco con dirección de **A** hacia **B** y donde la probabilidad de **B** dependerá de la configuración de **A**.

Figura 2.3: Ejemplo de un modelo causal gráfico.

Estas grafos representan las relaciones entre las variables de una base de datos y existen numerosas combinaciones para representar una base de datos con n variables, que responde a la siguiente fórmula:

$$C_n^p = \binom{n}{p} = \frac{n!}{p!(n-p)!} \tag{2.1}$$

Para el sustento de la propuesta de este documento sólo se desarrollará el concepto de red bayesiana. Por lo que debemos tomar en cuenta que solo podemos tomar redes acíclicas lo que reduce el número de soluciones factibles para la búsqueda de la red [12].

2.6. Redes Bayesianas.

Una red bayesiana o de probabilidad es un modelo que trata de predecir un fenómeno a partir de una colección de datos, estos datos se componen de un conjunto de variables, las cuales poseen características de forma individual o pueden estar relacionadas con otras variables ya que pueden presentar alguna dependencia.

A partir de una base de datos se puede crear una red bayesiana que consta de dos componentes, el primero es un grafo acíclico dirigido y el segundo una tabla de probabilidad condicional (véase fórmula 2.2), la cual es una representación de probabilidad de las relaciones entre las variables representadas en el grafo y la base de datos, para obtener las probabilidades que tiene cada variable de tomar un valor especifico [3] .

Considérese un conjunto finito de variables aleatorias $\mathbf{U} = \{X_1...X_n\}$. donde cada variable X_i puede tomar valores de un set finito de variables. Las variables mayúsculas *X,Y,Z* son para determinar los nombres de variables y *x,y,z* para nombrar los valores que pueden adquirir las variables, una red bayesiana es un par $B = \langle G, \Theta \rangle$ que representa una distribución de probabilidad con respecto a $\mathbf{U}$ donde G es un grafo acíclico con cada nodo correspondiente a cada variable de $\mathbf{U}$, es decir $\{X_1...X_n\}$, y cada nodo es independiente a sus descendientes y dependientes a sus padres, el segundo componente es Θ que representa la tabla de probabilidades condicionales representadas cada nodo y cuyos valores se obtienen a partir de la estructura y la base de datos. Una red B para un conjunto de datos $\mathbf{U}$ se define de la siguiente manera:

$$P_B(X_1, ..., X_n) = \prod_{i=1}^{n} P_B(X_i|\Pi_i) \tag{2.2}$$

lo cual quiere decir que la probabilidad conjunta de las variables es igual a la multiplicatoria de la probabilidad condicional de las variables X_i con respecto a sus padres Π_i citethiesson.

Las redes bayesianas son utilizadas para el estudio de fenómenos los cuales pueden ser medidos o registrados, y a través de estos modelos se pueden realizar predicciones estadísticas [3]. Sin embargo, una red bayesiana consta de dos componentes: El primero es un grafo acíclico dirigido que puede ser libre de restricciones lo que nos presenta un gran número de posibles combinaciones de las cuales elegir, y por ello se han implementado diferentes tipos de métricas para saber cuál es la red que mejor representa a un conjunto de datos, algunas de estas métricas son el porcentaje de clasificación, MDL (Minimum Description Length), AIC (the Akaike's Information Criterion), BIC (the Bayesian Information Criterion) y con ellas se pueden identificar a las redes que mejor representan a una base de datos. MDL por ejemplo, está basado en el principio de compresión de información e implementa dos factores uno es el ajuste y el otro es la complejidad de la red, la cual es un valor de penalización para redes complejas, es decir, favorece a redes simples con un buen ajuste [14].

El problema de obtener una red bayesiana que represente adecuadamente un conjunto de datos se presenta en cuatro escenarios, véase cuadro 2.1. Los peores escenarios marcados por la literatura son aprender la estructura y los parámetros (tabla de probabilidad conjunta) a partir de un conjunto de datos completos y el segundo es aprender la estructura y los parámetros de un conjunto de datos incompletos.[4]

En muchas ocasiones, algunas de estas técnicas consisten en eliminar o aproximar los datos faltantes e ignorar las variables ocultas, lo que implica que pueda existir una pérdida de información significativa para la obtención de la red bayesiana. Es por ello que se han implementado algoritmos que a partir de una base de datos se puedan aproximar los valores de una variable.

Sin embargo, se presentan dos problemas de alto costo computacional: Por un lado la obtención de la red bayesiana y por el otro la obtención de los valores escondidos. Al ser problemas en los cuales su espacio de soluciones es bastante amplio, se deben implementar algoritmos heurísticos, dado que una búsqueda exhaustiva resulta demasiado costosa, es posible aproximar los valores faltantes mediante algoritmos de

aprendizaje no supervisado como EM, el cual a partir de distribuciones gaussianas busca características de las variables en los datos y con ellas puede tratar de predecir los valores de las variables ocultas.[5] Y una vez que el conjunto de datos esté completo se podrá utilizar un algoritmo de búsqueda que sea capaz de encontrar una red bayesiana que represente adecuadamente los datos.

EM es un algoritmo iterativo donde entran dos partes importantes para su funcionamiento: la primera es referente a la esperanza (expectation), que es la estimación de valores de la distribución de probabilidad del fenómeno, a esta prueba se le conoce como log-likelihood. Esta función nos indica si los parámetros de la distribución gaussiana, es decir, la media y la amplitud, son los apropiados para representar los valores. La otra parte del algoritmo es la que se encarga de maximizar los valores de los parámetros para ajustar las distribuciones gaussianas de acuerdo a los datos a través del proceso iterativo. El criterio de elección de los parámetros óptimos es el que minimice el error ε. [15]

2.7. Maximum Likelihood.

El método de maximum likelihood es muy atractivo, porque tratamos de encontrar los valores reales de los parámetros que pudieron haber producido los datos observados. Para la mayoría de los casos su rendimiento es bueno donde las muestras son suficientemente grandes. Este es uno de los métodos más versátiles para el ajuste de parámetros de modelos estadísticos a partir de los datos. Para explicar el concepto de la función primero se debe presentar el concepto de la función Likelihood.

Sea la probabilidad conjunta (o densidad) una función de n variables aleatorias $X_1, ..., X_n$ con valores $x_1, ..., x_n$ la función likelihood es:

$$L(\theta; x_1, ..., x_n) = F(x_1, ..., x_n; \theta) \tag{2.3}$$

Si $X_1, ..., X_n$ son variables aleatorias discretas con una función de probabilidad $P(x, \theta)$ entonces la función likelihood es:

$$L(\theta) = P(X_1 = x_1, ..., X_n = x_n) = \prod_{i=1}^{n} P(X_i = x_i) = \prod_{i=1}^{n} P(x_i, \theta) \tag{2.4}$$

Ahora bien con la fórmula anterior con los conceptos que han estado presentando con anterioridad donde θ correspondería a la tabla de probabilidad conjunta de la base de datos, que se obtiene de una red bayesiana M, por lo que se obtendría la siguiente fórmula:

$$LL(Bs, \theta) = \prod_{i=1}^{n} PB(X_i|M, \theta) \tag{2.5}$$

Ahora bien como el objetivo es maximizar el valor de likelihood de la variable H cuyos parámetros se desconocen se obtendrá lo siguiente:

$$MLL = argMaxPB(H|M, \theta) \tag{2.6}$$

El algoritmo EM trata de maximizar la esperanza de la siguiente probabilidad de la fórmula 2.6 la cual representa la probabilidad de que x tenga de padre θ con los parámetros de configuración de la distribución de probabilidad gaussiana θ^H. El algoritmo tiene un conjunto de variables θ las cuales inician de forma aleatoria, y en cada iteración ajustan algunos valores para hacer que el algoritmo caiga en un punto de convergencia [16].

2.8. Maximización de la esperanza estructural

Para el algoritmo de maximización de la esperanza estructural SEM por sus siglas en inglés (Structural Estimization Maximization) al igual que para el algoritmo MS-EM, su función de verosimilitud está regida por una red bayesiana. Sin embargo, a diferencia de MS-EM, SEM no busca una red bayesiana libre de restricciones que muestre la relación de la variable latente con las variables observables de la base de datos, sino que tiene una estructura determinada donde se asume independencia entre las variables y una relación condicional entre la variable latente y todas las variables de la base de datos. Al tener una estructura predefinida el costo computacional es menor dado que no es necesario encontrar una nueva red, sino que el proceso iterativo se centra en ajustar las medias de los clusters [10].

En la fórmula 2.7 se muestra su función de verosimilitud, la cual va a realizar todo el proceso de ajuste con los datos, donde la variable o representa a los datos observados, M el modelo y h la variable no observada o latente [10].

$$Q(M : M_n) = E\left[logPr(H, o, M[h])|M_n^h, o\right] = \sum_h Pr(h|o, M_n^h) logPr(h, o, M^h) \tag{2.7}$$

El algoritmo de SEM consiste en dos pasos, el de esperanza que va a ser donde se ajusten las probabilidades de los clusters a través de la función de verosimilitud y un proceso de maximización que sustituye las medias de los clusters.

Capítulo 3

Análisis del algoritmo MS-EM.

Como ya se ha discutido en las secciones anteriores es posible obtener información de a partir de los datos. Sin embargo, puede llegar a ser un proceso bastante complejo si se consideran los siguientes escenarios ¿La ubicación de mi variable latente dentro de una red bayesiana es la que me va a proporcionar un mejor resultado? o ¿Cuántos valores puede obtener dicha variable? Ahora bien, la respuesta a estas dos preguntas sigue siendo un gran problema en el área, sin embargo, se han planteado diferentes propuestas para obtener un resultado a partir de ese escenario, una de ellas es el algoritmo de MS-EM planteado por la literatura el cual se desarrollará en este capítulo.

Ahora bien, la propuesta que ofrece la literatura es la siguiente, si se pudiera encontrar una red bayesiana que explique los datos y a su vez poder mostrar la presencia de una variable oculta, así como su relación con los datos observados se resolverían las dos problemáticas. El algoritmo propuesto es el siguiente[5]:

```
Procedure MS-EM:

        Choose M(0) and params(0) randomly
        Loop for n=0,1,... until convergence
                Find a Model Mn+1 that maximize Q(:Mn, params n)
                let params n+1 = arg max Q(Mn+1, params : Mn, params n)
```

Donde M es una red bayesiana y params es la distribución de probabilidad de la variable oculta. Ahora bien el algoritmo muestra sólo dos pasos, así que a continuación se analizará la función de esperanza (arg max Q) es la máxima verosimilitud véase capítulo 2.4 , teniendo un modelo inicial y un experimento aleatorio de clusters se busca obtener los parámetros o la probabilidad de que cada tupla pertenezca a un clúster. La versión tradicional de EM indica que el número de clusters será calculado de forma aleatoria y será un número entre $2 < c < n$ y la probabilidad de cada clúster es asignada de forma aleatoria de acuerdo al experimento inicial. En cada iteración las probabilidades de los clusters se ajustarán hasta que no exista un cambio significativo, lo que se puede definir en lo que llamaremos una variable de error ε, la cual es un parámetro del algoritmo, a través de este proceso se logra obtener la clusterización.

El siguiente paso es la maximización “Find a Model”. Como se ha mencionado anteriormente en este documento, encontrar una red bayesiana es un problema alta complejidad computacional, ya que existe un número bastante considerable de posibles redes que pueden representar una base de datos, por lo que para obtener una solución a este problema dentro de la propuesta de este documento se implementó un algoritmo de búsqueda que fuese capaz de encontrar una “mejor red”que representara a un conjunto de datos. Para ello se escogió el algoritmo Random Mutation Hill-Climbing véase sección 3.1.1. Para ello se definió como métrica del algoritmo el valor de MDL para determinar que una red sea mejor que otra. A continuación se desarrollará en primer lugar la métrica ocupada y después el algoritmo de búsqueda.

3.1. Longitud mínima de descripción.

Esta métrica también conocida como MDL por sus siglas en inglés (Minimum Description Length) está enfocado a la codificación de compresión de datos. Por lo que es necesario conocer la teoría de codificación de compresión de datos y luego aplicarla para seleccionar un modelo.

Teniendo el supuesto de enviar un mensaje codificado y que este contenga el menor número de caracteres posibles con la menor perdida de información posible, es decir, que sea la representación más simple, en el caso de que el mensaje que deseamos codificar fuese una base de datos. Y que tenemos n variables y para cada variable k padres se necesitaría $k\log(n)$ bits para listar a los padres para representar las probabilidades condicionales del modelo, por ejemplo: Si una variable puede tomar cinco valores y tiene cuatro padres y cada padre puede tomar tres valores distintos se necesitaran $3^4x(5-1)$bits para representar dicha probabilidad. Ahora bien MDL es una métrica que consta de dos componentes los cuales son la complejidad y el ajuste. Esta primera es una métrica de penalización dado que se busca la mejor representación simple y el ajuste es referente a como el modelo codifica los datos. La fórmula que compone esta métrica es la siguiente:

$$MDL(B_S, D) = -N \cdot H(B_S, D) - \frac{1}{2}K \cdot \log N \tag{3.1}$$

Donde $K = \sum_{i=1}^{n} q_i \cdot (r_i - 1)$ y $H(B_S, D) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} \frac{N_{ijk}}{N} \log \frac{N_{ijk}}{N_{ij}}$
En esta fórmula se expresa lo siguiente: N es el numero total de casos, $H(B_S, D)$ es la medida de ajuste y por último $-\frac{1}{2}K \cdot \log N$ es el valor de penalización donde se da preferencia a redes con menos arcos y donde el valor q_i es referente al número de combinaciones que se pueden realizar entre un nodo padre e hijo, en los casos de N_{ijk} y N_{ij} referente a la fórmula son frecuencias conjuntas de los nodos padres con respecto a los hijos, que se analizan en la base de datos D, dada la estructura B_S. [14] A través de esta fórmula se obtendrá un valor negativo que entre más cercano sea a cero mejor es la red que representa los datos, dando una función compromiso que se puede observar en la figura 3.1 [1].

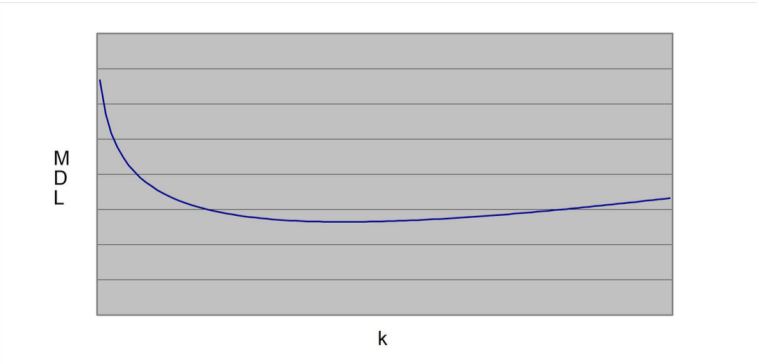


Figura 3.1: Gráfica de función compromiso entre la complejidad y el ajuste.

En la propuesta original de la literatura del algoritmo MS-EM se utiliza MDL como métrica para ver el comportamiento de la variable oculta que se esta obteniendo y su relación con la base de datos. De esta forma se realiza el proceso de sustitución iterativa del modelo siempre y cuando una nueva red obtenga un mejor valor de MDL. A este proceso dentro del algoritmo EM lo conocerá como esperanza.

3.2. Random mutation Hill-Climbing.

Se implementó el pseudocódigo del algoritmo de búsqueda aleatoria RMHC (Random Mutation Hill-Climber) el cual se encarga de buscar un mejor candidato a través de un enfoque Greedy, es decir, este algoritmo generará 10,000 grafos acíclicos dirigidos y compararlos mediante un valor de MDL (Minimum Description Length) y regresar aquella que represente a la mejor red que encontró.

Este algoritmo utiliza una probabilidad de mutación del 0.5 para cada nodo de la red existe una posibilidad de cambio del 50 % generando así nuevas estructuras, existen tres operaciones posibles para cada nodo de la red, *agregar, eliminar o invertir un arco* y dependiendo de la estructura analizará los movimientos válidos para modificar la red, mediante el algoritmo de ordenamiento tipológico y así generar posibles soluciones. Los valores de calibración del algoritmo *ne (número de evaluaciones)* y *PM (probabilidad de mutación)* se obtuvieron mediante pruebas exhaustivas y se asignaron los valores donde el algoritmo presentó un buen comportamiento. Al ser

un algoritmo estocástico las soluciones que arrojará dicho algoritmo serán diferentes dada su inicialización aleatoria. A continuación se detalla el funcionamiento del algoritmo RMHC en el Código 3.1.

Código 3.1: Hill-Climbing

```
Pseudocódigo RMHC

Input: DataSet, ConjuntoDeNodos[], MejorRed[].
ne=0 Número de evaluaciones.
Vecino = []
Mejor = MejorRed
MdlMejor = Mdl(Mejor, DataSet)
While (ne < 10000) do
        For_each ConjuntoDeNodos do
                ProbMutacion=rand(0.01 , 0.99)
                if probMutacion > 0.5 then
                        enlace = Seleccionar una modificación válida (
                            Agregar, Eliminar, Invertir)
                end_if
                Vecino.add(enlace)
        end_for_each
        MdlVecino = Mdl(Vecino, DataSet)
        if (MdlVecino < MdlMejor)then
                Mejor=Vecino
                MdlMejor = MdlVecino
                Vecino=[]
                MdlVecino=0
        end_if
        ne++
end_while

Return Mejor
```

A diferencia del algoritmo original del hill-climbing, RMHC permite una mayor exploración en el espacio de soluciones dado que su proceso de mutación que funciona de forma aleatoria generando así soluciones que son similares a la original. Del psuedocódigo anterior se muestra cómo el algoritmo es capaz de encontrar una nueva red, generando soluciones vecinas y escogiendo la mejor cuya función de aptitud es el valor de MDL (Minimum Description Length) véase la fórmula 3.1.

3.3. Algoritmo.

El algoritmo que propone la literatura es una versión simplificada y reducida de lo que se tiene que hacer, por lo que para su implementación se realizo un análisis del algoritmo MS-EM.

En primer lugar se procedió a hacer un algoritmo más extenso de MS-EM tomando en cuenta los conceptos que se presentaron anteriormente, véase código 3.2. En este algoritmo se inicializa una variable aleatoria para determinar el número de clusters que se van a calcular, después se ejecuta un experimento aleatorio con una distribución de probabilidad normal donde cada túpla tiene la misma probabilidad de pertenecer a un clúster. En esta primera iteración se obtienen las probabilidades de cada cluster, en este caso la probabilidad de algunos clusters puede ser igual a cero debido al experimento aleatorio inicial.

En segundo lugar es necesario ajustar el tamaño de los clusters, mediante el proceso de maximización a través del método del MLL (maxima verosimilitud). Una vez que se obtengan los nuevos datos se podrá buscar un nuevo modelo (o red bayesiana) que explique mejor los datos obtenidos. Este proceso continuará iterativamente hasta que se deje de apreciar una diferencia significativa. De esta forma se obtendrá como resultado un conjunto de datos que están explicados a través de su modelo y su tabla de probabilidad conjunta.

Hay que tener en cuenta que si se ejecuta varias veces este algoritmo existe una posibilidad de obtener diferentes variables en cada ejecución es posible obtener un número infinito de variables ocultas. El autor omitió en primera instancia incluir el valor de clasificación dado que el mismo proceso de clustering puede causar un proceso de sobre ajuste. Por ello se implementó como métrica el valor de MDL en lugar de la clasificación.

En el siguiente algoritmo se resume en cuatro pasos el funcionamiento general del algoritmo MS-EM teniendo como configuración de parámetros con una función de paro ε la diferencia entre el MDL actual y el obtenido cercano a 0. Así como un número de iteraciones máximo de 30.

Código 3.2: Algoritmo MS-EM

```
Algoritmo MS-EM

Paso 1: Generar de forma aleatoria un modelo y una distribución de
    probabilidad, donde el número de clusters es un valor aleatorio
    entre 2 < c < n y la suma de la media de cada cluster es igual a 1.
Paso 2: Obtener una nueva red que mejore el valor de MDL a través de un
     algoritmo de búsqueda código (3.1).
Paso 3: Obtener una nueva distribución de probabilidad a partir del
    Maximum Likelihood con la nueva estructura fórmula (2.6).
Paso 4: Se comparan las dos redes bayesianas en función al valor de MDL
                if (MDL_Nuevo - MDL_Actual != 0) AND (t < 30)then
                        La nueva red sustituye a la actual y regresa al
                            Paso 2.
                else
                        Regresa La mejor distribución de probabilidad,
                            mejor modelo, mejor MDL
```

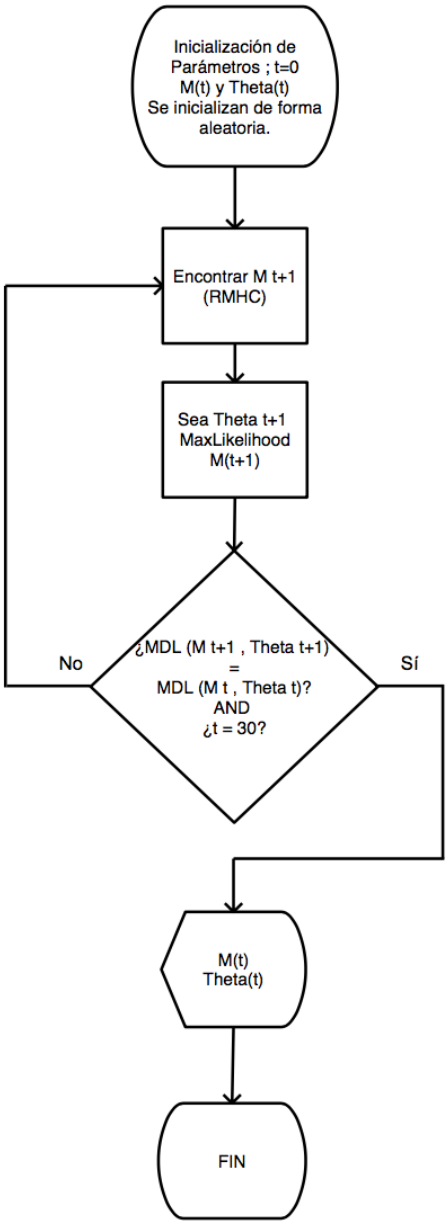
Del algoritmo MS-EM (código 3.2) se pude observar una descripción más detallada en su pseudocódigo véase código 3.3:

Código 3.3: Pseudocódigo MS-EM

```
Pseudocódigo MS-EM
INPUT: DataSet.
Begin:
c = Random(2,n) Donde n es el número de instancias del DataSet.
t=0 Donde t representa la iteración.
MDL_Nuevo=0 Variable temporal.
M(t)=Generar de forma aleatoria un modelo.
Theta(t)= Una distribución de probabilidad aleatoria donde la suma de
    la media de los clusters es igual a 1.
MDL_Actual= Calcular el valor de MDL de M(t) y Theta(t) fórmula (3.1).

while (MDL_Nuevo - MDL_Actual != 0) AND (t < 30)then
        MCandidato= Obtener una nueva red que mejore el valor de MDL a
            través de un algoritmo de búsqueda código(3.1).
        ThetaCandidato= Obtener una nueva distribución de probabilidad
            a partir del Maximum Likelihood de MCandidato obtenido con
            la fórmula (2.5).
        MDL_Nuevo= Calcular el valor de MDL de MCandidato y
            ThetaCandidato fórmula (3.1).
        if (MDL_Nuevo > MDL_Actual)then
                t++
                MDL_Actual=MDL_Nuevo
                M(t) = MCandidato
                Theta(t)=ThetaCandidato
                MDL_Nuevo=0
        end_if
end_while

Return Theta(t), M(t), MDL_Actual
```

Por último en esta sección de análisis y construcción del pseudocódigo del algoritmo MS-EM, se diseñó el diagrama de flujo de la figura 3.1 para mostrar su funcionamiento:

Figura 3.2: Diagrama de flujo de MS-EM



Con todo el análisis anterior se pretende dar una exposición con mayor profundidad del algoritmo propuesto por la literatura y con ello se pudo reproducir el algoritmo original.

3.4. Arquitectura.

A partir de la sección anterior se propuso una arquitectura para MS-EM (figura 3.2) y se seleccionaron los siguientes componentes:

- **Preprocesador:** El cual recibe un conjunto de datos como entrada, se encarga de asignar valores iniciales a los parámetros del algoritmo, así como dar un formato necesario a los datos para su tratamiento, también es el encargado de generar un red bayesiana aleatoria (M,θ).

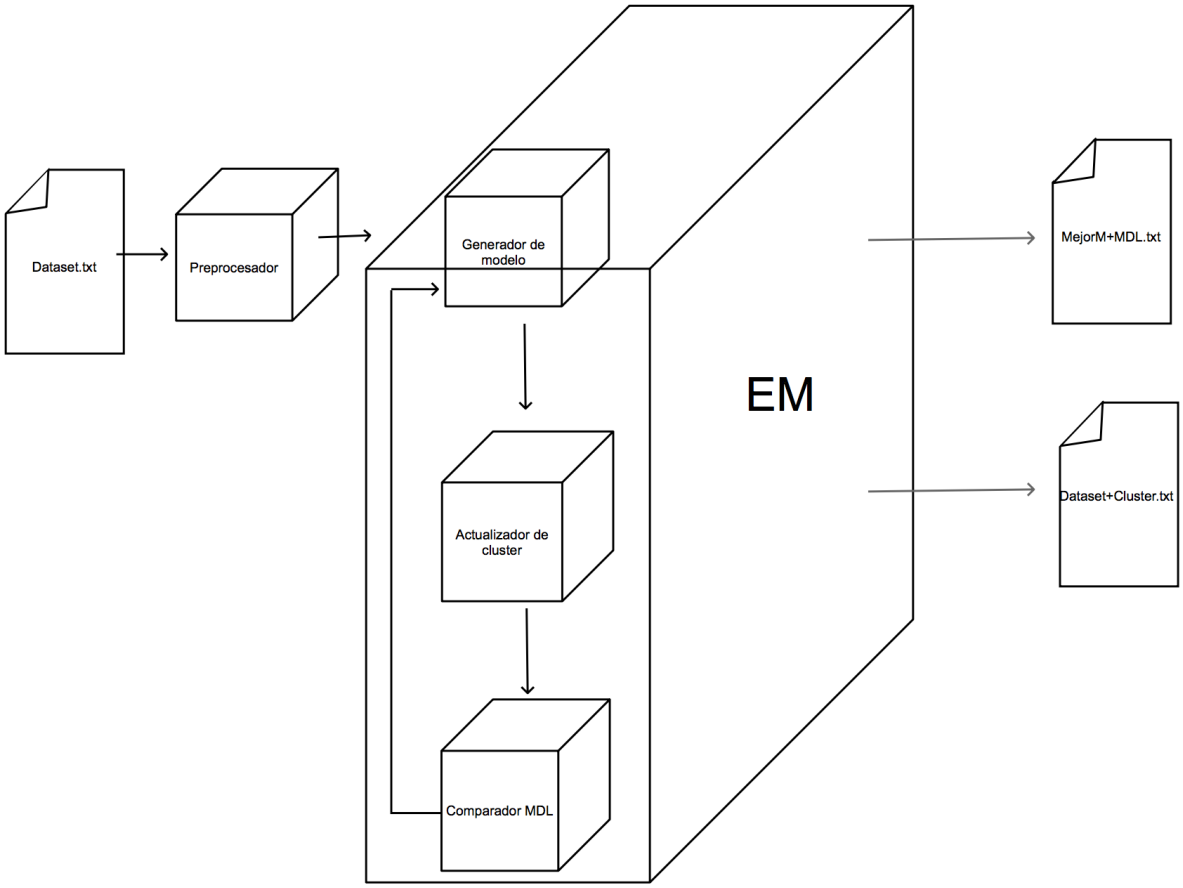
- **EM:** Es un componente conformado por tres subcomponentes, el cual recibe como parámetros una red bayesiana y su objetivo es regresar el mejor modelo en función a un valor de MDL y el nuevo conjunto de datos que se obtuvo.

- **Generador de modelo:** El cual recibe como parámetros una red bayesiana, se encarga de encontrar un grafo M(n+1) que maximice el valor de MDL con respecto al grafo anterior M(n).

- **Actualizador de clusters:** Su función es maximizar las probabilidades obtenidas en el nuevo grafo, modificando la distribución de probabilidad de cada cluster.

- **Comparador de MDL:** Va a determinar la función de paro, cuando el algoritmo no sea capaz de encontrar una mejor solución terminará o en caso contrario continuará su ejecución.

Figura 3.3: Arquitectura MS-EM



Con todo lo anterior se pretende dar una visión más extensa del algoritmo MS-EM y así poder desarrollar una alternativa que se plantea en este documento en el proceso de EM dentro del algoritmo, en este caso implementando la configuración de parámetros y funciones de REM y así lograr implementar la propuesta MS-REM.

Capítulo 4

Implementación de la propuesta MS-REM.

4.1. Marco general.

Las siglas de MS-REM significan selección de modelo mediante la maximización de la esperanza robusta. Esta propuesta recae totalmente en la parte de estimación de parámetros del algoritmo, es decir EM, la literatura ofrece múltiples variantes del algoritmo EM teniendo diferentes alternativas en su funcionamiento, una de ellas es en el cálculo de número de clusters, llamada EM robusto (REM), el cual empieza con el peor de los casos del clustering, cada túpla de nuestra base de datos pertenece a un cluster distinto y se va ajustando el número de clusters, hasta un número que determine el algoritmo de acuerdo a la función de verosimilitud (likelihood). El principal problema de esta propuesta es que el algoritmo original fue diseñado y probado con conjunto de datos en dos dimensiones y distribuciones generadas de forma sintética, por lo que se procedió a adaptar el concepto que plantea el autor con los conceptos que se plantean en la versión original de REM.

4.2. Algoritmo.

Durante la fase de diseño para la propuesta que une las ideas de un algoritmo que se enfoca en obtener una categorización a partir de su modelo y un algoritmo robusto de clusterización, se logró diseñar el pseudocódigo de MS-REM (Model Selection Robust Expectation Maximization) donde se añadieron las características que ofrece la literatura con respecto a este algoritmo, la mayoría de estas modificaciones están enfocadas en el cálculo de clusters ya que en su configuración inicial, el número de ellos ya no es un valor aleatorio entre 0 y n, sino que inicia con un valor inicial de n donde n es el número de casos observables,El algoritmo inicia tomando en cuenta que se encuentra en el peor de los casos y a partir de ello genera un experimento aleatorio, donde cada clúster tiene una probabilidad de aparecer de $1/n$ en esta primera iteración es posible que algunos las medias clusters tengan una probabilidad de aparecer igual a 0 y en otros casos van a incrementarse, generando así cúmulos de mayor tamaño. Es por esto que se optimizaron las funciones de ajuste del algoritmo para eliminar los clusters más pequeños (la literatura sugiere eliminar clusters cuya media sea menor a $1/n$) fomentando que los clusters más grandes tengan una mayor probabilidad de aparecer en las siguientes iteraciones. [2]

Aunque el proceso iterativo en primera instancia parezca más lento, el algoritmo parte de lo más general a lo específico fomentando así que los datos sean los que determinen las agrupaciones convenientes para cada conjunto de datos. Sin embargo, ésto también tiene sus debilidades, ya que en investigaciones posteriores se logró comprobar en datos generados por distribuciones artificiales, que el algoritmo permite clusters pequeños, generando así ruido en el proceso de clasificación. [17]

Código 4.1: Algoritmo MS-REM

```
Algoritmo MS-REM

Paso 1: Generar de forma aleatoria un modelo y una distribución de
    probabilidad, donde el número de clusters es n y la suma de la
    media de cada cluster es igual a 1.
Paso 2: Se ajusta el número de Clusters, eliminando a aquellos clusters
     cuya media sea menor a 1/n y se re-nombran los clusters.
Paso 3: Obtener una nueva red que mejore el valor de MDL (Fórmula 3.1)
    a través de un algoritmo de búsqueda código(3.1).
Paso 4: Obtener una nueva distribución de probabilidad a partir del
    Maximum Likelihood (Fórmula 2.6) con la nueva estructura.
Paso 5: Se comparan las dos redes bayesianas en función al valor de MDL
                if (MDL_Nuevo - MDL_Actual == 0)then
                        Regresa La mejor distribución de probabilidad,
                            mejor modelo, mejor MDL
                else
                        if t < 60 then
                                Se sustituye la red y regresa al (Paso
                                    2)
                        else
                                Se sustituye la red y regresa al (Paso
                                    3)
                        end_if
                        t++
                end_if
```

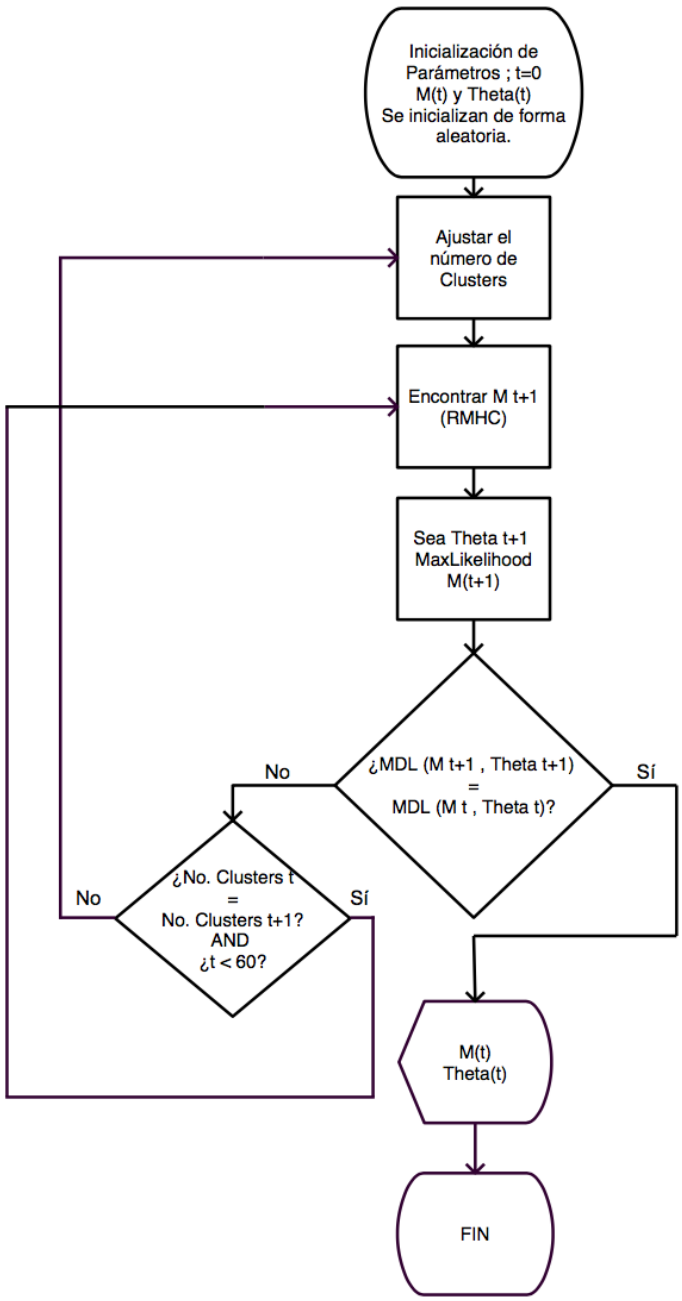
Del algoritmo MS-REM (código 4.1) se obtiene una descripción más detallada en su pseudocódigo, véase código 4.2, donde se añade el paso de ajuste y también una comparación con respecto al MS-EM (código 3.2) para determinar el número de clusters.

Código 4.2: Pseudocódigo MS-REM

```
Pseudocódigo MS-REM
INPUT: DataSet.
Begin:
c = n Donde n es el número de instancias del DataSet.
t=0 Donde t representa la iteración.
MDL_Nuevo=0 Variable temporal.
M(t)=Generar de forma aleatoria un modelo.
Theta(t)= Una distribución de probabilidad aleatoria donde la suma de la media de
    los clusters es igual a 1.
Se ajusta el número de Clusters, eliminando a aquellos clusters cuya media sea
    menor a 1/n y se re-nombran los clusters.
MDL_Actual = Calcular el valor de MDL de M(t) y Theta(t) fórmula (3.1).

while (MDL_Nuevo - MDL_Actual != 0) then
        nClustersActual=Obtener el número de Clusters en Theta(t)
        MCandidato = Obtener una nueva red que mejore el valor de MDL a través de
            un algoritmo de búsqueda código(3.1).
        ThetaCandidato = Obtener una nueva distribución de probabilidad a partir
            del Maximum Likelihood de MCandidato fórmula (2.6).
        MDL_Nuevo = Calcular el valor de MDL de MCandidato y ThetaCandidato fórmula
             (3.1).
        nClustersNuevos=Obtener el número de Clusters en ThetaCandidato

        if (MDL_Nuevo > MDL_Actual)then
                t++
                M(t) = MCandidato

                if (nClustersActual!=nClustersNuevos) AND (t < 60)then
                        Theta(t)=Se ajusta el número de Clusters de ThetaCandidato,
                             eliminando a aquellos clusters cuya media sea menor a
                            1/n y se re-nombran los clusters.
                        MDL_Actual=MDL_Nuevo
                        MDL_Nuevo=0
                else
                        Theta(t)=ThetaCandidato
                        MDL_Actual=MDL_Nuevo
                        MDL_Nuevo=0
                end_if
        end_if
end_while

Return Theta(n), M(n), MDL_Actual
```

Por último, en esta sección de análisis y construcción del pseudocódigo del algoritmo MS-REM, se diseñó el diagrama de flujo para mostrar su funcionamiento y donde se pueden apreciar los cambios respecto a la versión original(MS-EM) véase figura 4.1:

Figura 4.1: Diagrama de flujo de MS-REM



4.3. Arquitectura.

A partir de la propuesta original MS-EM se planteó proponer una arquitectura para MS-REM y se seleccionaron los siguientes componentes:

- **Ajustador de clusters:** Se encarga de eliminar todos los clusters cuya probabilidad sea menor a $1/n$ y reparando la base de datos para que se puedan seguir ajustando el número de clusters, en caso de que se elimine un cluster a la túpla afectada se le asignará aleatoriamente un cluster que esté presente en la distribución de probabilidad.

- **Preprocesador:** El cual recibe un conjunto de datos como entrada, se encarga de asignar valores iniciales a los parámetros del algoritmo, así como dar un formato necesario a los datos para su tratamiento, también es el encargado de generar un red bayesiana aleatoria (M,θ).

- **REM:** Es un componente conformado por de cuatro subcomponentes, el cual recibe como parámetros una red bayesiana y su objetivo es regresar: el mejor modelo con buen un valor de MDL, el nuevo conjunto de datos que obtuvo y el número de clusters que representan adecuidamente a los datos.

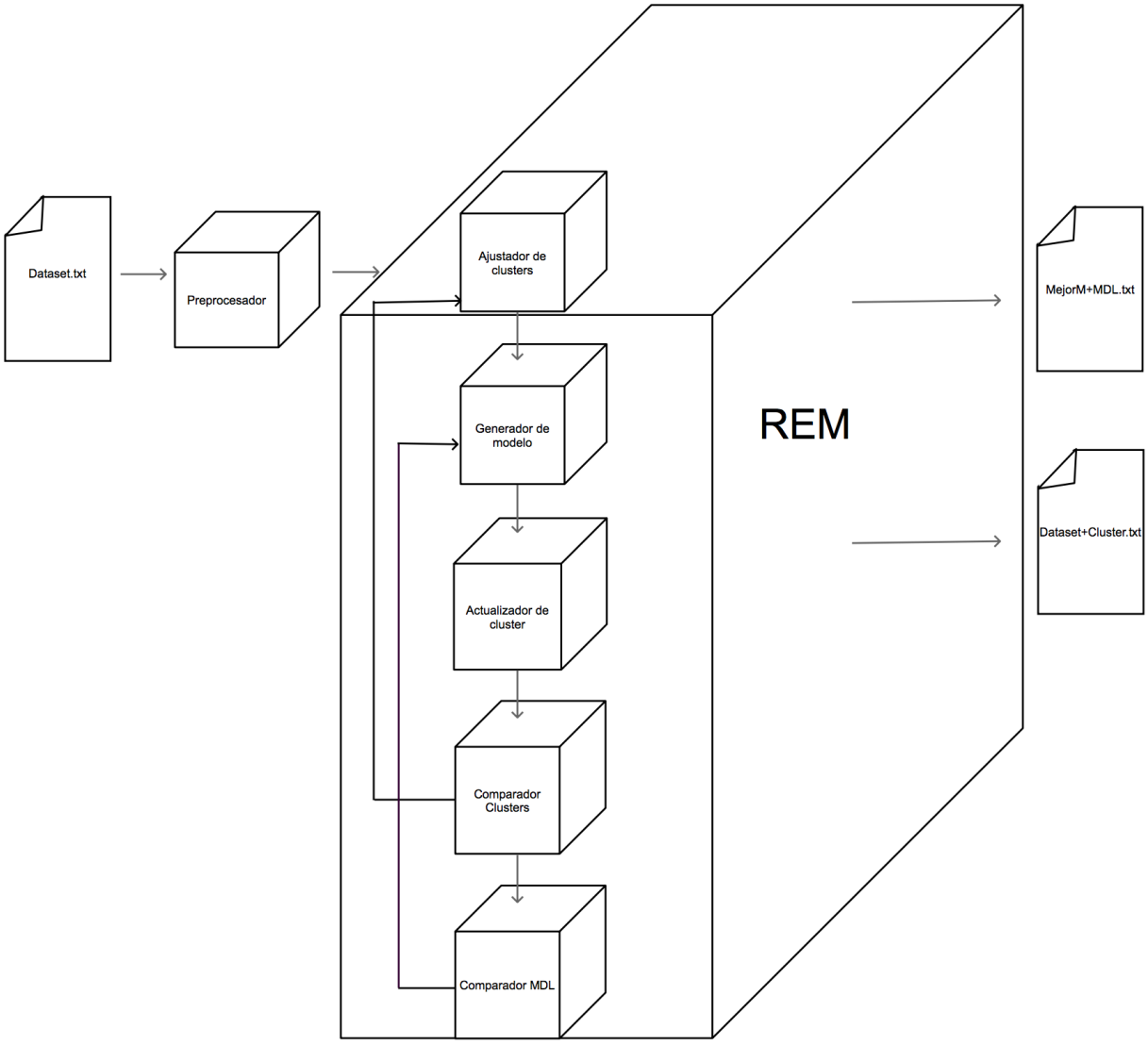
- **Generador de modelo:** El cual recibe como parámetros una red bayesiana, se encarga de encontrar un grafo M(n+1) que maximice el valor de MDL con respecto al grafo anterior M(n).

- **Actualizador de clusters:** Su función es maximizar las probabilidades obtenidas con el nuevo grafo, modificando la distribución de probabilidad de cada cluster.

- **Comparador de MDL:** Va a determinar la función de paro, cuando el algoritmo no sea capaz de encontrar una mejor solución terminará o en caso contrario continuará su ejecución.

- **Comparador de clusters:** Va a determinar si es necesario seguir reparando los datos o si se enfoca a maximizar la probabilidad de los que ya están representados en los datos.

Figura 4.2: Arquitectura MS-REM



Con todo el análisis anterior se procedió a implementar el algoritmo MS-REM en el lenguaje de programación Ruby, y se empezó a analizar su comportamiento con diferentes experimentos que se describirán en el capítulo 5.

Capítulo 5

Resultados y discusión

5.1. Comparativa a nivel arquitectónico entre MS-EM y MS-REM.

Una vez realizadas las dos arquitecturas se puede apreciar que la diferencia sustanciosa es el agregado de los componentes ajustador de clusters y comparador de clusters, los cuales añaden al proceso de búsqueda iterativo de EM, el encontrar un número de clusters que mejor va a representar la red, estos algoritmos proponen encontrar una red bayesiana con una variable H la cual puede ayudar a explicar la relación entre las variables que representan al fenómeno de estudio, consiguiendo así un mejor ajuste en los datos.

5.2. Pruebas con Bases de datos sintéticos.

En una primera etapa de experimentación se definieron dos redes bayesianas (3x1x3 y 3x2x4) con cinco distribuciones de probabilidad aleatorias y diferentes tamaños de muestra (250,500,1000,2000). Para un modelo se obtuvieron 20 Data set diferentes y se pretende observar el comportamiento de hasta tres variables ocultas. Se realizaron un total de 120 evaluaciones para todos los conjuntos de datos. Posteriormente se agruparon los valores de Log loss por tamaño de muestra y número de variables ocultas y con ello se obtuvo un promedio. Después se gráfico el promedio

Log loss de cada estructura por tamaño de muestra y número de variables.

Figura 5.1: Estructura 3x1x3

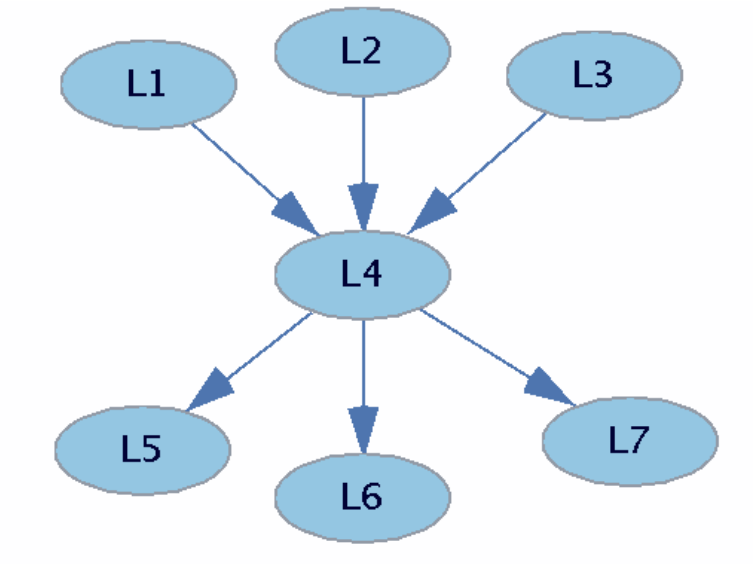


Figura 5.2: Estructura 3x2x4 con MS-REM

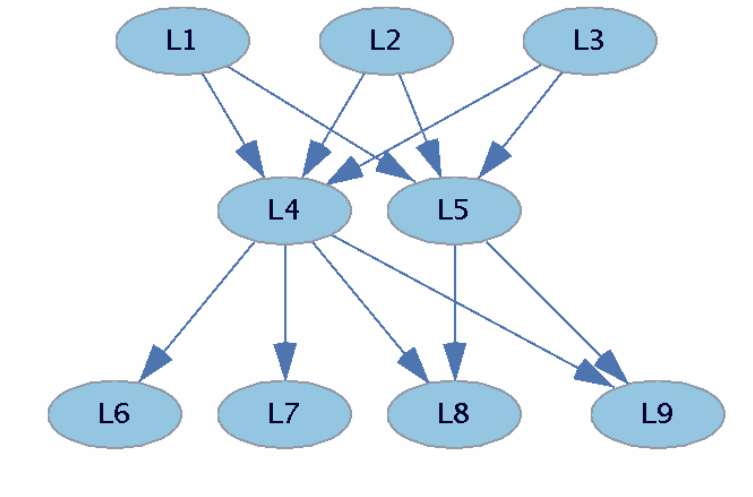


Figura 5.3: Log Loss con MS-REM en estructura 3x1x3 con hasta tres variables latentes

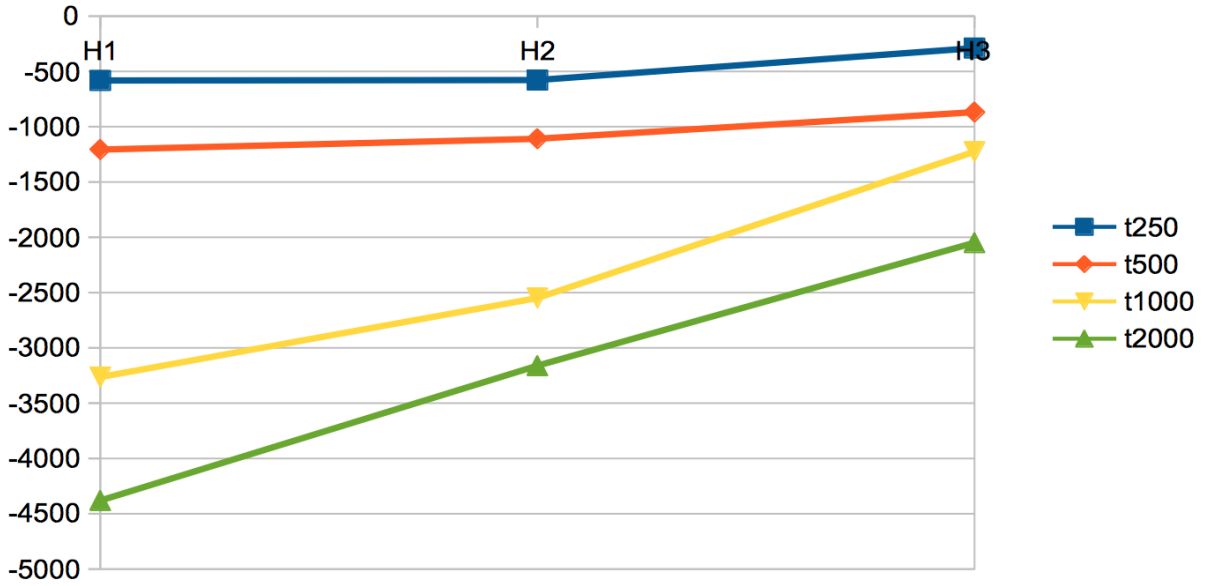


Figura 5.4: Log Loss con MS-REM en estructura 3x2x4 con hasta tres variables latentes

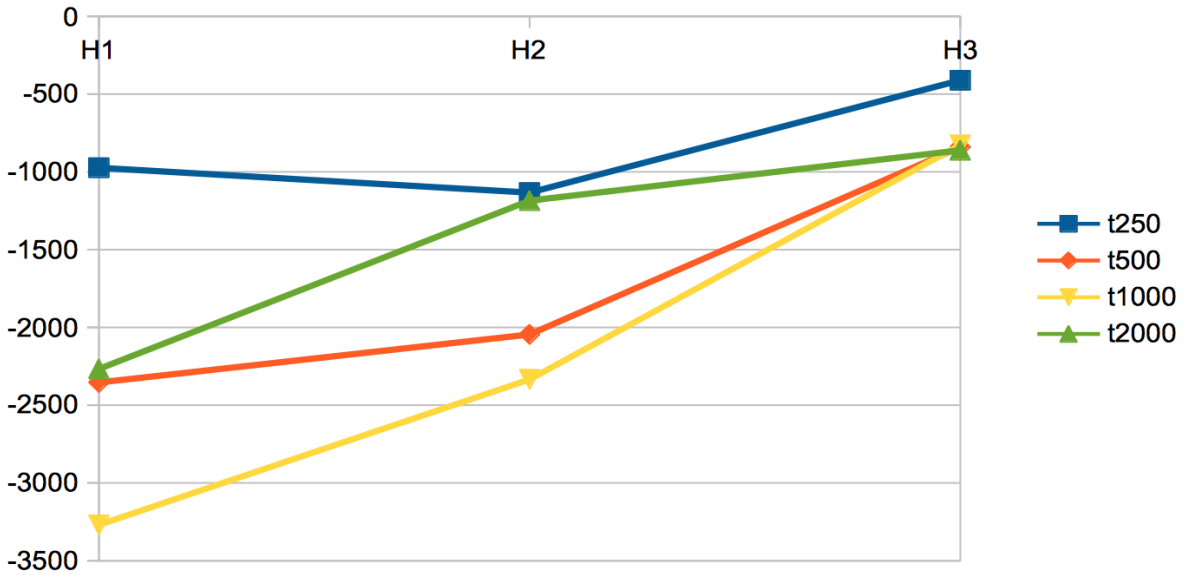


Figura 5.5: Log Loss con MS-EM en estructura 3x1x3 con hasta tres variables latentes

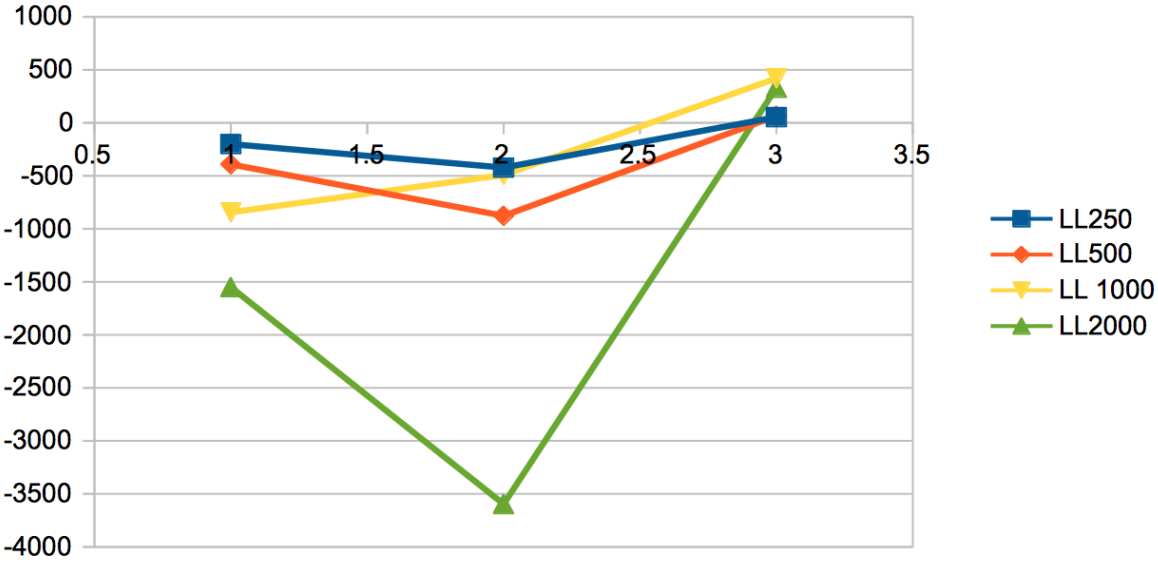
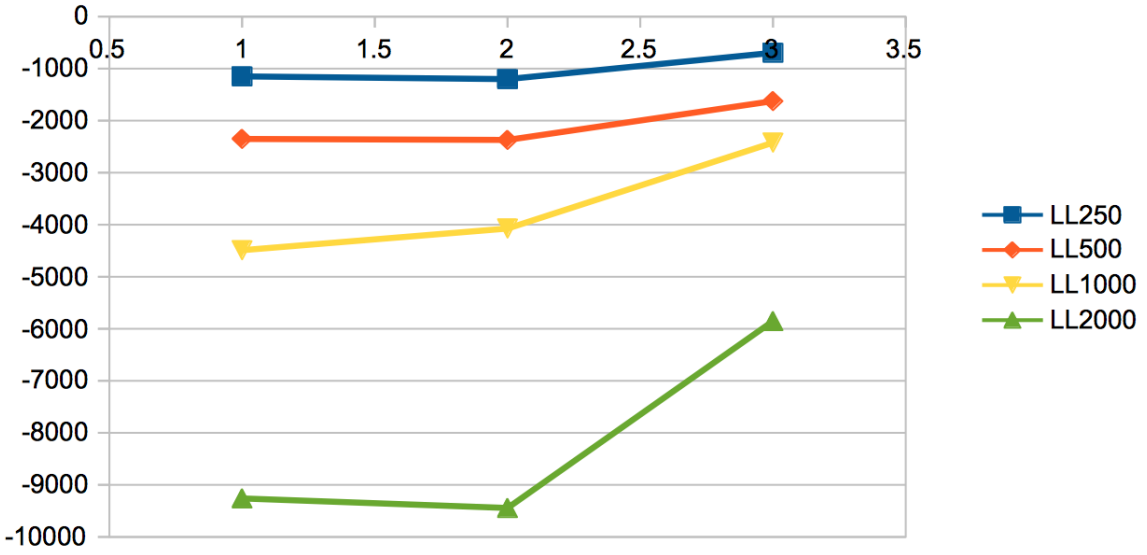


Figura 5.6: Log Loss con MS-EM en estructura 3x2x4 con hasta tres variables latentes



En estas gráficas (figura 5.1,5.2 y figura 5.3, 5.4) se aprecia cómo entre mayor sea el número de variables ocultas más le cuesta al algoritmo encontrar una red con un mejor MDL, ya que empieza a encontrar redes más complejas o redes que no representan adecuadamente los datos, lo que provoca que sus valores de MDL sean más altos. Esto se puede observar en las gráficas en H3, los valores de Log Loss son mas cercanos a cero.

En esta prueba también se pudo observar lo que el autor denomina como reducción de complejidad de la red, es decir, el algoritmo encontraba redes con menos arcos entrantes y un mejor valor de MDL. Las redes que se muestran en las figuras 5.7,5.8,5.9 se obtuvieron de la ejecución media del experimento para la red 3x1x3 con un cálculo de hasta 3 variables latentes, las figuras de 5.10, 5.11, 5.12 se obtuvieron de la estructura 3x2x4 en las mismas condiciones.

Figura 5.7: Estructura 3x1x3 y una variable latente

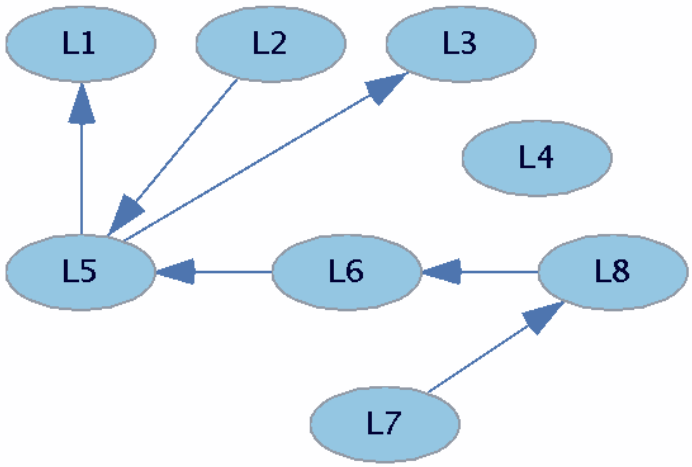


Figura 5.8: Estructura 3x1x3 y dos variables latentes

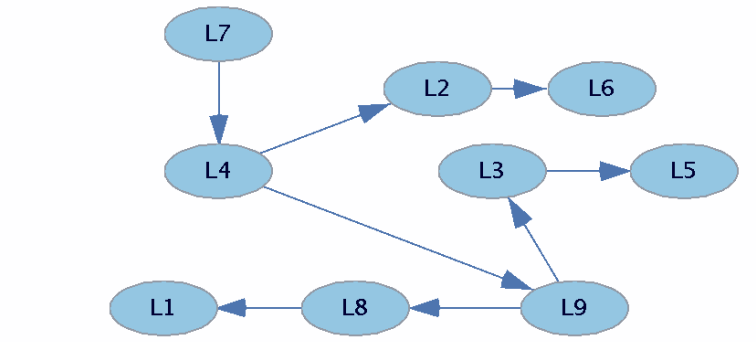


Figura 5.9: Estructura 3x1x3 y tres variables latentes

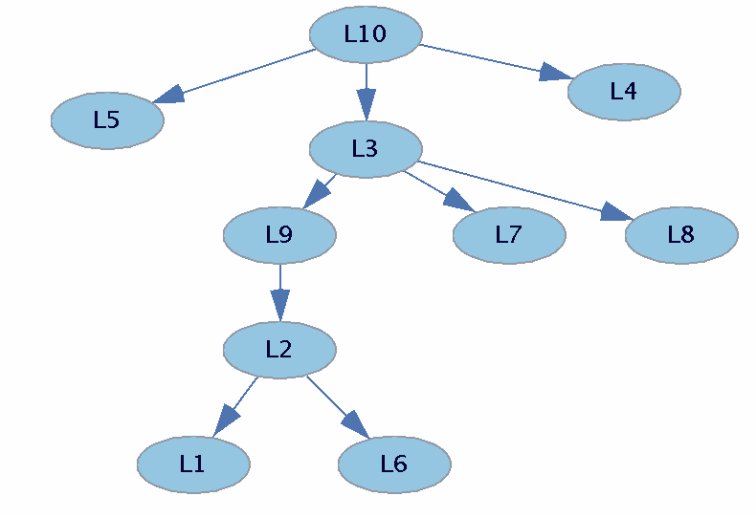


Figura 5.10: Estructura 3x2x4 y una variable latente

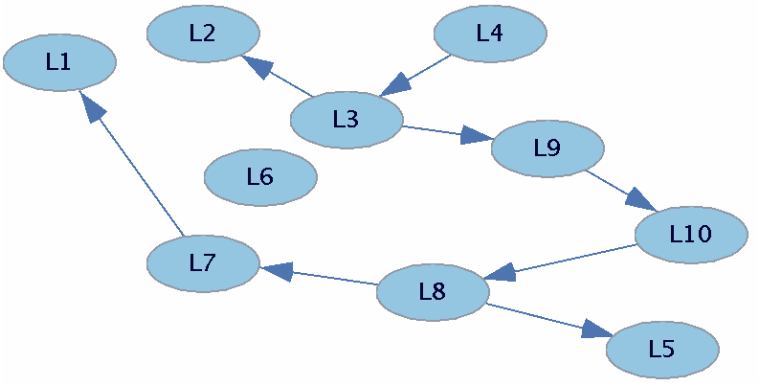


Figura 5.11: Estructura 3x2x4 y dos variables latentes

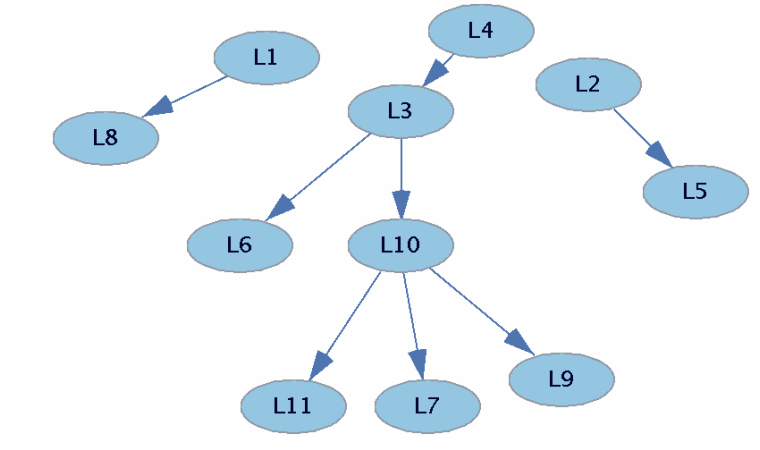
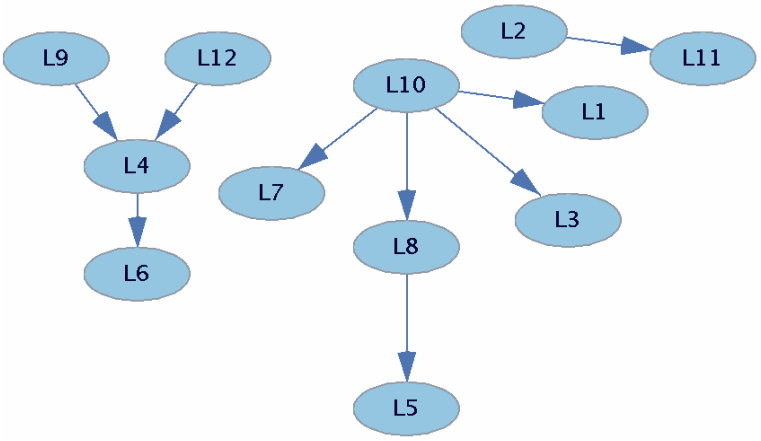


Figura 5.12: Estructura 3x2x4 y tres variables latentes



En el primer caso teniendo un grado de complejidad tres, al calcular la primera red se reduce la complejidad a un nivel dos, en el segundo caso se reduce a una complejidad de nivel uno y en el tercero permanece una red sin saturaciones manteniendo una complejidad simple de nivel uno.

Para el segundo caso donde existe una complejidad de tres se logra reducir, generando así un mejor valor de MDL, al calcular la primera variable latente se obtiene una complejidad uno, ocurre de forma similar en el segundo caso y al calcular la tercera variable oculta aumenta la complejidad de la red a nivel dos.

5.3. Estadísticas de rendimiento.

Para probar el funcionamiento de MS-REM se dío a diferentes bases de datos que se decriben en la cuadro 5.1, todos ellos ellos pertenecientes a la base de datos de la UCI, donde la variable clase es conocida, se eliminó esta variable y se realizaron 30 ejecuciones independientes para analizar sus resultados, MS-REM es un algoritmo aleatorio y es necesario realizar pruebas estadísticas para estudiar su comportamiento. [18]

	Iris	B. Cancer	Car	User	Wine
No. Clases	3	2	4	4	3
No. Instancias	150	286	1728	431	178
No. Variables	5	10	6	7	13
Dist. de clases	C1(0.33); C2(0.33); C3(0.33)	C1(0.70); C2(0.30)	C1(0.7); C2(0.24); C3(0.03); C4(0.03)	C1(0.12); C2(0.3); C3(0.28); C4(0.3)	C1(0.33); C2(0.39); C3(0.28)

Cuadro 5.1: Dónde C representa una clase dada.

	MS-REM		MS-EM	
Iris	MDL	No. Clusters	MDL	No. Clusters
Promedio	**-693.778899**	**10.4333334**	-735.5120	9.96666667
Media	**-677.5500297**	**12**	-791.5899	13
Mejor	**-635.19674**	**3**	-639.9313	3
Peor	**-784.546524**	**18**	-811.7601	17
DE	**43.90592259**	**4.1827505**	44.1580	3.9782

Cuadro 5.2: Tabla estadística del algoritmo MS-REM y MS-EM con la base de datos Iris, en relación MDL y número de clusters.

	MS-REM		MS-EM	
Wine	MDL	No. Clusters	MDL	No. Clusters
Promedio	**-3411.1798**	**15.5**	-3556.06285	17.2
Media	**-3439.5147**	**20**	-3444.62	15
Mejor	**-3264.215**	**7**	-3427.12	6
Peor	**-3554.1981**	**24**	-3746.34	23
DE	**62.6140**	**3.900**	139.9290	4.209429

Cuadro 5.3: Tabla estadística del algoritmo MS-REM y MS-EM con la base de datos Wine, en relación MDL y número de clusters.

	MS-REM		MS-EM	
B. Cancer	MDL	No. Clusters	MDL	No. Clusters
Promedio	-4573.24286	17.53	**-4472.01469**	**14.26**
Media	-4578.21657	15	**-4425.0089**	**14**
Mejor	-4365.966	16	**-4256.40323**	**8**
Peor	-4958.715817	23	**-4601.06887**	**14**
DE	164.41351	3.3808	**126.77428**	**2.947042**

Cuadro 5.4: Tabla estadística del algoritmo MS-REM y MS-EM con la base de datos B. Cancer, en relación MDL y número de clusters.

	MS-REM		MS-EM	
User	MDL	No. Clusters	MDL	No. Clusters
Promedio	**-948.357272**	**11.6666667**	-968.727833	11.4
Media	**-946.897761**	**11**	-952.361473	12
Mejor	**-906.993499**	**7**	-913.33975	8
Peor	**-1000.5133**	**8**	-1239.14843	14
DE	**19.934444**	**2.397316**	72.895105	2.954832

Cuadro 5.5: Tabla estadística del algoritmo MS-REM y MS-EM con la base de datos User, en relación MDL y número de clusters.

	MS-REM		MS-EM	
Car	MDL	No. Clusters	MDL	No. Clusters
Promedio	-21511.7231	26.3	**-21446.6814**	**21.5**
Media	-20647.463	18	-21796.1751	30
Mejor	-20647.463	18	**-19717.9403**	**2**
Peor	-22751.7638	34	**-22543.114**	**38**
DE	598.6598	4.620004	848.772215	12.204279

Cuadro 5.6: Tabla estadística del algoritmo MS-REM y MS-EM con la base de datos Car, en relación MDL y número de clusters.

De la experimentación anterior podemos observar que el algoritmo MS-REM presenta un mejor comportamiento con respecto a la versión original en bases de datos donde el número de valores de clase es mayor y la distribución de cada clase es menor. Como sucede con Iris, Wine y User. Mientras que en las bases de datos como Car o B. Cancer donde se encuentra una clase predominante, tiene un desempeño peor que la propuesta original. También el tamaño de la base de datos influye ya que en bases de datos de gran tamaño como Car, el algoritmo tiene un proceso de ajuste mucho mayor que la versión original.

5.4. Gráficas de convergencia

A continuación se mostrará el comportamiento de ajuste del algoritmo MS-REM en base al valor de MDL desde la primera iteración hasta alcanzar treinta iteraciones. Para realzar las siguientes gráficas se baso en la iteración media de las estadísticas de rendimiento.

Figura 5.13: Base de datos Iris

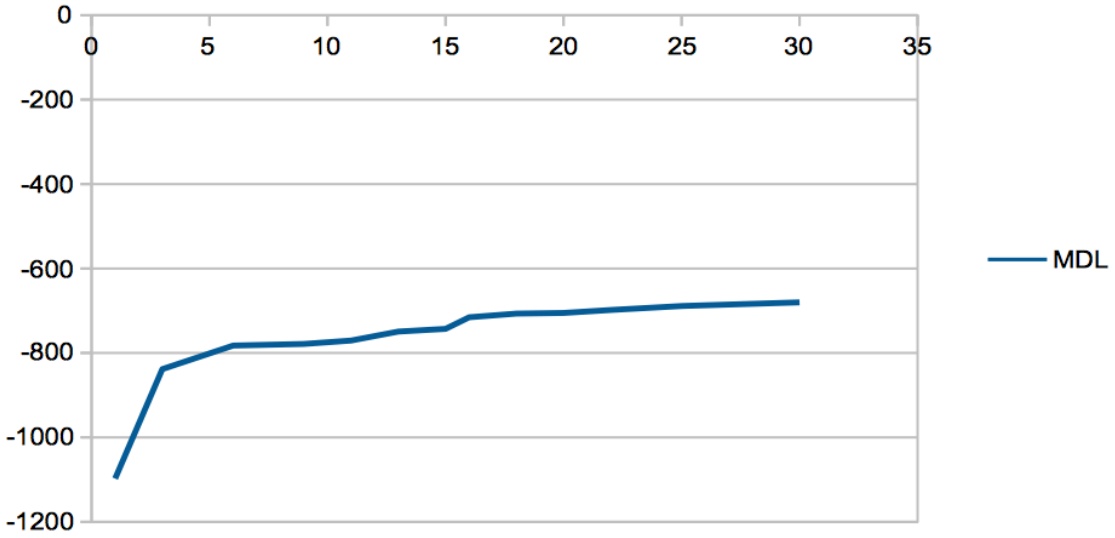


Figura 5.14: Base de datos Breast cancer

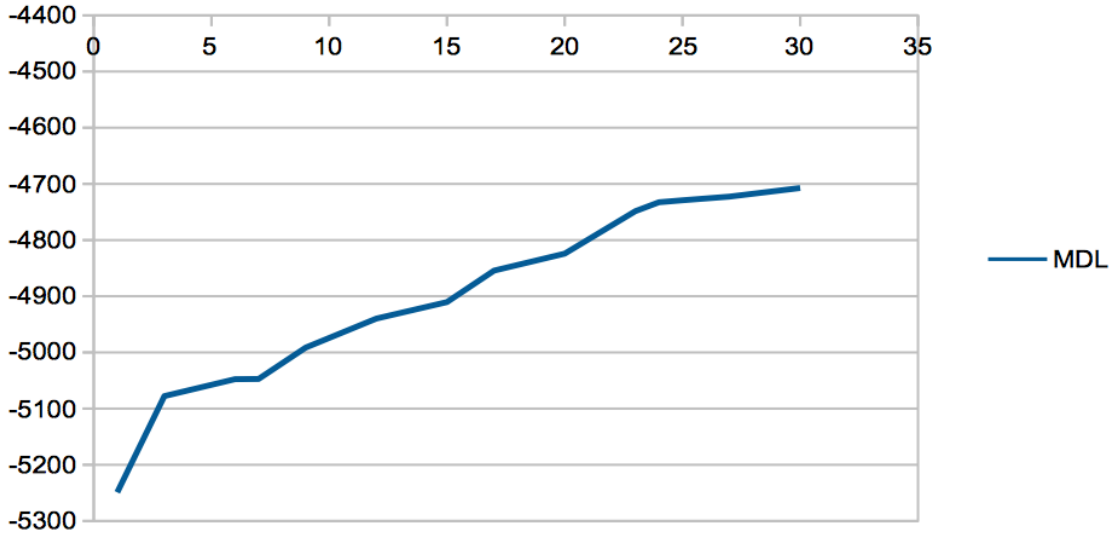


Figura 5.15: Base de datos Wine

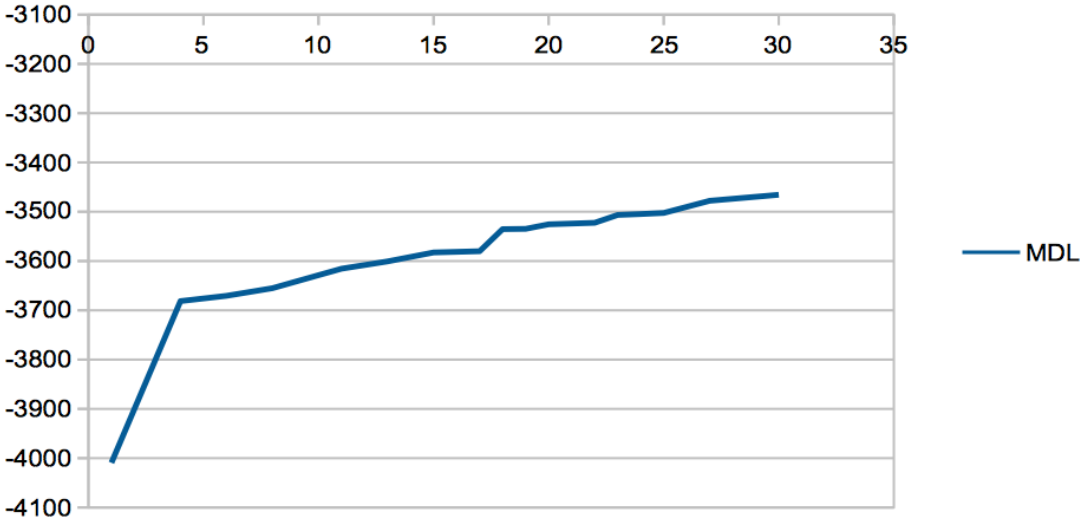
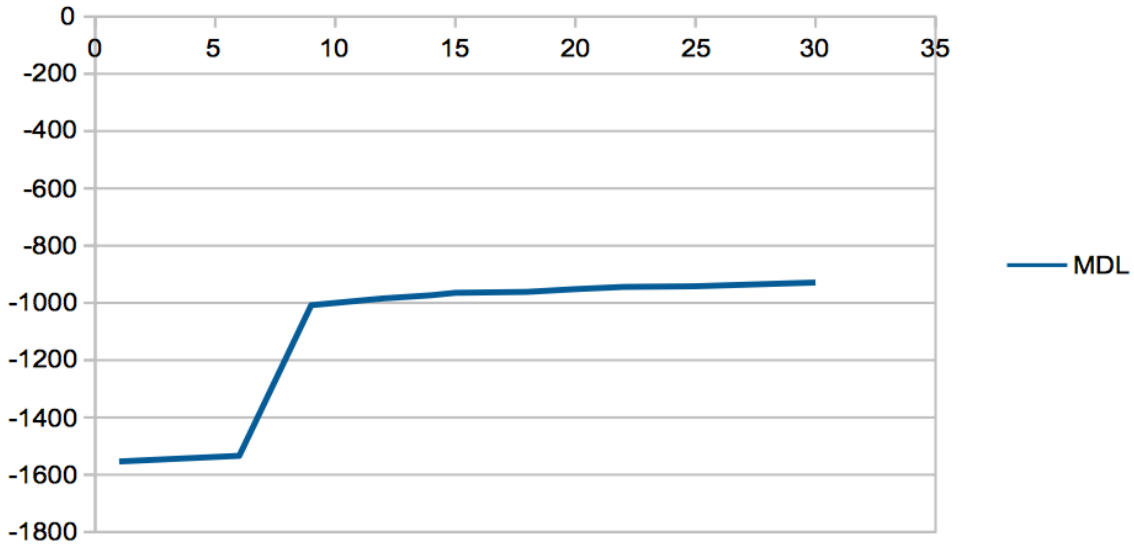


Figura 5.16: Base de datos User knowledge modeling



A través de estas gráficas se puede observar como el algoritmo inicia con un valor de MDL muy bajo en todas las bases de datos dado que el algoritmo tiene como configuración inicial el peor de los casos y como a través de las funciones de ajuste se va aproximando a cero. Obteniendo así un mejor proceso de agrupación con los datos observables y relación con ellas a través de la red bayesiana que se va modificando al mismo tiempo que las medias de los clusters.

Capítulo 6

Conclusiones y trabajo futuro

Durante el desarrollo de este documento se presento la construcción del algoritmo MS-REM, se realizaron diferentes pruebas para ver el comparar el funcionamiento del algoritmo original con el que se propone en este documento y ver de esta forma si las modificaciones de este son mejores o no. En primer lugar se realizo una prueba propuesta por la literatura donde se ejecuto el algoritmo con bases de datos sintéticas y se conocía la distribución de los datos. Se analizo la perdida de información que registraban los dos algoritmos. En segundo lugar se realizaron pruebas con bases de datos del repositorio de la UCI y se analizó el proceso de clustering que realizaron los dos algoritmos.

El algoritmo MS-REM al tener un comportamiento pensado en iniciar en el peor de los casos, muestra que su proceso de ajuste permite alojar pequeñas agrupaciones causando de esta forma que en bases de datos categóricas exista la división de clases. Sin embargo, lo que propone el autor es que puede existir un número infinito de variables ocultas dentro de una base de datos y que el algoritmo sólo es capaz de encontrar la "más importante" en un momento determinado y es por eso que la red bayesiana va cambiando de acuerdo al comportamiento del algoritmo [5].

Algo que nos indica la literatura sobre las variables ocultas y una de sus virtudes, es el hecho de que nos ayude a explicar un fenómeno[6], sin embargo como menciona el autor en la propuesta original del algoritmo el factor estocástico dentro

de la configuración inicial de EM, hace que solo encontremos la mejor variable para una configuración inicial determinada provocando que el estudio de estas variables sea complejo, ya que es difícil determinar que la variable obtenida en una ejecución determinada sea la más relevante de la base de datos [5].

Ahora bien a pesar de este pequeño inconveniente es una propuesta interesante dado que las variables que encuentra dicho algoritmo son obtenidas enteramente de los datos y más allá de configuración de parámetros del algoritmo, el usuario no tiene interacción alguna para modificar el comportamiento del algoritmo, proporcionando así soluciones interesantes que podrían explicar un fenómeno, ofrecer una alternativa de como se están categorizando los datos o incluso obtener los valores de una variable de gran impacto sobre la base de datos.

De acuerdo con la hipótesis planteada al inicio de este documento se logró desarrollar la implementación de un algoritmo capaz de realizar un proceso de detección de cúmulos que tomase en cuenta el peor de los casos y fuese ajustando sus cúmulos hasta encontrar la mejor distribución de los valores de una variable latente, tomando en cuenta el panorama más flexible para ello, lo que durante este documento se denomino robusto. Sin embargo en el proceso de ajuste que propone la literatura permite agrupaciones pequeñas debido a que el proceso de ajuste de clusters es mayor que en la versión original de la literatura. Es por eso que la versión robusta del algoritmo MS-EM funciona adecuadamente en bases de datos donde existen muchos cúmulos pequeños es por ello que cuando se compararon los algoritmos en bases de datos categóricas, el algoritmo tiene la mayoría de las veces un comportamiento inferior a la propuesta original.

6.1. Pruebas con grandes volúmenes de información.

Una de las principales intenciones de este trabajo es que el algoritmo sea capaz de darle un sentido e interpretación de los datos a través de la obtención de una variable oculta y que pueda mostrarse su relación con las variables observables a través de una red bayesiana. Uno de los posibles escenarios donde el algoritmo sería capaz de ser puesto a prueba en este panorama es el Análisis de impactos sobre la integridad ecológica tras alguna alteración humana al ecosistema, una investigación a cargo de CONABIO donde a partir de una base de datos, se necesita obtener ciertas variables para calcular la problemática presentada. Para calcular la variable integridad ecológica, es necesario en primer lugar encontrar tres variables latentes, por lo que es considerado para trabajo futuro hacer una experimentación donde se puedan analizar dichas variables y sus relaciones con las variables observadas. Y de esta forma poder calcular la variable integridad ecológica. En esta etapa no fue posible llevar a cabo este experimento dado al tamaño de la base de datos y el alto costo computacional del algoritmo, sin embargo, para el trabajo futuro se planea escalar la base de datos reduciendo su tamaño y así poder tener un análisis con los algoritmos MS-EM y MS-REM.

Bibliografía

[1] Nicandro Cruz-Ramírez, Héctor Gabriel Acosta-Mesa, Efrén Mezura-Montes, Alejandro Guerra-Hernández, Guillermo de Jesús Hoyos-Rivera, Rocío Erandi Barrientos-Martínez, Karina Gutiérrez-Fragoso, Luis Alonso Nava-Fernández, Patricia González-Gaspar, Elva María Novoa-del Toro, et al. How good is crude mdl for solving the bias-variance dilemma? an empirical investigation based on bayesian networks. *PloS one*, 9(3):e92866, 2014.

[2] Miin-Shen Yang, Chien-Yo Lai, and Chih-Ying Lin. A robust em clustering algorithm for gaussian mixture models. *Pattern Recognition*, 45(11):3950–3961, 2012.

[3] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[4] Nir Friedman and Moises Goldszmidt. *Learning Bayesian networks from data*. Morgan Kaufmann, 1999.

[5] Nir Friedman et al. Learning belief networks in the presence of missing values and hidden variables. In *ICML*, volume 97, pages 125–133, 1997.

[6] David Heckerman. A tutorial on learning with bayesian networks. In *Learning in graphical models*, pages 301–354. Springer, 1998.

[7] Peter Cheeseman, James Kelly, Matthew Self, John Stutz, Will Taylor, and Don Freeman. Autoclass: A bayesian classification system. In *Machine Learning Proceedings 1988*, pages 54–64. Elsevier, 1988.

[8] Andrew Moore. K-means and hierarchical clustering, 2001.

[9] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

[10] Nir Friedman. The bayesian structural em algorithm. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 129–138. Morgan Kaufmann Publishers Inc., 1998.

[11] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[12] John F Lemmer. The causal markov condition, fact or artifact? *ACM SIGART Bulletin*, 7(3):3–16, 1996.

[13] Bo Thiesson, Christopher Meek, David Maxwell Chickering, and David Heckerman. Learning mixtures of bayesian networks. In *in Cooper & Moral*. Citeseer, 1997.

[14] Remco R Bouckaert. Probabilistic network construction using the minimum description length principle. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 41–48. Springer, 1993.

[15] Dennis Wackerly, William Mendenhall, and Richard L Scheaffer. *Mathematical statistics with applications*. Nelson Education, 2007.

[16] Kenneth E Train. *Discrete choice methods with simulation*. Cambridge university press, 2009.

[17] Aakash Soor and Vikas Mittal. An improved method for robust and efficient clustering using em algorithm with gaussian kernel. *International Journal of Database Theory and Application*, 7(3):191–200, 2014.

[18] Arthur Asuncion and David Newman. Uci machine learning repository, 2007.