Universidad Politécnica
de Madrid

Escuela Técnica Superior de
Ingenieros Informáticos

Máster Universitario en Ciencia de Datos

Trabajo Fin de Máster

Estudio Empírico de Algoritmos de Aprendizaje de Redes Bayesianas a partir de Datos

Autor(a): Laura Cerro Gutiérrez

Tutor(a): Juan Antonio Fernández Del Pozo De Salamanca

Madrid, enero 2023

Este Trabajo Fin de Máster se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Máster
Máster Universitario en Ingeniería Informática
Título: Empirical Study of Learning Algorithms of Bayesian Network from Data

Enero 2023

Autor(a): Laura Cerro Gutiérrez

Tutor(a): Juan Antonio Fernández Del Pozo De Salamanca

Inteligencia Artificial
ETSI Informáticos
Universidad Politécnica de Madrid

Resumen

Este trabajo de fin de máster tiene como objetivo el estudio empírico de los modelos gráficos que estudian un conjunto de datos mediante independencias probabilísticas o condicionales de sus variables, llamados Redes Bayesianas.

Se comenzará estudiando la estructura de la Red Bayesiana y sus conceptos más importantes, como la independencia entre nodos y la dirección de los arcos.

A continuación, se aplicarán técnicas de aprendizaje de parámetros para conocer las distribuciones de probabilidad condicional de cada nodo a través de algoritmos de aprendizaje de parámetros como el algoritmo de maximización de verosimilitud. Para ello se usarán diferentes conjuntos de datos, así como diferentes gráficos de redes.

Se aprenderá la estructura de las redes Bayesianas dados los datos usando diferentes algoritmos, como los algoritmos basados en pruebas de independencia condicional o puntuación.

Por último, se compararán las redes aprendidas con las redes originales, tanto su estructura a través de medidas como la distancia estructural de Hamming, como su distribución de probabilidad condicional aprendida, a través de métodos como la divergencia de Kullback-Leibler.

Se realizarán tres experimentos con la diferencia de que cada uno tiene un número diferente de variables, el primero es una red Bayesiana pequeña con menos de 10 variables, el segundo es una red Bayesiana mediana con 30 variables y el tercero una red Bayesiana grande con más de 80 variables.

El objetivo principal es conocer qué algoritmos de aprendizaje de estructuras son los que mejor representan la estructura de la red comparándola con una red Bayesiana previamente dada, por lo que este trabajo nos proporcionará una serie de algoritmos que a priori son los que mejor aprenden una estructura de red dependiendo del tamaño de esta.

Abstract

The aim of this master's thesis is the empirical study of the graphical models that study a set of data by studying the probabilistic or conditional independence of its variables, called Bayesian networks.

We will begin by studying the structure of the Bayesian network and its most important concepts, such as the independence between nodes and the direction of the arcs.

Then, parameter learning techniques will be applied to learn the conditional probability distributions of each node through parameter learning algorithms such as the likelihood maximization algorithm. Different data sets will be used for this purpose, as well as different network graphs.

The structure of Bayesian networks given data will be learned using different algorithms, such as algorithms based on conditional independence tests or scoring.

Finally, the learned networks will be compared with the original networks, their structure, through measures such as the Hamming structural distance, and their learned conditional probability distribution, through methods such as Kullback-Leibler divergence.

Three experiments will be performed with the difference that each one has a different number of variables, the first one is a small Bayesian network with less than 10 variables, the second one is a medium Bayesian network with 30 variables and the third one is a large Bayesian network with more than 80 variables.

The main objective is to know which structure learning algorithms best represent the structure of the network by comparing it with a previously given Bayesian network, so this work will provide us with a series of algorithms that a priori best learn a network structure depending on the size of the network.

Tabla de contenidos

Referencias de Tablas

Referencias de Ecuaciones

Referencias de Figuras

1 Introducción

Las redes Bayesianas son modelos gráficos probabilísticos a través de los cuales podemos representar la realidad e interpretarla. Los modelos probabilísticos usan la teoría de la probabilidad para cuantificar la incertidumbre existente en un problema determinado. El uso de representaciones gráficas facilita la interpretación e interoperabilidad de la estructura de un modelo probabilístico y permite representar las relaciones entre las diferentes variables independientemente de las distribuciones de probabilidad de cada una de ellas.

La representación gráfica de una red Bayesiana viene dada por la Teoría de Grafos. La teoría de grafos estudia las propiedades de los grafos. La Teoría de Grafos define y muestra las características de un grafo, por lo que la representación cualitativa de las redes Bayesianas se basa en ella.

Las redes Bayesianas se componen de la parte cualitativa o grafos dirigidos acíclicos (DAGs), que representa gráficamente las posibles variables a través de nodos y las relaciones de dependencia entre ellas a través de arcos. La parte cuantitativa estudia las distribuciones de probabilidad condicionada de cada variable, que reproducen las creencias de las relaciones causa efecto entre las variables.

Uno de los problemas más estudiados en este tipo de modelos es su aprendizaje. El aprendizaje de redes Bayesianas se divide en dos, en primer lugar aprender la parte cualitativa o lo que es lo mismo la estructura de la red usando algoritmos de aprendizaje de estructuras y en segundo lugar aprender la parte cuantitativa o los parámetros de la red usando algoritmos de aprendizaje de parámetros.

El aprendizaje es una tarea compleja, para ello hay que tener en cuenta tres cuestiones importantes; la forma en la que se tiene la información, el número de variables que tienen los datos y el tipo de datos que contienen las variables.

Los modelos probabilísticos tienen como característica que pueden ser aprendidos a través de datos pero también usando el conocimiento experto de una persona en el tema, o usando una combinación entre ambos. Dependiendo de la información a priori que se tenga el aprendizaje podrá ser:

1- Aprendizaje basado en conocimiento experto del dominio: Se basa en mecanismos causales que a través de una modelización darán lugar a un gráfico causal del cual se tiene información de las probabilidades condicionales de las variables que representa. Se pueden generar datos aleatorios a partir de esta información para evaluar la red usando ecuaciones estructurales. [1]

2- Aprendizaje basado en datos: Se parte de unos datos previamente procesados y se usan algoritmos de aprendizaje tanto de estructura como de parámetros para crear la red Bayesiana.

3- Aprendizaje combinado: Para este aprendizaje hay dos posibilidades:

- El conocimiento experto nos proporciona la estructura, y a través de los datos aprendemos los parámetros de la red Bayesiana.
- El conocimiento experto nos proporciona la tabla de probabilidades, y a través de los datos se aprenderá la estructura de la red.
- A través de los datos se aprende su estructura y el modelo de probabilidad, y se amplia el conocimiento a través de un experto. El experto aporta información del contexto de los datos, tiene una visión general y puede evaluar su calidad y proponer nuevas variables, relaciones o eliminar algunas generadas por los algoritmos.

La segunda cuestión importante para el aprendizaje de redes es el número de variables. Dependiendo del número de variables la red contará con mayor o menor número de nodos y arcos entre ellos.

Según el número de variables o nodos las redes Bayesianas se pueden clasificar en:

- Redes pequeñas: son aquellas que tienen menos de 20 nodos o variables.
- Redes medianas: son aquellas que tienen entre 20 y 50 nodos o variables.
- Redes grandes: son aquellas que tienen entre 50 y 100 nodos o variables.
- Redes masivas: son aquellas que tienen más de 1000 nodos.

Cuanto más grande sea la red mayor complejidad tendrá y por lo tanto más difícil de aprender será.

Hay que tener en cuenta que si el número de arcos incidentes o el grado de las variables, es decir, el número de líneas que pasan por un nodo en el gráfico es muy alto más parámetros tendrá la tabla de probabilidad condicionada y por lo tanto, más compleja será.

Otro de los puntos importantes es la tipología de los datos. Las redes Bayesianas dependiendo del tipo de datos que contengan se clasifican principalmente de la siguiente manera [2]

- Multinomial Bayesian Networks: Son redes Bayesianas que usan datos discretos, y en las que se pondrá el foco en este trabajo
- Gaussian Bayesian Networks: Son redes Bayesianas que usan datos continuos.
- Conditional Gaussian Bayesian Networks: Son redes Bayesianas que usan datos mixtos, es decir, las variables podrán ser tanto discretas como continuas.
- Dynamic Bayesian Networks: Son redes Bayesianas que usan datos continuos y están especializadas en el estudio de series temporales.
- Hybrid Bayesian Networks: Son redes Bayesianas más complejas que son útiles para el estudio de tiempos de espera.

El aprendizaje de redes Bayesianas dependerá de estos tres bloques de características tanto para la elección del algoritmos, como las métricas de evaluación y comparación de redes.

1.1 Objetivos

El objetivo principal de este trabajo es estudiar el aprendizaje de redes Bayesianas en datos con diferentes número de variables. Además se cuenta con las estructuras de redes originales con las cuales se realizará el aprendizaje de la estructura de la red, por lo que el aprendizaje será combinado con el conocimiento experto.

Se inicia con tres bases de datos con diferentes números de variables. Las tres bases de datos cuentan con variables de tipo discreto, por lo que este trabajo se centrará exclusivamente en redes Bayesianas multinomiales.

Los objetivos para cada conjunto de datos son:

1. Estudio descriptivo de la red original para un mejor conocimiento prior de la estructura de los datos.

2. Aplicación de las técnicas de aprendizaje de parámetros a través de los datos.

4. Aplicación y evaluación de los algoritmos de aprendizaje de estructuras

3. Comparación de los resultados obtenidos con las redes originales.

La herramienta de programación que se usará es RStudio [3], para la visualización de las redes Bayesianas, el aprendizaje y la comparación de redes. Los paquetes que se usarán son: bnlearn [4] , gRain [5], visNetwork y seewaves.

2 Estado del arte

En esta parte del trabajo se hará un repaso de los conceptos teóricos que son actualmente las bases de las redes Bayesianas y su aprendizaje.

2.1 Definición y representación de Redes Bayesianas

Las redes Bayesianas son modelos gráficos probabilísticos que relacionan las dependencias probabilísticas entre dos o más variables aleatorias combinando la teoría de grafos con la teoría de la probabilidad de manera simultánea. [6]

Sus dos componentes son:

- Una estructura de red o DAG $G = (V, E)$ en el que cada nodo $v_i \in V$ corresponde a una variable aleatoria X_i que se relacionan a través de arcos $e_{ij} \epsilon E$.
- Una distribución de probabilidad global o conjunta X con parámetros Θ que puede ser factorizada en pequeñas distribuciones condicionadas o locales acordes con los arcos $e_{ij} \epsilon E$ representados en el gráfico.

La estructura de red se define como un grafo acíclico dirigido $G = (V, E)$ siendo $V = \{v_i, \dots, v_N\}$ un conjunto finito de nodos y E un conjunto de arcos identificados por sus pares de nodos en V, $e_{ij} = (v_i, v_j)$. Un grafo acíclico dirigido o DAG es un gráfico que tiene las siguientes características:

- Solo contiene arcos directos, esto ocurre si $(v_i, v_j) \neq (v_j, v_i)$ es un par ordenado y el arco tiene una dirección específica $v_i \rightarrow v_j$
- No contiene bucles, un bucle ocurre cuando un arco va desde un nodo a el mismo nodo $v_i \rightarrow v_i$
- No contiene ciclos, un ciclo ocurre cuando una secuencia de arcos $v_i \rightarrow v_i \rightarrow \cdots \rightarrow v_k \rightarrow v_i$ empieza y acaba en el mismo nodo.

En la *figura 1* hay un ejemplo de un gráfico acíclico dirigido o lo que es lo mismo, una estructura de una red Bayesiana.

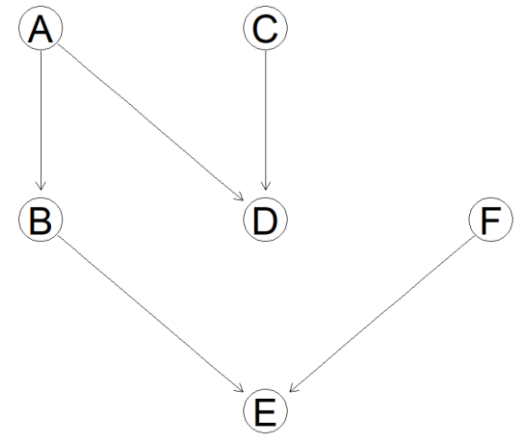

Figura 1 Ejemplo de DAG

La distribución de probabilidad conjunta se especifica mediante distribuciones marginales (modelos globales) y condicionales (modelos locales), teniendo en cuenta las relaciones de independencia condicional entre las variables.

La probabilidad conjunta, que requiere un numero de parámetros exponencial en el número de nodos. Sin embargo, usando la factorización se reducirá el número de parámetros, lo que derivará en reducir la

complejidad de la red eliminando arcos. La factorización sigue la regla de la cadena y la teoría probabilística de independencia condicional.

Definición 1.1: Factorización. Dada una red Bayesiana, los nodos representan variables aleatorias del dominio $X_1, X_2, \dots, X_n$ y los arcos representan relaciones de dependencia entre variables. Las redes Bayesianas asumen que un nodo depende solamente de sus padres y que cada nodo esta asociado a una distribución condicionada de probabilidad, que definen la probabilidad de cada estado en los que puede estar una variable, dados los posibles estados de sus padres. Por lo tanto, x_i representa el valor que toma la variable X y Π_{X_i} denota los valores que tienen el conjunto de los padres en la red Bayesiana del nodo X_i.

Finalmente la distribución conjunta definida por un grafo es dada por el producto de las probabilidades condicionales con respecto a sus padre, que especifica la factorización de la distribución global:

Ecuación 1 Factorización de distribuciones de probabilidad conjunta

$$P(x_1, x_2, \dots, x_n) = \prod_{i=1..n} P\left(X_i | \Pi_{X_i}; \Theta_{X_i}\right)$$

$$\text{Donde } \Pi_{X_i} = \{padres\ de\ X_i\}$$

Cada arco indica dependencia probabilística entre nodos, aunque esto no tiene porque indicar causalidad.

Una red Bayesiana representa relaciones de independencia condicional entre las variables. La independencia entre variables puede ser condicional o marginal.

Definición: 1.2 Independencia. Una variable X tiene una relación de independencia condicional con Y dado Z, $I_p(X, Y|Z)$, si se cumple que

Ecuación 2 Definición de probabilidad condicionada

$$P(x|y, z) = P(x|z).$$

Una variable X es marginalmente independiente de Y si

Ecuación 3 Definición de independencia

$$P(x|y) = P(x) \leftrightarrow P(x, y) = P(x)P(y).$$

En la *figura 2* se muestra un ejemplo sencillo de relación de dependencia entre nodos.

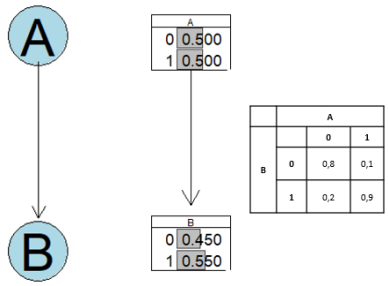

Figura 2 Ejemplo relación de dependencia entre nodos.

Dos nodos que no están unidos directamente pueden o no ser probabilísticamente independientes, esto depende de la estructura del grafo y del estado en que se encuentren los nodos (observados o no observados). Existen diferentes relaciones de dependencia entre nodos que se pueden resumir con las siguientes estructuras principales de la *figura 3*:

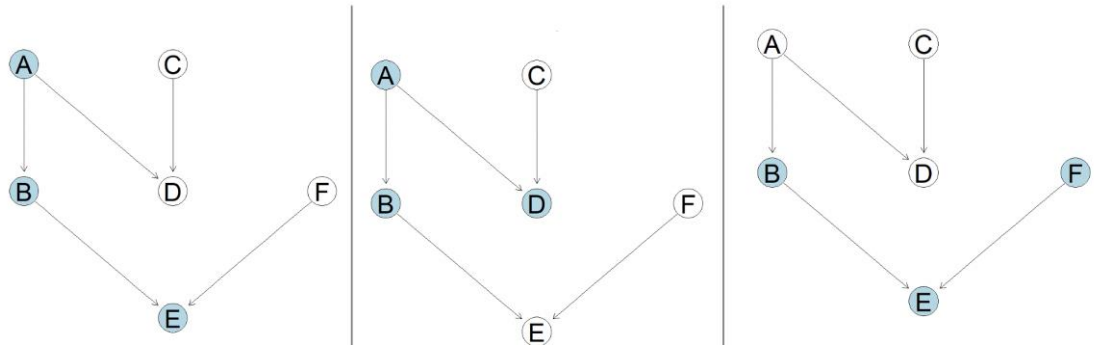

Figura 3 Ejemplo estructuras. De derecha a izquierda: V-estructura, Divergencia y Cascada

En la estructura en cascada los nodos A y E son dependientes entre si, pero si condicionamos el nodo B pasan a ser independientes dado el nodo B.

$$A \perp E \mid B$$

En la estructura divergente pasa algo parecido, el nodo A contiene todo la información que determina el estado de los nodos B y D, por lo tanto, separa entre si probabilísticamente a los otros nodos.

$$B \perp D \mid A$$

En los dos anteriores casos los nodos de los extremos no compartían un nodo hijo, esto es lo que ocurre en la estructura en V. Los nodos B y F son independientes entre si, pueden tomar cualquier valor independientemente del otro nodo. Pero si se conoce el estado del nodo hijo E, los nodos padres B y F pasan a ser dependientes, es decir, compiten entre sí para explicar el estado del nodo E.

$$B \not\perp F \mid E$$

El concepto de separación de dependencias probabilísticas se extiende a una red Bayesiana mayor usando el concepto de separación-d. Las independencias derivadas quedan identificadas de forma gráfica mediante el concepto de d-separación, que verifica la independencia condicional.

Definición 1.3. Separación-D. Dos variables X e Y se dicen que están d-separadas si todos los caminos no dirigidos entre X e Y están inactivos dados un nodo observado Z, excepto en las estructuras V. [7]

El concepto de separación-d corresponde con el de independencia condicional. Por lo tanto, dos variables o conjunto de variables X e Y serán condicionalmente independientes dada una variable o conjunto de variables Z si y solo si Z d-separa a X e Y.

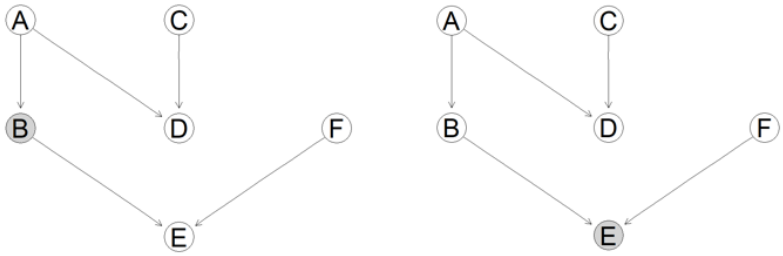

Figura 4 Ejemplos D-Separación

En la *figura 4* los nodos A y E están d-separados porque el nodo observado B está bloqueando todo los posibles caminos entre ambos nodos. En cambio, en el segundo gráfico los nodos A y F no están d-

separados ya que el nodo E al estar observado y formar una v-estructura permite que exista un camino que une A y F.

Un concepto importante para resolver el problema de selección de variables es el manto de *Markov* o *Markov Blanket*, dado un objetivo identifica las variables que influyen probabilísticamente en cierta variable objetivo. Un *manto de Markov* es el conjunto de nodos que lo hacen independiente del resto de la red y está formada por los nodos padre, nodos hijo y otros padres de los hijos. Además, una estructura en V siempre será parte de un *manto de Markov.*

Definición 1.4: *Manto de Markov*. Para cualquier variable $X \in V$, el *manto de Markov* $BL(X) \subseteq V$ es cualquier conjunto de variables para cualquier $Y \in V - BL(X) - \{X\}, X \perp Y | BL(X)$. En otras palabras BL(X) protege completamente (d-separa) la variable X de cualquier variable fuera del *manto de Markov* $BL(X) \cup \{X\}$.

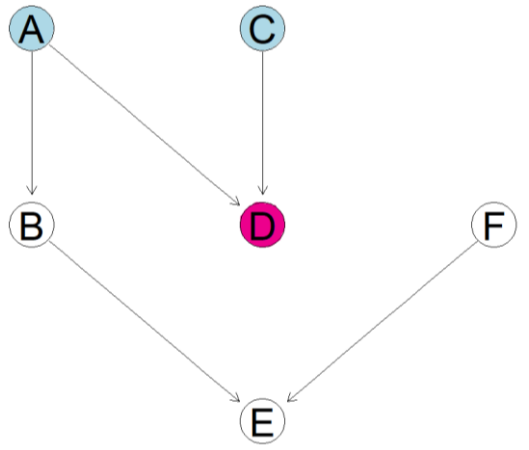

Figura 5 Ejemplo de manto de Markov

En la *figura 5* tenemos un ejemplo, tomando el nodo objetivo D vemos que su manto de Markov está formado por los nodos A y C, que en este caso son sus nodos padres. Estos nodos d-separan el nodo D del resto de nodos B, F y E.

Para entender la unión de la independencia entre variables de forma probabilística como de forma grafica usamos el teorema de separación.

Definición 1.5: Teorema de separación. Según el teorema de separación $X \perp Y \mid Z$ dada una distribución de probabilidad P de las variables en V={X, Y, Z} y $G = (V, E)$, si (G,P) cumple la condición de Markov en la que $X \perp_G Y|Z \implies I_p(X, Y|Z) \quad \forall X, Y \subseteq V$. X es independiente de Y dado la d-separación definida en G, lo que nos indica que X e Y son condicionalmente independientes dado P. [8]
El gráfico G representan todas las dependencias de P $\neg I_p(X, Y|Z) \implies \neg(X \perp_G Y|Z)$ aunque algunas independencias de P podrían no ser identificadas por la separación d en G.

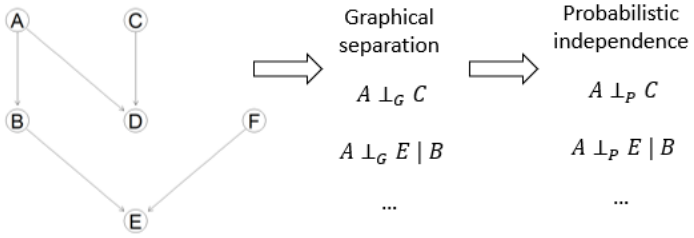

Figura 6 Ejemplo de Mapa-P

Dada una distribución de probabilidad P y una representación gráfica de dependencias G debe existir una correspondencia entre las independencias representadas en ambos. Podemos tener el caso en que las

variables independientes están separadas en el grafo, el caso donde las variables separadas en el grafo son independientes y el caso en el que se cumplen ambas cosas. No es siempre posible tener un grafo perfecto ya que hay distribuciones con relaciones de independencia que no se pueden representar como un DAG.

Las estructuras de las redes Bayesianas explican de manera gráfica una distribución de probabilidad, pero esta distribución no es única. Toda distribución de probabilidad conjunta sobre n variables aleatorias puede factorizarse de n! formas y escribirse como producto de las distribuciones de probabilidad de cada una de las variables condicionadas a otras variables. Cada una de estas factorizaciones puede representarse mediante una red Bayesiana, esto se puede observar en la *figura 7*:

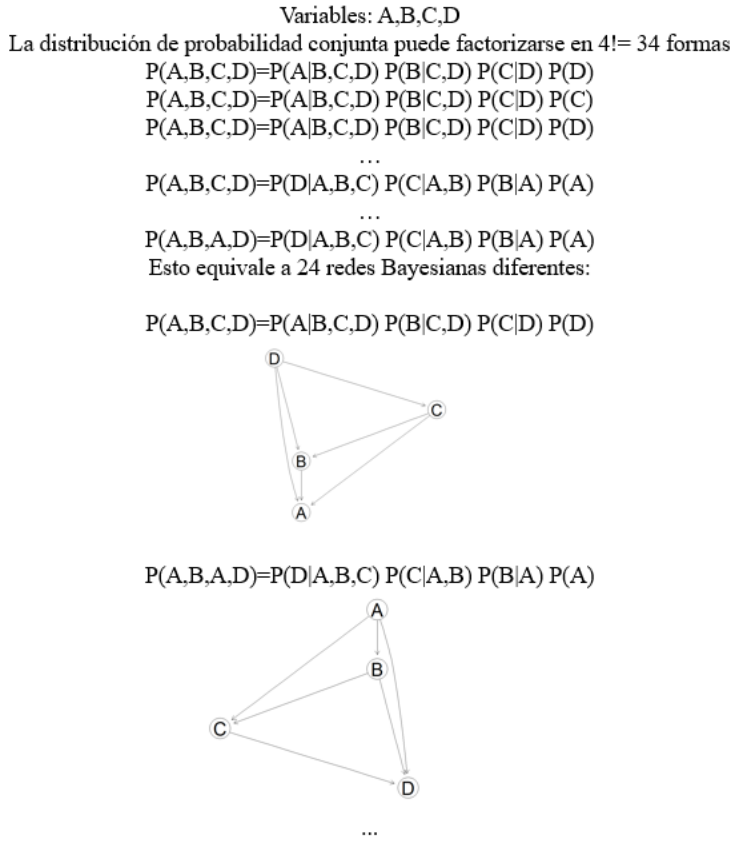

Figura 7 Ejemplo factorización distribución de probabilidad conjunta

Como anteriormente se ha comentado, existen varias factorizaciones de una distribución de probabilidad. Un DAG identifica de forma única una factorización de P(X), pero una factorización no identifica de forma única un DAG.

Dos DAGs pueden tener la misma distribución pero diferente orden topológico, estas redes serán equivalentes si y solo si los DAGs asociados tienen el mismo esqueleto y V-estructuras.

Cuando un DAG es probabilísticamente equivalente a otro, podemos invertir las direcciones de sus arcos como queramos siempre que no creamos una V-estructura nueva. Esto significa que se pueden agrupar los DAGs en clases de equivalencia que son identificadas de forma única por el grafo no dirigido subyacente y las V-estructuras. Las direcciones de otros arcos pueden ser:

- Identificables de forma única porque una de las direcciones introduciría ciclos o nuevas estructuras V el grafo (arcos obligados).
- Completamente indeterminado.

El resultado de esto es lo que se llama grafo parciamente dirigido completo (CPDAG).

Definición 1.6: CPDAG. Un CPDAG es un grafico completo parcialmente directo y acíclico, es una clase de equivalencia de DAGs. La dirección causal se muestra si todos los miembros de la clase de equivalencia

coinciden, la dirección causal puede ser ambigua si hay desacuerdo interno entre los miembros de la clase de equivalencia. [9]

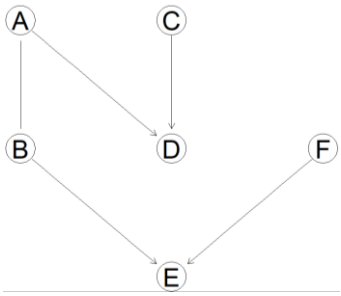

Figura 8 Ejemplo de CPDAG

La estructura de la red Bayesiana también puede interpretarse numéricamente mediante una matriz adyacente. Una matriz adyacente es una matriz cuadrada booleana que se utiliza para representar un gráfico finito. Los elementos de la matriz indican si los pares de vértices son adyacentes o no en el grafo. Para un gráfico dirigido la matriz está formada por 0 y 1 en su diagonal.

En resumen, el gráfico de una red Bayesiana puede considerarse como una estructura de datos que provee el esqueleto para representar la distribución de probabilidad conjunta en su versión factorizada, y como una representación de un conjunto de supuestos sobre la independencia condicional de la distribución de probabilidad.

Además, se puede medir la fuerza de las relaciones representadas en cada arco por una red Bayesiana. Esto será útil para eliminar relaciones que no sean significativas y por lo tanto reducir la complejidad de la red, además de validar la estructura de la red.

Las principales medidas que se pueden usar son:

- Pruebas de independencia condicional que eliminarán los arcos individuales en una red. El p-valor mide el grafo de confianza de cada arco, un valor pequeño del p-valor nos indicará que la relación entre los nodos es fuerte, por lo tanto ese arco entre ellos debe permanecer.
- Nivel de una función de puntación eliminación los arcos individuales presentes. Un valor negativo del p-valor nos indica que esa conexión entre arcos disminuye la puntuación de la red.
- Estimar la fuerza de cada arco como su frecuencia empírica sobre un conjunto de redes aprendidas a partir de muestras *Bootstrap* [10]. Identifica las probabilidades de inclusión de todos los arcos posibles y sus direcciones.
- Estimar la fuerza de cada arco utilizando factores de Bayes. Identifica las probabilidades de inclusión de todos los arcos posibles y sus direcciones.

Finalmente es importante recalcar que dependencia no es sinónimo de causalidad, por lo que la causalidad de las variables solo puede ser definida y verificada por el conocimiento experto. Si en nuestro modelo contamos con una variable X_i que es dependiente de otra X_j, no podemos aceptar que existe una relación causal entre X_i y X_j.

Una vez se tenga la estructura es importante estudiar las distribuciones de probabilidad de cada variable, así como, los parámetros asociados a cada nodo y en total para comprobar la complejidad numérica de nuestra red Bayesiana. El número de parámetros de una red Bayesiana discreta se define como la suma del número de parámetros independientes de cada nodo dado sus padres. [11]

Las redes creadas teniendo en cuenta el conocimiento experto hacen que las relaciones causales sean más compactas, maximizando las independencias condicionales sin arcos innecesarios. Aunque las técnicas de minería de datos pueden ayudar en la búsqueda de redes, pero generalmente necesitan ser complementadas por los expertos del dominio, bien añadiendo o borrando arcos, bien definiendo o corrigiendo direcciones de relaciones causales.

2.2 Algoritmos de aprendizaje de Redes Bayesianas

Estudiaremos los dos bloques de algoritmos de aprendizaje, en primer lugar los relacionados con la estructura de la red y en segundo lugar los relacionados con los parámetros.

2.2.1 Aprendizaje de estructuras

El aprendizaje de estructuras se define como el entrenamiento, a partir de un conjunto de datos $\mathcal{D}$, de la estructura G que define las relaciones de independencia condicional de las variables del modelo $\mathcal{X}$. El aprendizaje estructural pretende recuperar una estructura G' que factoriza la distribución de probabilidad conjunta desconocida de las variables $P'(\mathcal{X})$. [12]

En el aprendizaje de la estructura de una red Bayesiana podemos encontrarnos dos dificultades principales:

- Los datos disponibles contienen mucho ruido, lo que puede dar lugar a la generación de una estructura de red imprecisa $(\mathrm{G} \neq \mathrm{G}')$
- La estructura es demasiado grande; el número de grafos aumenta exponencialmente a medida que crece el número de nodos.
- Algoritmos con alto coste computacional e incluso no tratables para un número alto de variables.

Una de las soluciones para trabajar con redes Bayesianas muy grandes es el uso de algoritmos paralelos [11] que utiliza procesos en los que cada uno gestiona un subconjunto de datos sobre todas las variables, aunque en este trabajo no se entrará en detalle sobre ellos.

Los tres métodos de aprendizaje más importantes para redes Bayesianas discretas son los siguientes [14]:

1- Aprendizaje de estructuras basadas en restricciones.: estos algoritmos usan pruebas estadísticas para evaluar las relaciones de independencia condicional en las variables y buscan una clase equivalente de estructuras que las cumpla, bajo el supuesto de que la independencia condicional implica la separación gráfica. Su principal desventaja es la alta sensibilidad a errores debido a los errores de las pruebas de independencia.

2- Aprendizaje de estructuras basada en puntuaciones: son algoritmos de optimización que clasifican las estructuras de red propuestas basados en puntuaciones. En primer lugar definen un espacio de búsqueda de modelos, que define el conjunto de estructuras que van a ser evaluadas, a continuación, se define una función de puntuación de bondad de ajuste, que puntúa como de bien se ajusta el modelo a los datos y por último un algoritmo de búsqueda que permitirá recorrer el espacio de búsqueda.

3- Aprendizaje de estructuras híbrido: combinan aspectos de los dos algoritmos anteriores. Usan pruebas de independencia condicional para reducir el espacio de búsqueda, y puntuación de redes para encontrar la red más optima en el espacio ya reducido, usando estas dos técnicas al mismo tiempo.

En este trabajo, se elegirán varios algoritmos de cada grupo y se aplicarán a las muestras de datos.

2.2.1.1 Aprendizaje de estructuras basados en restricciones

Los algoritmos basados en restricciones usan pruebas estadísticas para aprender las relaciones de condiciones de independencia desde los datos y asume que el DAG es un mapa perfecto que determina la estructura correcta de la red. Este enfoque da lugar a un conjunto de restricciones de independencia que identifican una única clase de equivalencia.

Los algoritmos basados en test de independencia tratan las redes Bayesianas como mapas perfectos:

$$A \perp B|C \Leftrightarrow A \perp_G B|C$$

La principal contra de esto es que no todos las distribuciones de probabilidad de una variable X tienen un DAG que refleje fielmente esta distribución.

2.2.1.1.1 Pruebas de independencia

Las pruebas de independencia condicional que usamos para aprender una red Bayesiana son funciones de las frecuencias observadas $\{n_{ijk}, i = 1, \dots, R, j = 1, \dots, C, k = 1, \dots, L\}$ para las variables aleatorias X e Y, y todas las configuración de la variable condicionada Z. Estos son dos de los test que usaremos:

- Información mutua condicional. Es una medida de asociación basada en la teoría de la información. Cuantifica la información de dos variables aleatorias X e Y observando el valor de Z.

Ecuación 4 Formula información mutua condicional

$$MI(X,Y|Z) = \sum_{i=1}^{R}\sum_{j=1}^{C}\sum_{k=1}^{J}\frac{n_{ijk}}{n} log\frac{n_{ijk}n_k}{n_{ik}n_{jk}}$$

Siendo n_{ijk}, n_{ik}, n_k y n el número de observaciones en cada nivel de las variables.

Esta prueba de independencia contrasta la hipótesis de que el estadístico del test sea menor a 0, frente a la hipótesis alternativa de el valor del estadístico es mayor a 0. Un valor del estadístico de 0 indica que X e Y son independientes dado 0

- Prueba *chi-cuadrado de Pearson*. Es una prueba no paramétrica que mide la independencia de la variable X sobre Y dada la variable Z mediante la presentación de los datos en tablas de contingencia.

Ecuación 5 Formula Pearson Chi-cuadrado

$$X^2(X,Y|Z) = \sum_{i=1}^{R}\sum_{j=1}^{C}\sum_{k=1}^{L}\frac{\left(n_{ijk} - m_{ijk}\right)^2}{m_{ijk}}$$

$$Donde\ m_{ijk} = \frac{n_{ik}n_{jk}}{n_{k.}}$$

Esta prueba de independencia contrasta la hipótesis de que las variables son independientes, frente a la hipótesis alternativa de que una variable se distribuye de modo diferente para diversos de la otra.

Las hipótesis son las siguientes:

$$H_o: Las\ variables\ son\ independientes$$

$$H_a: Las\ variables\ no\ son\ independientes$$

Aquellas pruebas que tenga un p-valor menor a 0.05 nos indican que se rechaza la hipótesis nula y por lo tanto no podemos afirmar que las variables sean independientes. Las pruebas que tenga un p valor mayor a

0.05 nos indica que no se rechaza la hipótesis nula y por lo tanto podemos interpretar que las variables son independientes dada un subconjunto de variables.

Para cada algoritmo de aprendizaje identificamos los argumentos de los parámetros de ajustes y son iguales para toda función de aprendizaje:

- Los datos para entrenar el algoritmo.
- *Test*: El test estadístico de independencia condicional que usaremos
- *Alpha*: El umbral de error de tipo I para las pruebas de independencia condicional individuales. Usaremos un alpha > 0.5 para restringir la elección de los nodos condicionalmente dependientes.
- *Debug*: Argumento que nos permite imprimir los pasos realizados por el algoritmo

Se usará como argumento *test* la información mutua condicional y el estadístico *chi-cuadrado*. Se usará como argumento *alpha* un valor del 0.5. Por último, se usará el argumento *debug=true* para la interpretación del modelo.

2.2.1.1.2 Algoritmos

- Algoritmo *Peter y Clark* (PC- estable):

Es un algoritmo de aprendizaje de estructuras basado en restricciones, fue diseñado para el aprendizaje de grafos acíclicos dirigidos (DAG) bajo el supuesto de suficiencia causal, es decir, sin causas comunes no medidas y sin variables de selección [15]. Este algoritmo se basa en pruebas de independencia entre variables $I(X_i, X_J|Z)$ donde Z es un subconjunto de variables.

A continuación, se explicarán paso a paso los detalles de este algoritmo:

1. Dado un conjunto de datos sobre un conjunto de variables, se parte de un gráfico completo G. Mediante una serie de pruebas estadísticas sobre cada par de variables, se eliminan los ejes que unen dos nodos X-Y si se ha verificado un conjunto condicional de independencia y separación $I(\mathrm{X} \perp \mathrm{Y}|Z)$. En primer lugar aquellas de orden 0, después las de orden uno y así sucesivamente. Cuando el algoritmo PC se aplica a los datos suele ser dependiente del orden, es decir, el resultado depende del orden en que se dan las variables.

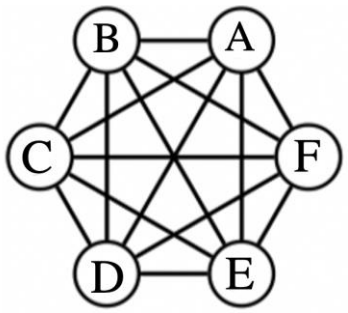

Figura 9: Paso 1.1 Algoritmo PC: Crear gráfico completo

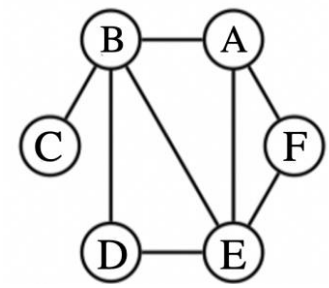

Figura 10 Paso 1.2 Algoritmo PC: Esqueleto del grafico

2. El gráfico unidireccional resultante se llama esqueleto e identificaremos la estructura en v si se cumple que $I(\mathrm{X} \perp \mathrm{Y}|Z)$. El resultado será un completado gráfico acíclico parcialmente directo (CPDAG) donde los ejes deberán ser transformados en arcos.

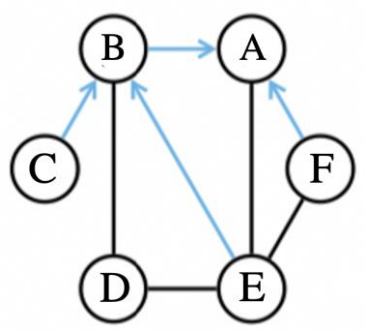

Figura 11: Paso 2 algoritmo PC: Identificación estructuras en V

3. El último paso será determinar la orientación de los ejes entre los nodos. Supuestos adicionales (por ejemplo, la aciclicidad) permiten la propagación de las orientaciones de la estructura-V a algunas de las aristas no dirigidas restantes. [16]

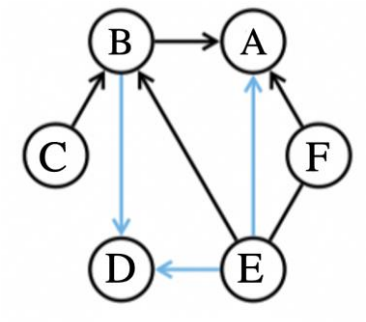

Figura 12: Paso 3 algoritmo PC: Propagación

Este algoritmo ayuda fundamentalmente a encontrar grupos de variables fuertemente relacionadas, usando las pruebas de independencia previamente explicadas.

- Algoritmo Grow Shrink

El algoritmo *Grow Shrink* es el primer algoritmo de detección de *Markov Blanket* utilizado en aprendizaje de estructuras basado en restricciones [17]. Este algoritmo consiste en aprender la manta de *Markov* de cada nodo de la red con el objetivo de simplificar la identificación de los vecinos de cada nodo. Por lo tanto, el número de pruebas de independencia condicional necesarias para el algoritmo de aprendizaje se reducen y con ello la complejidad global del modelo.

El algoritmo *Grow-Shrink* tiene dos fases, una de crecimiento y otra de reducción:

1. Fase de crecimiento: Esta fase consiste en añadir variables al manto de *Markov* siempre que sean dependiente de X dado el contenido del manto de *Markov*. Comenzamos con un conjunto vacío y testeamos que $I(X \perp Y \mid [\,])$, si no es independiente Y se incluirá en el gráfico. Seguimos testeando la independencia de cada nuevo nodo con el nodo X condicionado a cada una de las variables que ya están incluidas en el *manto de Markov*. Finalmente obtendremos el *manto de Markov* total para la variable X.

En la primera imagen, se comienza con *manto de Ma*rkov de la variable A vacío y se testea si B es dependiente de A dado el *manto de Markov* el resultado positivo y por lo tanto se añade la variable B al *manto de Markov* de A.

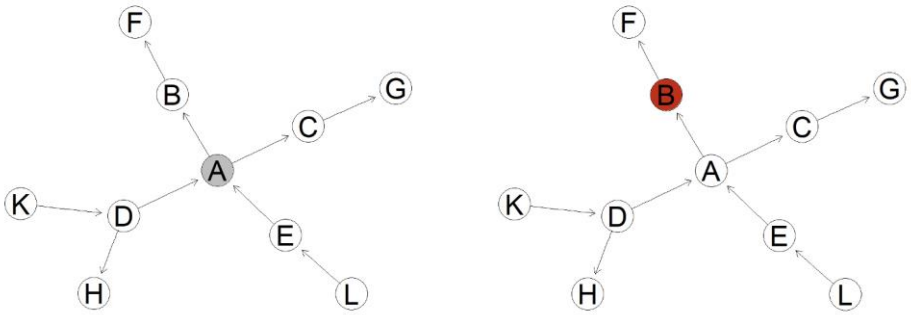

Figura 13 Fase se crecimiento algoritmo GS parte 1

En la siguiente imagen se ve que ya se han incorporado varias variables al *manto de Markov* y se quiere comprobar que la variable L esta dentro, por lo tanto se testea $A \perp L \mid \{B, G, C, K, D, H, E\}$ y el resultado es negativo. El nodo L no es dependiente de A dado su *manto de Mark*ov.

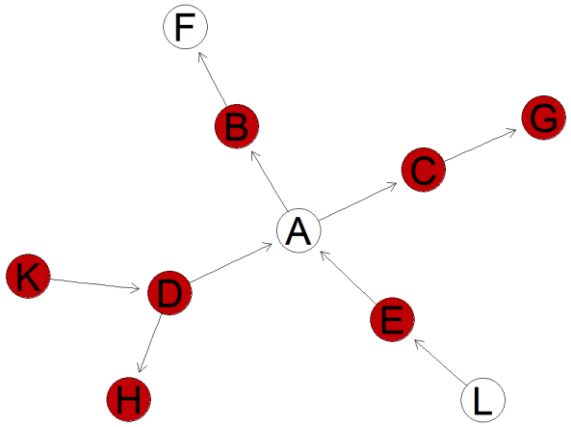

Figura 14 Fase de crecimiento algoritmo GS parte 2

2. Fase de reducción: Como la elección del orden de las variables a incluir ha sido aleatoria es posible que en la fase de crecimiento se podrían haber añadido nodos innecesarios en el *manto de Markov*, y en esta fase los eliminaremos. Comprobamos su independencia con respecto a X dado todas las variables restantes de la cadena de *Markov*, si su resultado es verdadero, eliminaremos este nodo.

Este algoritmo es muy útil a la hora de reducir los test por cada par de variables y así simplificar la complejidad del aprendizaje, se implementará este algoritmo para cada conjunto de datos y se evaluará su rendimiento.

2.2.1.2 Aprendizaje de estructuras basados en puntuaciones

Los algoritmos basados en puntuaciones se encargan de asignar a cada red Bayesiana candidata una función de puntuación, un espacio y un método de búsqueda.

Una función de puntuación es definida como la medida de ajuste entre el grafo y los datos, intenta encontrar un grafo que maximice esta función. La función de puntuación se combina con un método de búsqueda con el objetivo de medir la bondad de cada estructura explorada sobre un espacio de búsqueda de estructuras o soluciones factibles.

El espacio de estructuras donde la búsqueda se lleva acabo puede ser el espacio de un gráfico directo acíclico, el espacio de una clase equivalentes o dependiendo de un orden entre variables previamente dado. [18]

Los métodos de búsqueda que exploran de una manera inteligente el espacio de soluciones pueden ser métodos de búsqueda local o heurísticos.

Las funciones de puntuación pueden estar basados en diferentes principios como el enfoque Bayesiano o la teoría de la información. [19]

2.2.1.2.1 Funciones de puntuación

Dado un conjunto de datos $D = \{u^1, \dots, u^N\}$, encontrar un DAG G* que maximice una función de puntuación g() :

$$G^* = \arg max_{G \in G_n} g(G{:}D)$$

Donde $g(G{:}D)$ es la función de puntuación que mide el grado de ajuste de cualquier DAG G candidato, al conjunto de datos Y G_n es la familia de DAGs definida por el espacio de búsqueda.

Las funciones de puntuaciones basadas en un enfoque bayesiano se basan en el calculo de la distribución de probabilidad posterior $P(B|D)$, partiendo de una probabilidad a priori sobre las redes condicionada a el conjunto de datos D. La mejor red será aquella que maximice la probabilidad posterior.

Las funciones basadas en la teoría de la información seleccionan el grafo que mejor se ajuste a los datos penalizando el número de parámetros de la distribución conjunta.

- *Bayesian Dirichlet score* (BD)

Esta función asume que la distribución de probabilidad a priori sigue una distribución Dirichlet, que es positiva, es decir, todas las probabilidades condicionales $\pi_{ijk} > 0$ y que sus parámetros son independientes.

Ecuación 6 Función de puntuación Bayesian Dirichlet (BD)

$$g_{BD}(\mathcal{G}{:}D) = \log\big(p(G)\big) + \sum_{i=1}^{n}\left[\sum_{j=1}^{q_i}\left[log\left(\frac{\eta_{ij}}{(N_{ij}+\eta_{ij})}\right) + \sum\nolimits_{k=1}^{r_i} log\left(\frac{(N_{ijk}+\eta_{ijk})}{(\eta_{ijk})}\right)\right]\right]$$

- *Bayesian Dirichlet equivalent uniform* (BDeu)

Se asume que no tenemos conocimiento previo de la distribución a priori y por lo tanto, establecemos una distribución uniforme como prior.

Ecuación 7 Función de puntuación Bayesian Dirichlet equivalent uniform

$$g_{BDeu}(\mathcal{G}{:}D) = \log\big(p(G)\big) + \sum_{i=1}^{n}\left[\sum_{j=1}^{q_i}\left[log\left(\frac{\frac{\eta}{q_i}}{(N_{ij}+\frac{\eta}{q_i})}\right) + \sum\nolimits_{k=1}^{r_i} log\left(\frac{(N_{ijk}+\frac{\eta}{r_i q_i})}{\left(\frac{\eta}{r_i q_i}\right)}\right)\right]\right]$$

- *Akaike information criterion* (AIC)

Es una medida de bondad de ajuste de un modelo dado un conjunto de datos e incluye una función de penalización creciente en torno al número de parámetros.

Ecuación 8 Función de puntuación Akaike criterio de información

$$g_{AIC}(\mathcal{G}: D) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}\log\left(\frac{N_{ijk}}{N_{ij}}\right) - \left(\frac{1}{2}\sum_{i=1}^{n}(r_i - 1)q_i\right)$$

- *Bayesian information criterion* (BIC)

Es una medida de bondad de ajuste que usa una función de penalización mayor que el AIC, lo que provoca la selección de un modelo más sencillo

Ecuación 9 Función de puntuación del criterio de información bayesiano

$$g_{BIC}(\mathcal{G}: D) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}\, log\left(\frac{N_{ijk}}{N_{ij}}\right) - \left(\sum_{i=1}^{n}(r_i - 1)q_i\right)\frac{1}{2} log\,(N)$$

Cada una de las métricas de puntuación están basadas en una medida diferente lo que dificulta su comparación. Es posible usar varios scores usando un método multiobjetivo, pero en este estudio no pondremos el foco en ellos.

Los métodos basados en teoría de la información son más sencillos y restrictivos a la hora de elegir una red Bayesiana, además no asumen una distribución a priori. Dentro de estos, AIC sobreajusta las relaciones entre variables y BIC tiende a penalizar más y no crea estructuras de datos innecesarias.

Para cada algoritmo de aprendizaje identificamos los argumentos de los parámetros de ajustes y son iguales para toda función de aprendizaje:

- Los datos para entrenar el algoritmo.
- *Start*: El grafo acíclico directo que se usa para inicializar el algoritmo, si no se especifica por defecto es un grafo vacío.
- *Score*: La función de puntuación elegida.
- *Debug*: Argumento que nos permite imprimir los pasos realizados por el algoritmo.

Se usará el argumento *start* por defecto, es decir, comenzará el aprendizaje con un grafo vació. Se usará como argumento *score* las puntuaciones BIC, AIC y BDeu. Por último, se usará el argumento *debug=true* para la interpretación del modelo.

2.2.1.2.2 Algoritmos

- Algoritmo Hill Climbing (HC)

El algoritmo *Hill Climbing* [20] [21] es un algoritmo de búsqueda local que recorren el espacio de búsqueda de los grafos acíclicos dirigidos partiendo de una red. Utiliza la definición de vecindad mediante la adicción eliminación e inversión de un solo arco, con reinicios aleatorios para evitar los óptimos locales.

El algoritmo sigue los siguientes pasos:

1. Se define una estructura inicial que puede ser una red sin enlaces, una red aleatoria o una red previamente construida y se evalúa la función de puntuación elegida.

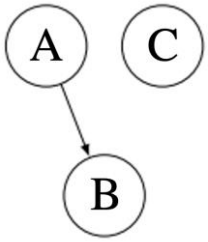

Figura 15 Paso 1 Algoritmo HC

2. Se obtienen los grafos vecinos, que será un conjunto de DAGs idénticos a la estructura pero con cambios locales realizados. Se consideran cambios locales como la adición, eliminación e inversión de un solo arco

Hay que tener en cuenta cada vez que se haga una modificación en la red y no introducir ciclos directos en el grafico. Existen $O(n^2)$ posibles cambios, siendo n el número de variables.

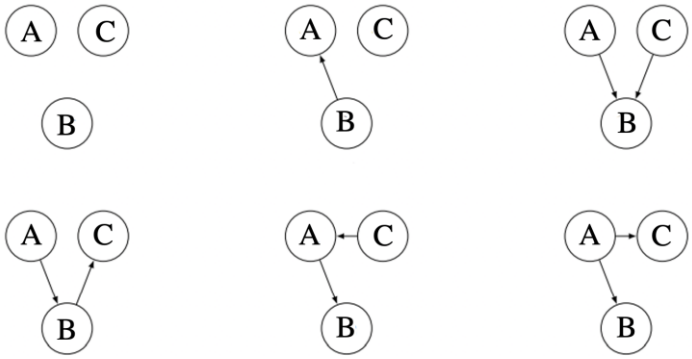

Figura 16 Paso 2 algoritmo HC

3. Se evalúan las estructuras vecinas calculando la puntuación en cada una de ellas y se selecciona aquella con la mejor puntuación, es decir, que maximice la puntuación local.

Existen diferentes estrategias que se usan para evitar el optimo local y encontrar una solución que maximice la función de puntuación globalmente, como los reinicios aleatorios o la aleatoriedad.

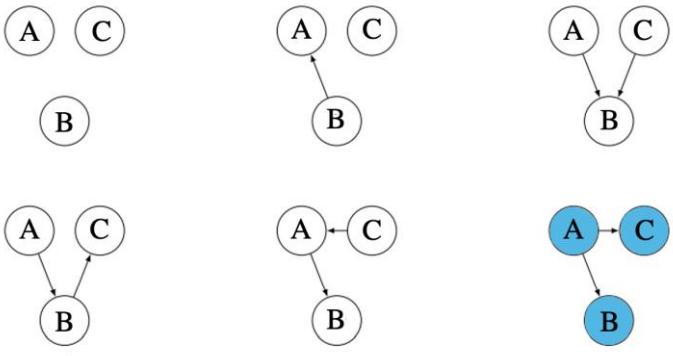

Figura 17 Paso 3 algoritmo HC

4. Se repiten los pasos 2 y 3 hasta encontrar una estructura cuyos vecinos no tengan una puntuación más elevada.

La eficiente evaluación de los grafos resultantes tras los diferentes cambios se basa en la propiedad de descomposición de las métricas de puntuación. Esta propiedad evalúa un DAG como la suma de las puntuaciones de su familia de nodos, es decir, los subgrafos de un nodo y sus padres en G.

Este tipo de métodos reutilizan los cálculos realizados en etapas anteriores y así solo deberán ser recalculados los estadísticos de las variable cuyos padres hayan sido modificados. El uso de un *caché* en la

que se almacenan las puntuaciones calculadas anteriormente evita rehacer cálculos innecesariamente y hacer más sencillo el algoritmo computacionalmente.

Una de las principales ventajas es que el modelo que devuelve un gráfico en el que todas las independencias de G están en P lo que hace al modelo más fácil su interpretación.

- Algoritmo *Tabu Search*

El algoritmo *Tabu Search* igual que el algoritmo *Hill Climbing* se basa en la definición de vecindario, procede iterativamente de una red Bayesiana o solución a otra red hasta que se maximice la función de puntuación. Cada red G tiene un vecindario asociado y cada red Bayesiana óptima se alcanza desde G mediante operaciones (eliminar, añadir o cambiar la dirección de un arco)

El defecto que tenía *Hill Climbing* es que este método permite alcanzar el óptimo local pero en la mayoría de los casos no será un óptimo global, no maximizará la función de puntuación. Para solucionar esto el algoritmo *Tabu Search* utiliza un método de vecindad dinámica, esto significa que el conjunto de vecinos puede cambiar según el historial de búsqueda. Las vecindades dinámicas pueden incluir la consideración serial o simultanea de varios tipos de operaciones o movimientos.

Además el algoritmo *Tabu Search* se basa en la memoria atributiva. Estas estructuras guardan la información sobre las propiedades de las soluciones previamente dadas. Los enfoques más comunes son; la memoria de recurrencia y la memoria basada en la frecuencia. La memoria basada en la recurrencia registra las operaciones que han cambiado durante el pasado reciente y la memoria basada en la frecuencia registra las relaciones sobre el número de iteraciones en las que una operación ha hecho cambiar o no a la solución de la red. El uso de la memoria cumple la función de evitar que los procesos de búsqueda se repitan, es decir, que se ajuste la misma secuencia de movimientos.

Por lo tanto el método *Tabu Search* sigue los mismo pasos 1,2 y 3 que el algoritmo *Hill Climbing* exceptuando el 4 ya que solo evaluará aquellas estructuras que no se han evaluado previamente o que hayan sufrido una mayor frecuencia de cambio, lo que evitará un óptimo local.

2.2.1.3 Aprendizaje híbrido

Los algoritmos de aprendizaje hibrido combinan las ideas de los algoritmos de aprendizaje basados en restricciones y los algoritmos de aprendizaje basados en puntuación, para compensar las debilidades de cada método.

- Algoritmo *Max-Min Hill Climbing*

Este algoritmo hibrido, en primer lugar reconstruye el esqueleto de una red antes de realizar una búsqueda codiciosa y restringida para orientar las aristas, sin tener en cuenta el orden inicial de los ejes. [20]

El algoritmo *Max-Min Hill-Clim*bing aprende el esqueleto sin tener en cuenta la orientación de los ejes utilizando un algoritmo de búsqueda local llamado *Max-Min Parents and Children* (MMPC) y orienta el esqueleto usando el algoritmo *Hill-Climbing* usando una puntuación Bayesiana. El aprendizaje se divide en dos fases, la fase de restricción y la fase de maximización.

Dada la estructura de la red G sobre cada nodo en V, no necesariamente vacía, se aplica cada uno de los algoritmos:

1. Se implementa la fase de restricción usando el algoritmo *Max-Min Parents and Children*. El algoritmo MMPC usa una técnica de selección para la detención de vecinos (padres e hijos) basada en la maximización de la medida de asociación mínima observada en cualquier conjunto de nodos seleccionados en las iteraciones previas. Es decir, selecciona un conjunto $C_i \subset V$ de los padres e hijos candidatos para cada nodo $X_i \in V$. El resultado de este algoritmo es utilizado para reconstruir el esqueleto de la Red Bayesiana antes de realizar una búsqueda codiciosa para orientar las aristas.

2. Se da lugar a la fase de maximización usando el algoritmo *Hill-Climbing*. El algoritmo HC encontrará la estructura de la red G que maximiza la puntuación entre las redes en las que los padres e hijos de cada nodo X_i se encuentran incluidos en el correspondiente conjunto C_i

La principal ventaja de este algoritmo es que es capaz de escalar distribuciones con miles de variables y mejorar el aprendizaje en términos de tiempo y calidad.

2.3 Aprendizaje de parámetros

Los parámetros de las distribuciones locales asociadas a cada variable pueden ser definidas por un sistema experto, o pueden ser estimadas o aprendidas dada una muestra de datos observables y conocida la estructura del grafo.

En las redes Bayesianas discretas, los parámetros a estimar son las probabilidades a priori de los nodos raíz y las probabilidades condicionales de las demás variables dados sus padres. Este aprendizaje es simple cuando todas las variables son completamente observables. Se asume que conociendo todas las variables es sencillo obtener las probabilidades, ya que las probabilidades a priori corresponden a las probabilidades marginales de los nodos raíz, matemáticamente:

Ecuación 10 Probabilidad a priori

$$P(A_i) \sim NA_i/N$$

donde NA_i es el número de ocurrencias del valor i de la variable A y N es el número total de casos.

Y las probabilidades condicionales se obtienen teniendo en cuenta los padres de cada nodo,

Ecuación 11 Probabilidad a posteriori

$$P(B_i|A_j, C_k) \sim \frac{NB_iA_jC_k}{NA_jC_k}$$

donde $NB_iA_jC_k$ es el número de casos siendo,

B= B_i, A= A_j y C= C_k y N A_jC_k

es el número siendo,

A= A_j y C =C_k.

La calidad de las estimaciones dependerá de si existe un número suficiente de datos en la muestra. Si no existe un número suficiente de datos, podemos representarlos mediante una distribución de probabilidad usando distribuciones a priori Beta para variables binarias y distribuciones *Dirichlet* para variables multinomiales, esto solo es útil si un conjunto de expertos concreta los valores de los parámetros de las distribuciones.

También es importante tener en cuenta si los datos están incompletos por faltas de valores o por nodos ocultos, es decir, faltan todos los valores de una variable.

Cuando existen valores faltantes existen varias alternativas, como eliminar los casos donde aparecen estos valores nulos, considerar un nuevo valor adicional para la variable, tomar el valor más probable usando el promedio de la variable, considerar la probabilidad de los diferentes valores en base a las otras variables o usar algoritmos para estimar su valor.

Cuando existen nodos ocultos, se puede usar el algoritmo EM (*Expected Maximization*). [22] Este algoritmo consiste en dos pasos:

- El paso E, donde se estiman los datos faltantes en base a los parámetros actuales. Iniciando los parámetros desconocidos con valores aleatorios o estimaciones de expertos.
- El paso M, se estiman los parámetros teniendo en cuenta los datos estimados previamente. Utiliza los datos estimados de los parámetros para estimar los valores de las variables ocultas, utiliza estos valores para completar la tabla de datos y re-estima estos parámetros con los nuevos datos hasta que no haya cambios significativos en los parámetros

Los pasos para un aprendizaje de parámetros en una red Bayesiana, teniendo en cuenta que no contamos con datos insuficientes, ni valores nulos o nodos ocultos son los siguientes:

En primer lugar, se identifica el espacio de parámetros a estimar. Consideramos una variable X con r observaciones independientes: $\{1,2,\dots,r\}$. Se tienen N observaciones de $X: D = \{X_1, \dots, X_n\}$, que es una muestra de tamaño N extraída de X.

Estamos interesados en estimar: P(X = k). Por lo tanto, el espacio paramétrico será :

Ecuación 12 Espacio paramétrico

$$\Theta = \left\{\theta = (\theta_1, \dots, \theta_r) | \theta_i \in [0,1], \sum_{i=1}^{r} \theta_i = 1\right\}$$

Dados esto parámetros estimaremos:

Ecuación 13 Definición de los parámetros a estimar

$$P(X = k \,|\theta_i, \dots, \theta_r) = \theta_k.$$

En segundo lugar, elegimos la técnica de estimación más adecuada para nuestro problema. Las dos técnicas mas comunes son; la frecuentista y la Bayesiana. A continuación se explicarán cada una de ellas.

Una vez elegido el método de estimación el proceso de estimación de parámetros en una red Bayesiana comienza con los nodos raíz, es decir, que no tienen predecesores y se estima su probabilidad marginal. Para aquellos nodos hoja, se estima la probabilidad condicional de la variable dados sus padres.

2.3.1 Estimación Frecuentista

Uno de los estimadores frecuentitas más usados es el estimador de máxima verosimilitud (MLE). El estimador de máxima verosimilitud se basa en las correspondientes frecuencias empíricas del conjunto de datos.

El algoritmo MLE pretende asignar un valor a los parámetros θ^{MLE} que maximiza la verosimilitud de los datos con el modelo. Es decir, la función de verosimilitud mide la probabilidad de obtener el conjunto de casos para un valor concreto del parámetro θ que tiene mayor probabilidad de ocurrir según lo observado.

La función logarítmica de verosimilitud se define como

Ecuación 14 Función logarítmica de verosimilitud

$$l(\theta : D) = \log P(D|\theta) = P(X = x_1, \dots, X = x_n|\theta).$$

Asumiendo que los casos son independientes, se cumplirá:

Ecuación 15 Distribución de probabilidad de máxima verosimilitud

$$P(D|\theta) = \prod_{i=1}^{N} P(X = x_i|\theta) = \prod_{k=1}^{r} \theta_k^{N_k}$$

Siendo N_k el número de casos en el conjunto para los cuales X=k

Escogemos el parámetro θ que maximice el valor de la función logarítmica de verosimilitud

$$\theta^* = (\theta_1^*, \theta_2^*, \dots, \theta_r^*) = \arg max_{(\theta_1, \theta_2, \dots, \theta_r)} l(\theta : D),$$

para una distribución categórica podemos simplificar la búsqueda de este parámetros calculando la frecuencia relativa, es decir, el estimador de máxima verosimilitud para P(X = k) que sería:

Ecuación 16 Estimador de máxima verosimilitud

$$\theta_k^* = \frac{N_k}{N}$$

2.3.2 Estimación Bayesiana

Se basa en la estadística Bayesiana ya que asume que los parámetros se modelan como una variable aleatoria $\Theta = (\theta_1, \theta_2, \dots \theta_r)$ esto genera una gran incertidumbre que será representada por media de las distribuciones de probabilidad.

La estimación Bayesiana tiene como objetivo estimar la probabilidad a posteriori de los parámetros considerando la función de probabilidad y distribución a priori de los datos. Las probabilidades posteriores estimadas se calculan a partir de una prioridad uniforme sobre cada tabla de probabilidad condicional.

Antes de que los datos sean observados, el parámetro desconocido sigue una distribución de probabilidad $f_\Theta(\theta)$ llamada distribución a priori. Esta distribución representa nuestra creencia previa sobre el valor de este parámetro.

Después de observar los datos, calculamos la distribución a posteriori de θ, que representa nuestro conocimiento sobre el parámetro θ después de observar los datos.

$$f_{\Theta|X}(\theta) \propto f_{x|\Theta}(x|\theta) f_\Theta(\theta)$$

Distribucción de densidad a posteriori $\propto$ función de verosimilitud x Distribucción de densidad a priori

Finalmente la estimación Bayesiana se representa a través de la media de la probabilidad a posteriori:

Ecuación 17 Estimación de parámetros Bayesiana

$$\theta_k^* = \frac{N_k + a_k}{N + \sum_{i=1}^{r} a_i}$$

Siendo N_k el número de casos en el conjunto de datos para los cuales X=k

Existen otras reglas para estimar como pueden ser la *regla Lindstone* que añade un parámetro lambda a la estimación de los parámetros, si el parámetro lambda es igual a 1 la regla que seguirá es la de *Laplace*.

Por lo tanto, la estimación frecuentista asume que los parámetros son constantes desconocidas y por el contrario la estimación Bayesiana asume que los parámetros desconocidos son variables aleatorias. Ambos algoritmos son buenas opciones para el aprendizaje de parámetros de la red Bayesiana, aunque en los experimentos empíricos se usará el enfoque frequentista ya que es más sencillo y mas correcto para trabajar con variables discretas.

2.4 Comparación y evaluación de Redes Bayesianas

Dado un grafo previamente llamado red Bayesiana original y las Redes Bayesianas aprendidas a través de los algoritmos de aprendizaje de parámetros y estructuras. Se comparan las redes aprendidas con la red original para encontrar aquel que mejor refleje la realidad.

La evaluación se divide en dos, en primer lugar, se comparan las estructuras de las redes Bayesianas y en segundo lugar, se comparan las distribuciones de probabilidades de las redes Bayesianas.

Se usará el paquete *bnlearn* [4] para la primera parte, usando las funciones compare() y *shd()*. Para la segunda parte, se usará el paquete *gRain* [5] para la marginalización de las distribucciones de probabilidad, y el paquete *seewaves* para el calculo de la divergencia de *Kullbak-Leibler*.

2.4.1 Comparación de estructuras de red

En este punto se evalúa si las redes aprendidas tienen la misma estructura que la red original, es decir, tiene los mismos arcos y las mismas independencias o relaciones directas.

Las métricas de evaluación que se usan son:

- El número de verdaderos positivos, falsos positivos y falsos arcos negativos incluidos en la red, comparadas con la original u objetivo.
- La distancia estructural de *Hamming* (SHD).
- La función de puntuación equilibrada (BSF).

El número de verdaderos arcos positivos (VP) corresponde con los arcos verdaderos presentes en el gráfico aprendido. El número de falsos arcos positivos (FP) corresponde con los arcos que no existen en el gráfico original, presentes en la red aprendida. Y el número de falsos arcos negativos (FN) corresponde con las falsas independencias directas representadas en la red aprendida.

La distancia estructural de Hamming (SHD) [20] es una métrica que calcula cuantos arcos difieren entre las CPDAGs de las dos redes con una penalización de ½ para aquellos arcos diferentes. De otra manera, se define como el número de operaciones necesarias para que los gráficos de probabilidad directa acíclica coincidan. Siendo operaciones añadir o eliminar un eje, eliminar o invertir la orientación de un eje, etc.

Ecuación 18 Distancia estructural de Hamming

$$SHD = FN + FP$$

La métrica SHD penaliza cada cambio requerido para transformar el grafo aprendido en el verdadero grafo. Sin embargo, es importante destacar que la puntuación SHD está sesgado hacia la sensibilidad de la identificación de aristas frente a la especificidad. La sensibilidad representa la proporción de dependencias directas aprendidas respecto a la red original y la especificidad la tasa de dependencias directas correctas aprendidas. Para resolver el problema del sesgo se propone la métrica BSF.

La función de puntuación equilibrada o BSF [23] [24]es una métrica que elimina el sesgo de la sensibilidad teniendo en cuenta todos los parámetros de la matriz de confusión lo que balanceará la puntuación entre las independencias y las dependencias directas.

Ecuación 19 Función de puntuación equilibrada

$$BSF = 0.5\left(\frac{VP}{\alpha} + \frac{VN}{i} - \frac{FP}{i} - \frac{FN}{\alpha}\right)$$

Donde α es el número de ejes, i es el número de independencias directas en el gráfico verdadero y $|V|$ es el tamaño del conjunto de variables V. Los valores de la métrica BSF van desde -1 hasta 1, donde -1 corresponde al peor gráfico posible, 1 al gráfico que más coincide con el verdadero, y 0 corresponde a un gráfico vacío.

Con las métricas anteriormente explicadas comparamos los grafos de manera estructural, pero dos redes Bayesianas muy distintas gráficamente pueden representar la misma distribución conjunta, debido a que la factorización no es única. Es por lo que en la siguiente sección explicaremos otras técnicas para comparar

las distribuciones de probabilidad conjunta entre redes y así conseguir la red Bayesiana que mejor se ajusta a la red Bayesiana original en todos sus componentes.

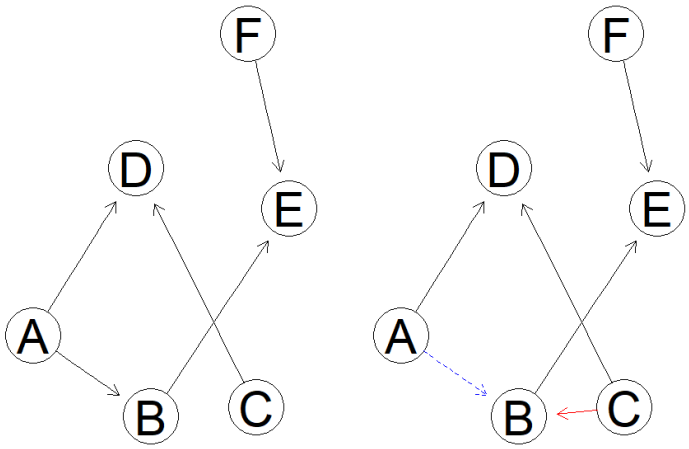

Figura 18 Ejemplo comparación estructural de Redes Bayesianas

La *figura 17* representa a la izquierda la red original y a la derecha la red aprendida. Los arcos en color negro representan los verdaderos arcos positivos, los rojos los falsos arcos positivos y los azules los falsos arcos negativos. Además su distancia estructural de Haming es de 2 y su función de puntuación equilibrada es igual a 0.5.

2.4.2 Comparación de distribuciones de probabilidad.

En esta parte se calcula la diferencia entre las distribuciones de probabilidad de las Redes Bayesianas.

En primer lugar, se calculan las distribuciones de probabilidad. Esto es una tarea muy compleja por lo que se calcularán las distribuciones marginales de todos los nodos y se agregarán usando la media.

En segundo lugar, compararemos ambas distribuciones usando métricas de la teoría de la información como la *Divergencia de Kullback-Leibler*.

La *Divergencia de Kullback-Leibler* [24] calcula una puntuación que mide la distancia relativa no simétrica entre dos distribuciones de probabilidad, es decir, calcula cuanto una distribución de probabilidad difiere de otra. El objetivo de la *divergencia de Kullback-Leibler* es medir la distancia entre dos distribuciones de probabilidad p(X) y q(X), teniendo una de ellas como referencia, definidas sobre la misma variable aleatoria X.

Ecuación 20 Divergencia de Kullback-Leible

$$D_{KL}(p||q) = \sum_{i=1}^{n} p(x_i) log \frac{p(x_i)}{q(x_i)}$$

siendo $KL(p||q) \geq 0$

La divergencia KL puede calcularse como la suma de la probabilidad de cada evento en P multiplicada por el logaritmo de la probabilidad del evento Q sobre la probabilidad del evento en P. El valor dentro de la suma es la divergencia para un evento determinado.

Cuando la probabilidad de un evento de P es grande, pero la probabilidad del mismo evento en q es pequeña, existe una gran divergencia. Cuando la probabilidad de P es pequeña y la de Q es grande, hay divergencia pero en menor medida. Cuando el valor de $KL(p||q) = 0$ las dos distribuciones de probabilidad serán iguales $p(x_i) = q(x_i); i = 1, \dots, n$. De esta manera se interpreta esta medida de información.

A continuación, se realizarán los experimentos para cada red de diferentes números de nodos y se compara su aprendizaje.

3 Experimentos

Para cada experimento se parte de una red Bayesiana previamente creada, que será la red original y servirá para comparar la estructura creada por los algoritmos de aprendizaje de estructuras a través de los datos [25]. Se realizan tres experimentos diferentes usando las mismas técnicas de aprendizaje tanto de parámetros como de estructuras. Las propiedades de los tres casos se detallan en la siguiente tabla

Tabla 1 Características de los datos usados en los experimentos empíricos

Nombre del conjunto de datos y redes originales	Nodos/Variables	Arcos
Deportes	9	15
Propiedades	27	31
Integración	88	138

Categorizaremos los experimentos en función del número de arcos y nodos. El primer experimento hace referencia a una red Bayesiana pequeña, el segundo a una red Bayesiana mediana y el tercero a una red Bayesiana grande.

El pre-procesado de los datos ha sido bastante sencillo ya que nuestro conjunto de datos no presentaba valores nulos o atípicos, además las variables ya estaban discretizadas y por lo tanto todas ellas serán categóricas

3.1 Red Bayesiana pequeña

En este primer experimento, la red Bayesiana original es categorizada como una Red Bayesiana pequeña ya que cuenta con 9 nodos y 27 arcos.

El conjunto de datos [25] que contiene 10.000 observaciones y 9 variables, igual que el número de nodos de nuestra red. Representan clasificaciones de equipos de futbol a través de varias estadísticas de rendimiento dependiendo de el lugar de juego, en su estadio o fuera de su estadio de pertenencia. Los datos entrenaran nuestros algoritmos de aprendizaje de parámetros y estructuras.

Las variables categóricas que están representadas las podemos agrupar en tres grupos:

- Variables que contienen información sobre el equipo como la posesión o el nivel y el resultado del partido
- Variables que contienen información técnica sobre los partidos jugados fuera de casa, como los tiros a puerta y los goles.
- Variables que contienen información técnica sobre los partidos jugados en su propio estado, como los tiros a puerta y los goles.

A continuación, pasamos a analizar los aspectos más importantes de las representaciones de las redes Bayesianas, que nos dará más conocimiento a cerca de la realidad de los datos.

Esta es la red original de la que partimos, ilustrada en orden ancestral:

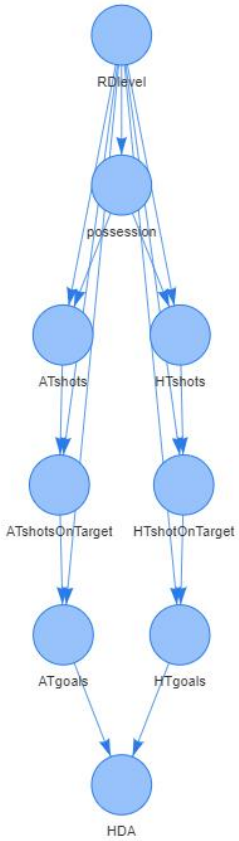

Figura 19 Red Bayesiana original pequeña

En la *figura 18* esta red Bayesiana cuenta con 9 nodos y 15 arcos directos. El nodo raíz es *RDlevel* y el nodo hoja *HDA*.

Contiene varios arcos reversibles, que son aquellos que cambiar su dirección no provocaría ciclos. En realidad, todos los arcos son reversibles, dependen del orden de inversión el que alguno no se pueda invertir. El orden de inversión implica un factorial de posibles ordenes y complejidades en el cálculo de las probabilidades, ya que la inversión de un arco implica la herencia cruzada de padres entre las variables, lo que podría implicar tablas muy grandes.

También cuenta con varios arcos completos que son identificables de forma única porque si cambia la dirección introduciría ciclos y forman parte de una V-estructura. Estos arcos podemos identificarlos en las siguientes imágenes como los formados por guiones.

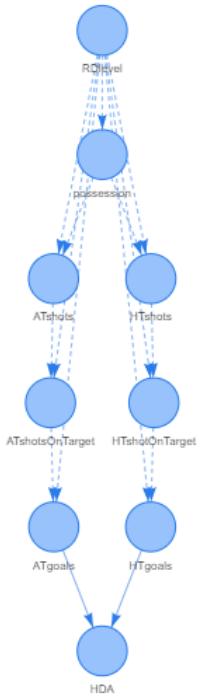

Figura 20 Arcos Reversibles Red Bayesiana Pequeña

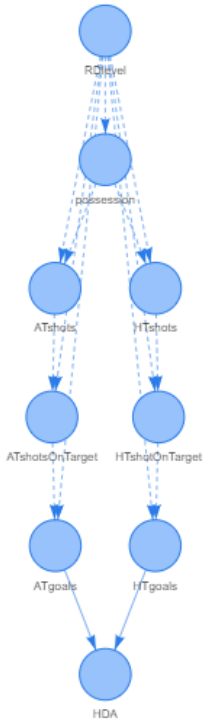

Figura 21 Arcos Completos Red Bayesiana Pequeña

En la *Tabla 2* vemos un resumen de los vecinos padres e hijos. Por ejemplo, para el nodo HTshots su nodo padre es *posession* y *RDlevel* y su nodo hijo es tiros a puerta fuera de cada *HTshotsOnTarget*, siendo ambos sus vecinos. Lo que significa que los disparos en un partido fuera de casa son condicionalmente dependientes de la posesión del balón durante el partido y el nivel que se le establece al equipo.

Tabla 2 Vecinos, padres e hijos de la Red Bayesiana Original Pequeña

Nodos	Neighborhood	Parents	Childrens
RDlevel	*Possession,ATshots,ATshotsOnTarget,ATgoals,HTshots, HTshotOnTarget, HTgoals*		*Possession,ATshots,ATshotsOnTarget,ATgoals,HTshots, HTshotOnTarget, HTgoals*
possession	*HTshots, ATshots, RDlevel*	*RDlevel*	*HTshots, ATshots*
HTshots	*HTshotOnTarget, possession,RDlevel*	*Possession, RDlevel*	*HTshotOnTarget*
ATshots	*ATshotOnTarget, possession, RDlevel*	*Possession, RDlevel*	*ATshotOnTarget*
HTshotOnTarget	*HTshots, HTgoals, RDlevel*	*HTshots, RDlevel*	*HTgoals*
ATshotOnTarget	*ATshots, ATgoals, RDlevel*	*ATshots, RDlevel*	*ATgoals*
ATgoals	*ATshotOnTarget, HDA, RDlevel*	*ATshotOnTarget, RDlevel*	*HDA*

HTgoals	*HTshotOnTarget, HDA, RDlevel*	*HTshotOnTarget, RDlevel*	*HDA*
HDA	*HTgoals, ATgoals*	*HTgoals, ATgoals*	

Usando la separación-d estudiamos si los nodos son independientes o no entre ellos. Por ejemplo, la variable HTshots y *HTgoals* están d-separados, porque el nodo observado *HTshotsOnTarget* está bloqueando todos los posibles caminos entre ambos nodos. Por lo tanto, los tiros fuera de casa son condicionalmente independientes de los goles marcados fuera de casa dados los tiros a portería fuera de casa.

Estudiaremos el conjunto de nodos que hacen independiente a un nodo del resto de la red, esto es lo que se llama *Manto de Markov*. El *manto de Markov* es el conjunto mínimo de nodos que *d*-separa al nodo objetivo de todos los demás nodos, nuestra red tiene tamaño medio de la *manta de Markov* de 2.25, esto significa que aproximadamente dos variables son las que d-separan el nodo objetivo. En la *figura 21* vemos un ejemplo; las variables que hacen independiente al nodo *possession* de los demás.

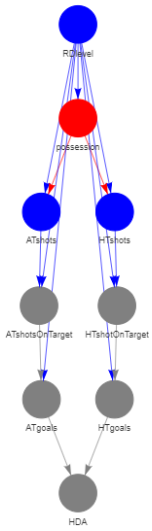

Figura 22 Ejemplo de Manta de Markov para la Red Bayesiana Pequeña

Los nodos *HTShots*, *ATShots* y *RDlevel* hacen independientes al nodo *posession* del resto de variables.

También se observa que tiene una V-estructura y es única:

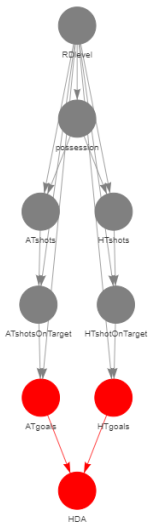

Figura 23 Estructura V en la Red Bayesiana pequeña

Esta estructura nos explica que ATgoals ⊥ HTgoals | HDA si *HDA* no es observada, pero Tgoals⊥⊥ HTgoals | HDA si *HDA* es observada. En otras palabras, la cantidad de goles marcados en casa es independiente de la cantidad de goles marcados fuera conociendo el resultado del partido, pero si no conocemos el resultado non son independientes.

Por último, representamos el gráfico que representa una clase equivalente al DAG original, o lo que también llamamos CPDAG. Como hemos visto en la literatura esto significa que podemos agrupar DAGs en clases de equivalencia probabilística siempre que tengan el mismo CPDAG.

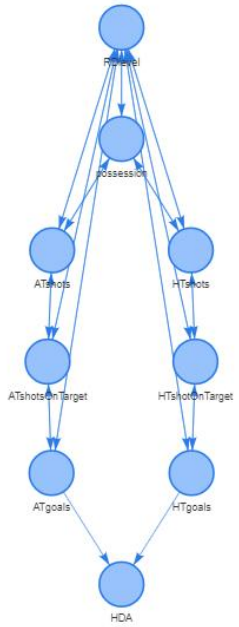

Figura 24 CPDAG de la Red Bayesiana pequeña

El número de parámetros de la red Bayesiana es de 1049, este cálculo se basa en el grado de las variables y el número de nodos. El número de parámetros para cada nodo es el siguiente:

Tabla 3 Parámetros red Bayesiana pequeña

Nodos	Número de parámetros
ATgoals	160
ATshots	160
ATShotsOnTarget	160
HTgoals	160
HTshots	160
HTShotsOnTarget	160
Possession	32
RDLevel	7
HDA	50

Una vez aprendidos los parámetros, aplicamos los algoritmo de aprendizaje de estructuras. Comenzamos con los algoritmos basados en pruebas de independencia condicional. Los algoritmos que aplicamos son el *PC* y *Grow Shrink* con los tests de independencia información mutua y Pearson X^2. Y estas son las estructuras aprendidas por cada una de ellas (*figuras 34 y 25*):

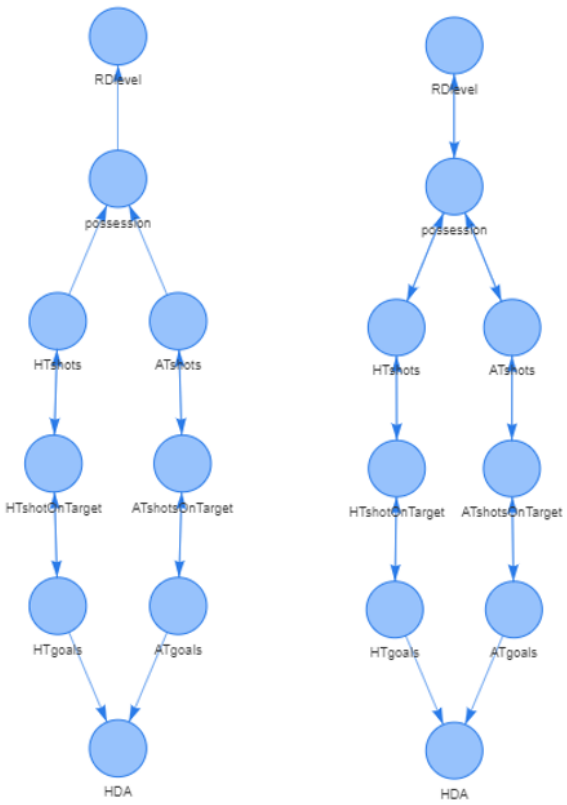

Figura 25 Estructuras aprendidas por el algoritmo PC con pruebas de independencia Pearsons's x2 e Información Mutua (de izq a drch)

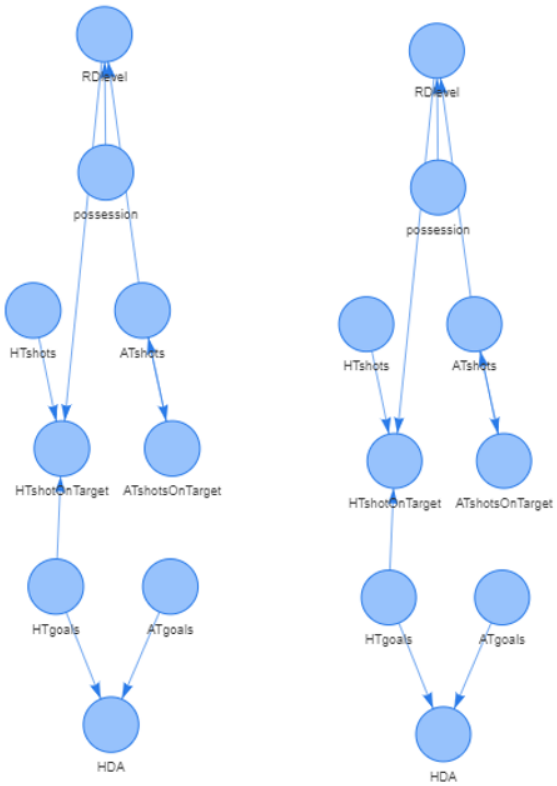

Figura 26 Estructuras aprendidas por el algoritmo GS con pruebas de independencia Pearsons's x2 e Información Mutua (de izq a drch)

Seguimos con los algoritmos basados en puntuación que son *Hill-Climbing* y *Tabu Search* usando como medidas de puntuación BIC, AIC, BDeu (*figuras 26 y 27*):

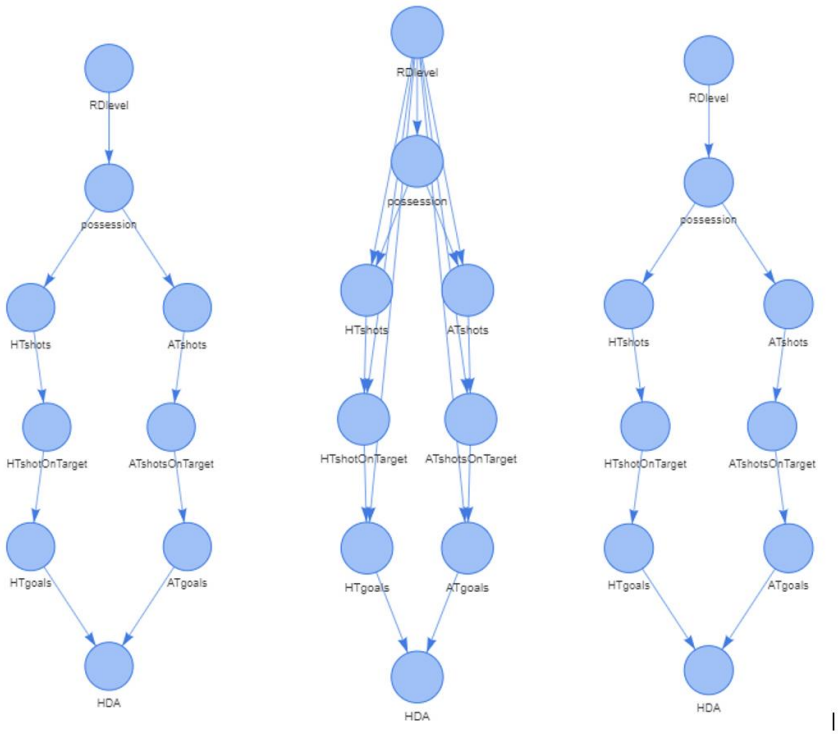

Figura 27 Estructuras aprendidas por el algoritmo HC con las puntuaciones BIC, AIC y BDeu (de izq a drch)

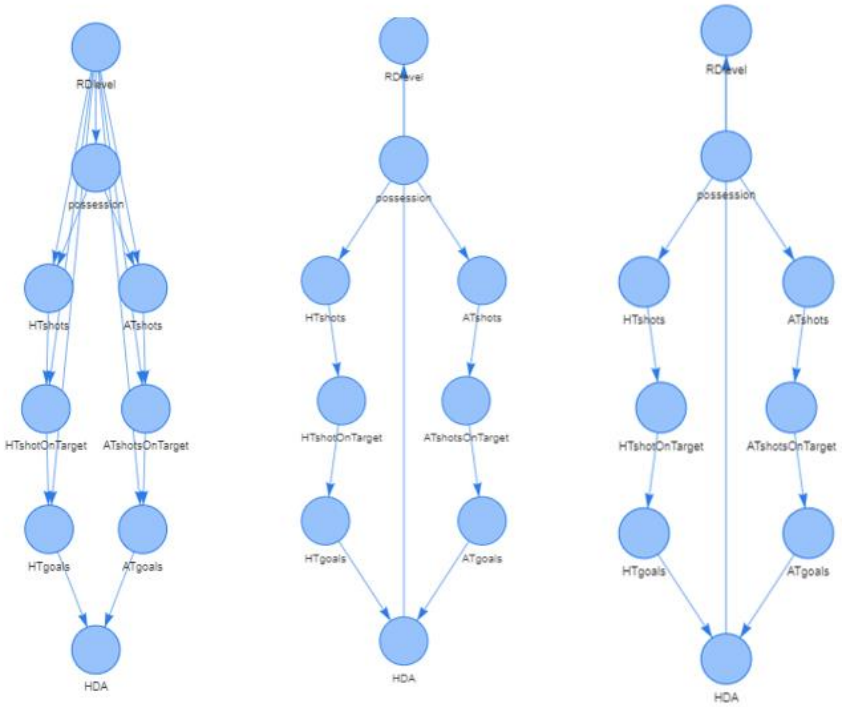

Figura 28 Estructuras aprendidas por el algoritmo TS con las puntuaciones BIC, AIC y BDeu (de izq a drch)

Se calcula la puntuación de cada algoritmo en la siguiente tabla:

Tabla 4 Puntuación algoritmos de aprendizaje de puntuaciones para la red Bayesiana pequeña

Puntuación		
	Hill-Climbing	*Tabu Search*
BIC	-109376.9	-109276.7
AIC	-111130-5	-111130.5
BDeu	-109376.9	109276.7

La puntuación máxima para el algoritmo *Hill-Climbing* y *Tabu Search* es AIC, por lo tanto es la puntuación que mejor ayudará al algoritmo a capturar la estructura de red correcta.

Por último implementaremos el algoritmos hibrido *Max-Min Hill Climbing,* que aprende la siguiente estructura (*figura 28*):

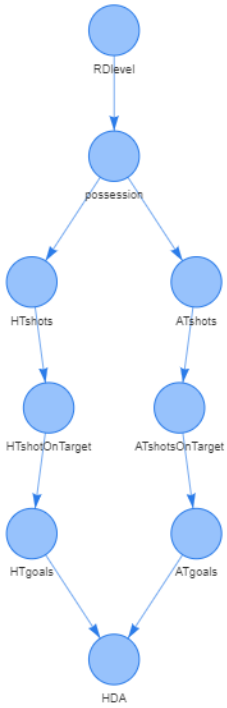

Figura 29 Estructura aprendida por el algoritmo MMHC

Todos los grados aprendidos han captado las relaciones con mayor significatividad que anteriormente calculamos a través de la fuerza de los arcos.

Los grafos calculados por los algoritmos de pruebas de independencia condicional crean arcos directos, por lo que son grafos parcialmente directos y no es posible aprender sus parámetros y por ende su distribución de probabilidad conjunta.Los grafos calculados con los algoritmos de puntuación y el algoritmo híbrido crean una estructura completamente indirecta.

A continuación, se aprenderán los parámetros de todos algoritmos usando el algoritmo de aprendizaje de parámetros frecuentitas, estimación de máxima verosimilitud y obtendremos las distribuciones de probabilidad condicional de cada nodo. Aquí vemos el aprendizaje de parámetros de la red Bayesiana original.

Los métodos basados en aprendizaje de pruebas de condición independiente incluyen ciclos en la red y por lo tanto se ha elegido de manera aleatoria una dirección del arco para poder proceder con el aprendizaje de parámetros.

En la *figura 29* observamos el valor de cada estado del nodo, lo que compone la distribución de probabilidad de la red. Lo que es equivalente a:

P (RDlevel, possession, HTgoals, ATgoals, HTshotsOnTarget, ATshotsOnTarget, HTshots, ATshots, HDA) =

P(RDlevel)P(posession | RDlevel) P(HTshots| possession, RDlevel)P(ATshots| possession, RDlevel)P(HTshotsOnTarget | HTshots, RDlevel) P(ATshotsOnTarget| ATshots, RDlevel) P(HTgoals| HTshotsOnTarget, RDlevel) P(ATgoals| ATshotsOnTarget, RDlevel) P(HDA| HTgoals, ATgoals)

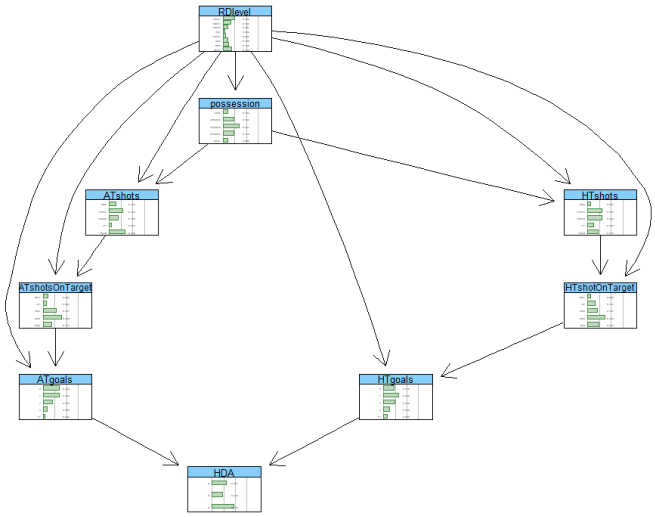

Figura 30 Probabilidades de cada nodo de la Red Bayesiana pequeña

En la *figura 30* podemos ver las distribuciones de probabilidad de los nodos *RDlevel*, *ATshots* y *HDA*.

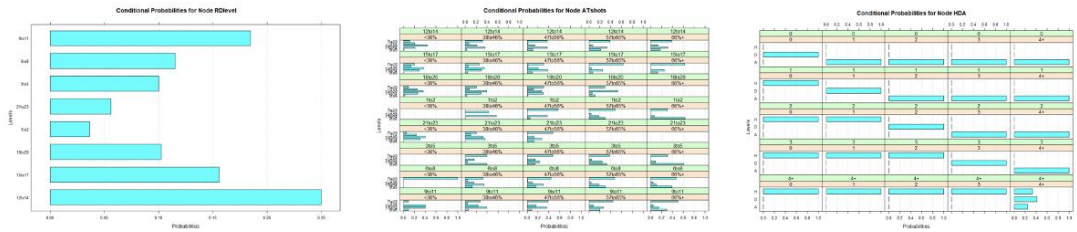

Ilustración 31 Ejemplo distribuciones de probabilidades marginales red Bayesiana pequeña

La única variable cuya distribución de probabilidad es igual a su distribución de probabilidad marginal es *RDlevel* ya que no cuenta con ningún nodo padre. Las variables *ATshots* y HDA están condicionadas a sus nodos padres, *ATshots* a *possession* y *HDA* a *HTgoals* y *ATgoals*.

3.2 Red Bayesiana mediana

En este segundo experimento, la red Bayesiana original es categorizada como una Red Bayesiana mediana ya que cuenta con 27 nodos y 31 arcos.

El conjunto de datos [25] que contiene 10.000 observaciones y 27 variables, igual que el número de nodos de nuestra red. Representan variables de evaluación de decisiones de inversión en el mercado inmobiliario del Reino Unido.

Las variables categóricas que están representadas las podemos agrupar en tres grupos:

- Variables que recogen información del porcentaje y el valor de los gastos de propiedad, inmobiliarios, tasas de intereses, etc.
- Variables que recogen información tanto bruto como el porcentaje del alquiler de propiedades.
- Beneficios y ganancias brutos o en porcentaje sobre el alquiler, el capital, etc.
- Variables que recogen el valor de compra de la propiedad, de los impuestos, etc.

Esta es la red original de la que partimos ilustrada en orden ancestral:

Figura 32 Red Bayesiana Original Mediana

Como se puede observar en la *figura 31*, esta red Bayesiana cuenta con 27 nodos y 31 arcos todos ellos directos. Los nodos raíz son: *PropManagement, otherPropExpens, rentalIncomeLoss%, rentalIncome, PropPurchValue, PropExpensGrowth, rentalGrowth, capitalGrowth, incomeTax, interestRate, borrowing y otherInrerestFees* representandos en la *figura 26*. Los nodos hoja son: *rentalGrossYield, rentalIncomeT+1, LTV, PropValueT+1, stampDutyTax, otherPropExpensT+1 y netProfit* representados en las *figura 27*.

Figura 34 Nodos raíz Red Bayesiana Mediana

Contiene tres arcos reversibles representados en la *tabla 5*, que son aquellos que cambiar su dirección no provocaría ciclos. Y 28 arcos completos representados en la t*abla 6*, que son identificables de forma única porque si cambia la dirección introduciría ciclos y forman parte de una estructura v

Tabla 5 Arcos Reversibles Red Bayesiana Mediana

Arcos reversibles	
Desde	Hasta
PropPurchValue	*stampDutyTaxBand*
PropPurchValue	*stampDutyTax*
stampDutyTaxBand	*stampDutyTax*

Tabla 6 Arcos Completos Red Bayesiana Mediana

Arcos completos	
Desde	Hasta
PropManagement	*PropExpenses*
PropExpenses	*rentalGrossProfit*
rentalGrossProfit	*rentalNetProfitBefInt*
actualRentalIncome	*rentalGrossProfit*
actualRentalIncome	*PropExpenses*
actualRentalIncome	*rentalGrossProfit*
actualRentalIncome	*rentalGrossYield*
otherPropExpens	*rentalIncomeT+1*
otherPropExpens	*PropExpenses*
rentalNetProfitBefInt	*otherPropExpensT+1*
rentalIncomeLossp	*actualRentalIncome*
rentalIncome	*actualRentalIncome*
PropPurchValue	*LTV*
PropPurchValue	*PropValueT+1*
PropPurchValue	*capitalGains*
PropPurchValue	*rentalNetProfitBefInt*

capitalGains	*PropValueT+1*
PropExpensGrowth	*otherPropExpensT+1*
rentalGrowth	*rentalIncomeT+1*
capitalGrowth	*capitalGains*
incomeTax	*rentalNetProfitBefInt*
intTaxRelief	*netProfit*
Interest	*intTaxRelief*
interest	*netProfit*
interestRate	*interest*
borrowing	*LTV*
borrowing	*Interest*
otherInterestFees	*interest*

En la *tabla 7* vemos un resumen de los vecinos padres e hijos de cada uno de los nodos. Por ejemplo el nodo *intTaxRelief* , su nodo padre es *interest* y su nodo hijo es *netProfit*, siendo ambos sus vecinos. Lo que significa que el porcentaje de ingresos sobre la renta es condicionalmente dependiente dado los intereses ,y que el beneficio neto anual es condicionalmente dependiente del porcentaje de ingresos sobre la renta.

Tabla 7 Vecinos, Padres e Hijos de la Red Bayesiana Mediana

Nodos	Neighborhood	Parents	Childrens
rentalIncomeLossp	*actualRentalIncome*		*actualRentalIncome*
rentalGrowth	*rentalIncomeT+1*		*rentalIncomeT+1*
rentalIncomeT+1	*actualRentalIncome, rentalGrowth*	*actualRentalIncome, rentalGrowth*	
LTV	*PropPurchValue, borrowing*	*PropPurchValue, borrowing*	
intTaxRelief	*netProfit, interest*	*interest*	*netProfit*
netProft	*rentalNetProfitBefInt, intTaxRelief, interest*	*rentalNetProfitBefInt, intTaxRelief, interest*	
…	…	…	…

Usando la d-separación estudiamos si los nodos son independientes o no entre ellos. Por ejemplo, los nodos *rentalIncome* y *propExpenses* están d-separados, porque el nodo observado *actualRentalIncome* está bloqueando todos los posibles caminos entre ambos nodos. Por lo tanto, los ingresos por alquiler son condicionalmente dependientes de los gastos inmobiliarios totales dado los ingresos reales por alquiler.

Estudiaremos el conjunto de nodos que hacen independiente a cada nodo del resto de la red, esto es lo que se llama *Manto de Markov*. El *manto de Markov* es el conjunto mínimo de nodos que d-separa al nodo objetivo de todos los demás nodos, nuestra red tiene tamaño medio de la *manta de Markov* de 3.41, esto significa que aproximadamente dos variables son las que d-separan el nodo objetivo. En la *figura 34* vemos un ejemplo de la *manta de Markov* del nodo *capitalGains* :

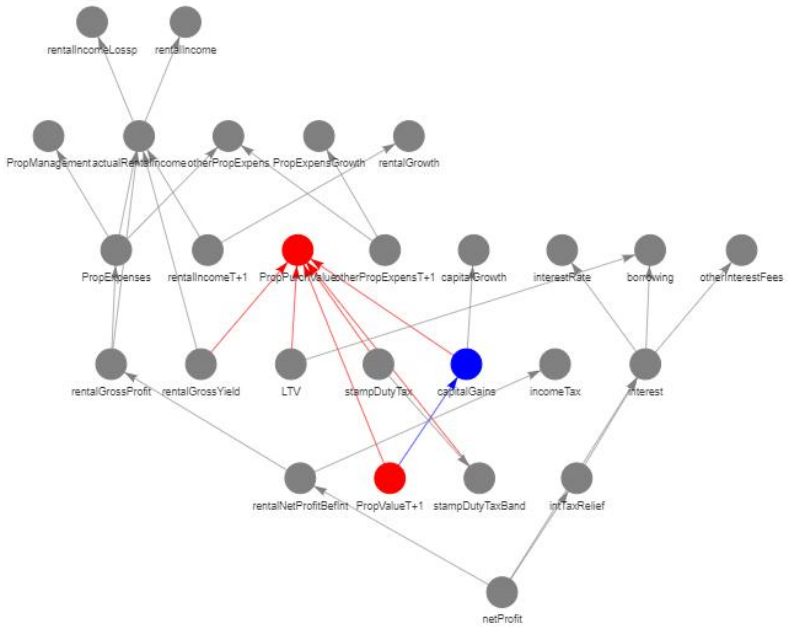

Figura 35 Ejemplo Manto de Markov para Red Bayesiana Mediana

Los nodos *PropPurchaseValue* y *PropValueT+1* hacen independiente al nodo capitalGains del resto de variables.

Existen 5 V-estructuras de la red y en la *figura 35* vemos un ejemplo de dos de ellas:

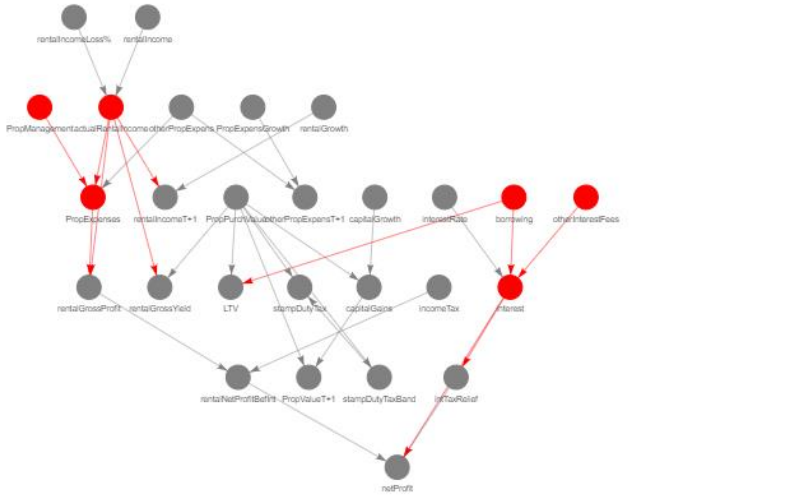

Figura 36 Ejemplo Estructuras en V de la Red Bayesiana Median

Esta estructura nos explica que borrowing ⊥ otherInterestFees | interest si *interest* no es observada, pero borrowing ⊥̸ otherInterestFees | interest si *interest* es observada. En otras palabras, la cantidad de prestamos pedidos es independiente de otras comisiones de intereses conociendo el valor totales de los intereses.

Por último, visualizamos la *figura 36* que representa una clase equivalente al DAG original, o lo que también llamamos CPDAG.

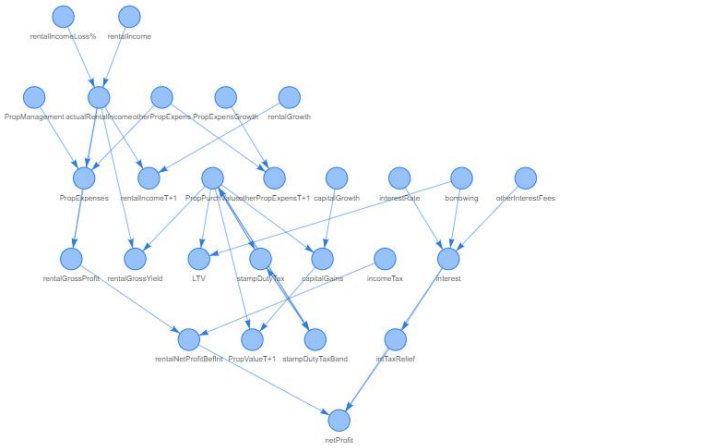

Figura 37 CPDAG Red Bayesiana Mediana

El número de parámetros de Red Bayesiana es de 3056, este cálculo se basa en el grado de las variables y el número de nodos. El número de parámetros para cada nodo es el siguiente:

Tabla 8 Parámetros de la red Bayesiana mediana

Nodos	Parámetros	Nodos	Parámetros
PropManagemet	3	rentalGrowth	4
PropExpenses	216	capitalGrowth	4
rentalGrossProfit	120	incomeTax	3
actualRentalIncome	150	intTaxRelief	30
otherPropExpens	2	netProfit	1296
rentalNetProfitBefInt	120	interest	360
rentalGrossYield	144	interestRate	5
rentalIncomeT+1	150	borrowing	3
rentalIncomeLossp	4	otherInterestFees	2
rentalIncome	5	PropExpensGrowth	4
PropPurchValue	5	otherPropExpensT+1	30
LTV	96	stampDutyTax	18
PropValueT+1	150	capitalGains	120
stampDutyTaxBand	12		

Una vez conocidos los parámetros, aplicamos los algoritmo de aprendizaje de estructuras. Comenzamos con los algoritmos basados en pruebas de independencia condicional. Los algoritmos que aplicamos son el *PC* y *Grow Shrink* con los tests de independencia información mutua y *Pearson* X^2. Y estas son las estructuras aprendidas por cada algoritmo (*figuras 37 y 38*):

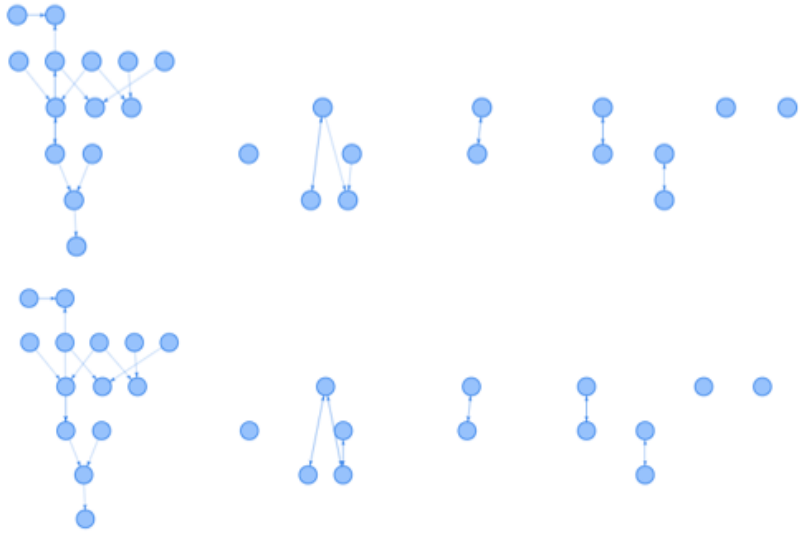

Figura 38 Estructuras aprendidas por el algoritmo PC con pruebas de independencia Pearsons's x2 e Información Mutua (de arriba a abajo)

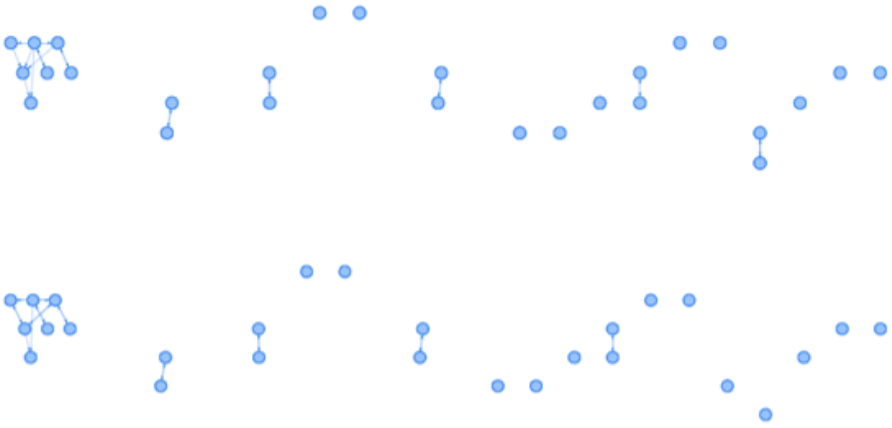

Figura 39 Estructuras aprendidas por el algoritmo GS con pruebas de independencia Pearsons's x2 e Información Mutua (de arriba a abajo a drch)

Seguimos con los algoritmos basados en puntuación que son *Hill-Climbing* y *Tabu Search* usando como medidas de puntuación BIC, AIC, BDeu (*figuras 39 y 40*)

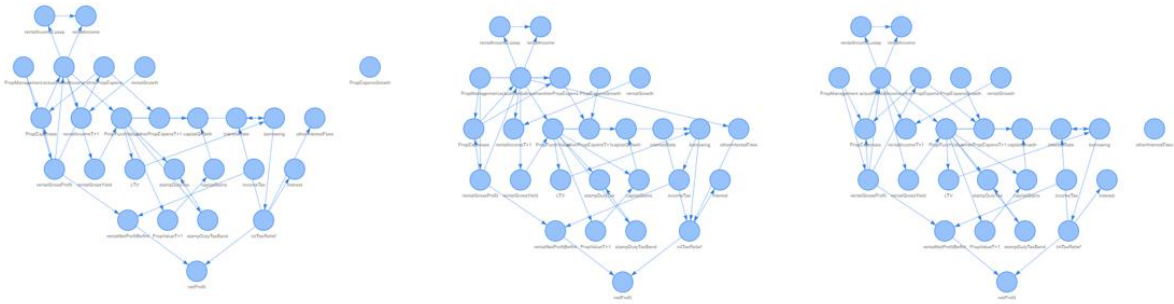

Figura 40 Estructuras aprendidas por el algoritmo HC con las puntuaciones BIC, AIC y BDeu (de izq a drch)

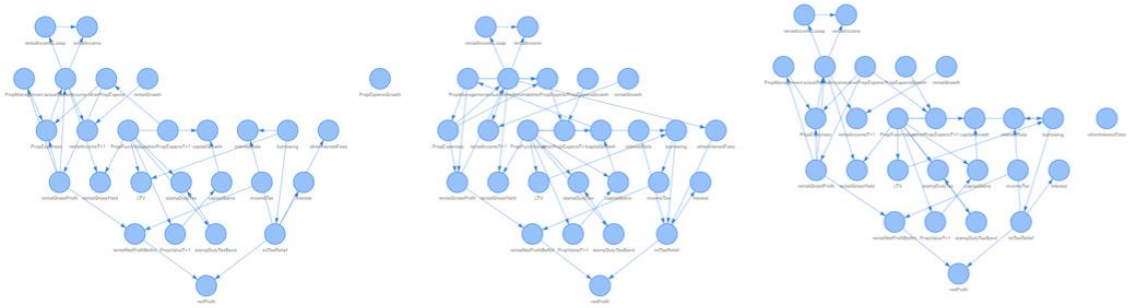

Figura 41 Estructuras aprendidas por el algoritmo TABU con las puntuaciones BIC, AIC y BDeu (de izq a drch)

Se calcula la puntuación de cada algoritmo en la *tabla 9*:

Tabla 9 Puntuaciones algoritmos de aprendizaje red Bayesiana mediana

Puntuación		
	Hill-Climbing	Tabu Search
BIC	-241292.6	-241140
AIC	-242175.8	-242117.5
BDeu	-241583.4	-241484.8

La puntuación máxima para el algoritmo *Hill-Climbing* y *Tabu Search* es AIC, por lo tanto es la puntuación que mejor ayudará al algoritmo a capturar la estructura de red correcta.

Por último implementaremos el algoritmos hibrido *Max-Min Hill Climbing,* que aprende la siguiente estructura (*figura 41*):

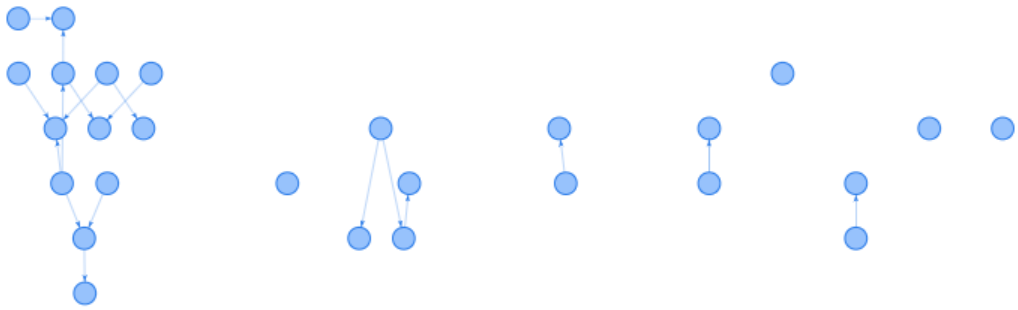

Figura 42 Estructuras aprendidas por el algoritmo MMHC

Solo los grafos aprendidos con algoritmos de puntuación han captado las relaciones con mayor significatividad que anteriormente calculamos a través de la fuerza de los arcos.

A continuación, se aprenderán los parámetros de la red original y de las redes aprendidas por todos los algoritmos usando el método frecuentita *máximum likelihood* y obtendremos las distribuciones de probabilidad condicional de cada nodo. La distribución de probabilidades de la red original es la siguiente:

P (rentalIncomeLoss%, rentalIncome, PropManagement, actualRentalIncome, otherPropExperts, PropExpensGrowth, rentalGrowth, PropExpenses, rentalncomeT+1, PropPurchValue, otherPropExpensT+1, capitalGrowth, interestRate, borrowing, otherInterstFees, rentalGrowthProfit, rentalGrossYield, LTV, stampDuty, capitalGains, incomeTax,Interest, rentalNetProfitBenfit, PropValueT+1, stampDutyTaxBand, intTaxRelief, netProfit) =

P(rentalIncomeLoss%)P(rentalIncome)P(PropManagement)P(actualRentalIncome|rentalIncomeLoss%,rentalIncome)P(otherPropExperts)P(PropExpensGrowth)P(rentalGrowth)P(PropExpenses| PropManagement, actualRentalIncome, otherPropExperts)P(rentalncomeT+|rentalGrowth,actualRentalIncome)P(PropPurchValue)P(otherPropExpensT+1|otherPropExperts,PropExpensGrowth)P(capitalGrowth)P(interestRate)P(borrowing)P(otherInterstFees)P(rentalGrossProfit| PropExpenses)P(rentalGrossYield| actualRentalIncome, PropPurchValue)P(LTV| PropPurchValue, borrowing)P(stampDutyTax| PropPurchValue, stampDutyTaxBand)P(capitalGains| capitalGrowth, PropPurchValue)

P(incomeTax)P(Interest| interestRate, borrowing, otherInterstFees) P(rentalNetProfitBenfit| incomeTax, rentalGrossProfit)P(PropValueT+1| capitalGains, PropPurchValue)P(stampDutyTaxBand| stampDutyTax)P(intTaxRelief| Interest)P(netProfit| rentalNetProfitBenfit, intTaxRelief)

Los métodos basados en aprendizaje de pruebas de condición independiente incluyen ciclos en la red y por lo tanto se ha elegido de manera aleatoria una dirección del arco para poder proceder con el aprendizaje de parámetros.

3.3 Red Bayesiana grande

En el último experimento, la red Bayesiana original es categorizada como una Red Bayesiana grande ya que cuenta con 88 nodos y 138 arcos.

El conjunto de datos [26] que contiene 10.000 observaciones y 88 variables, igual que el número de nodos de nuestra red. Contiene información acerca del riesgo de reincidencia de presos con enfermedades mentales, medido a través de múltiples intervenciones de gestión del riesgo de recaída.

Las variables categóricas que están representadas se pueden agrupar en siete grupos:

- Características de la persona a evaluación como su genero, edad o inteligencia.
- Características psicológicas como ansiedad, estrés, victimización, violencia, síntomas de enfermedad mental, etc.
- Test psicológicos realizados como el PCLR.
- Vida pasada, si ha tenido dificultades económicas, familia criminal o ha sufrido abusos en su infancia.
- Comportamiento con las drogas.
- Características de su vida en prisión.

A continuación, pasamos a analizar los aspectos más importantes de las representaciones de las redes Bayesianas, que nos dará más conocimiento a cerca de la realidad de los datos.

Esta es la red original de la que partimos ilustrada en orden ancestral:

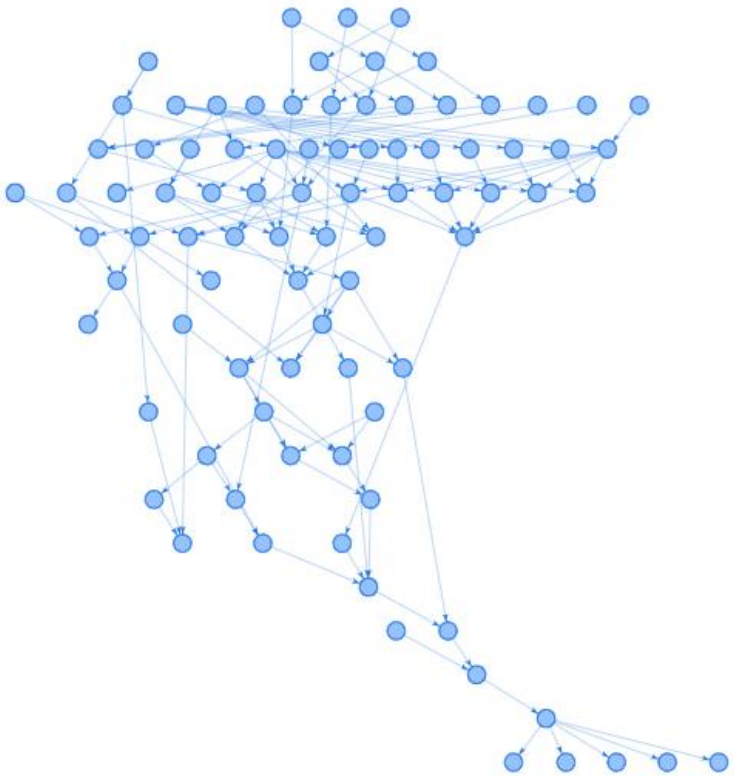

Figura 43 Red Bayesiana original grande

Esta red Bayesiana cuenta con 88 nodos y 138 arcos directos. Los nodos raíz y nodos hoja los podemos identificar en las *figuras 43 y 44*.

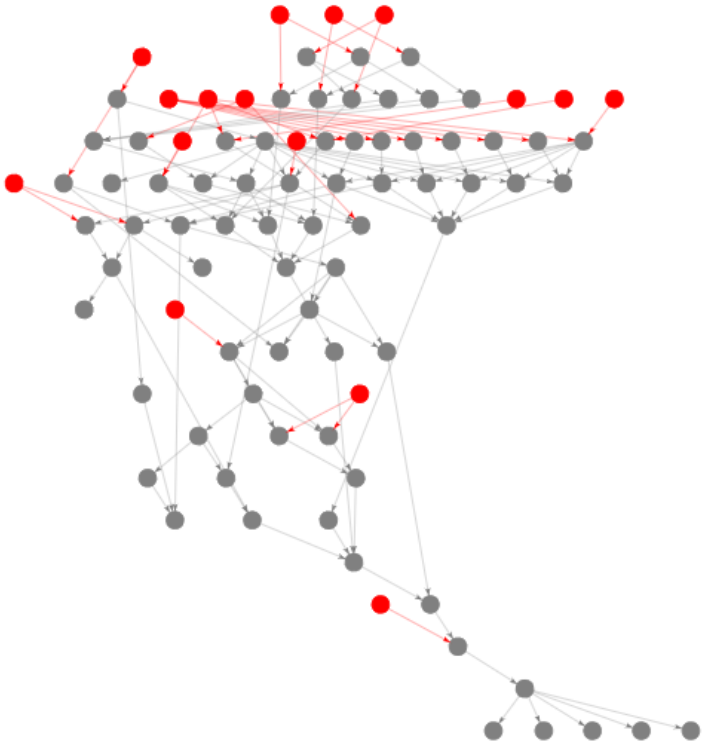

Figura 44 Nodos raíz red Bayesiana grande

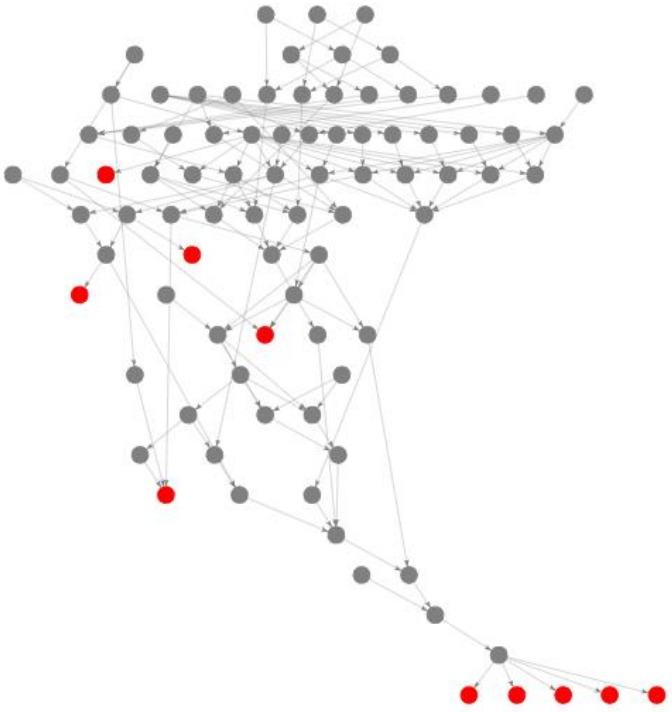

Figura 45 Nodos hijos red Bayesiana grande

Contiene 24 arcos reversibles y 114 arcos completos . Estos arcos podemos identificarlos en *la tabla 10.*

Tabla 10 Arcos reversibles red Bayesiana grande

Arcos reversibles	
Intelligence	AbilityToCope
Education	EmploymentOrTraining
Education	Intelligence
Intelligene	ResponsivenessToTreatment
SymptompsOfMentalIllnes	ParanoidDelusions
SymptompsOfMentalIllnes	StrangeExperiences
...	...

Tabla 11 Arcos completos red Bayesiana grande

Arcos completos	
AbilityToCope	Stress
ProblematicLifeEvents	Stress
Victimisation	ProblematicLifeEvents
ViolentThoughts	AggressionDL
Impulsivity	PCLRfacet3
Impulsivity	AggressionDL
...	...

Estudiaremos el conjunto de nodos que hacen independiente a un nodo del resto de la red, esto es lo que se llama *Manto de Markov*. El *manto de Markov* es el conjunto mínimo de nodos que *d*-separa al nodo objetivo de todos los demás nodos, nuestra red tiene tamaño medio de la *manta de Markov* de 4.98 , esto significa que aproximadamente dos variables son las que d-separan el nodo objetivo. En *figura 45* vemos un ejemplo

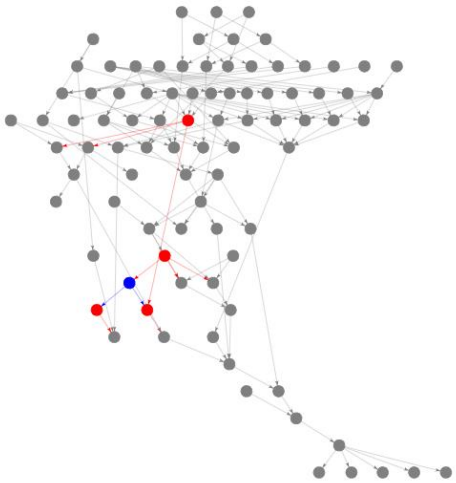

Figura 46 Ejemplo Manto de Markov red Bayesiana grande

Representamos representa el gráfico que representa una clase equivalente al DAG original, o lo que también llamamos CPDAG. Como hemos visto en la literatura esto significa que podemos agrupar DAGs en clases de equivalencia probabilística siempre que tengan el mismo CPDAG.

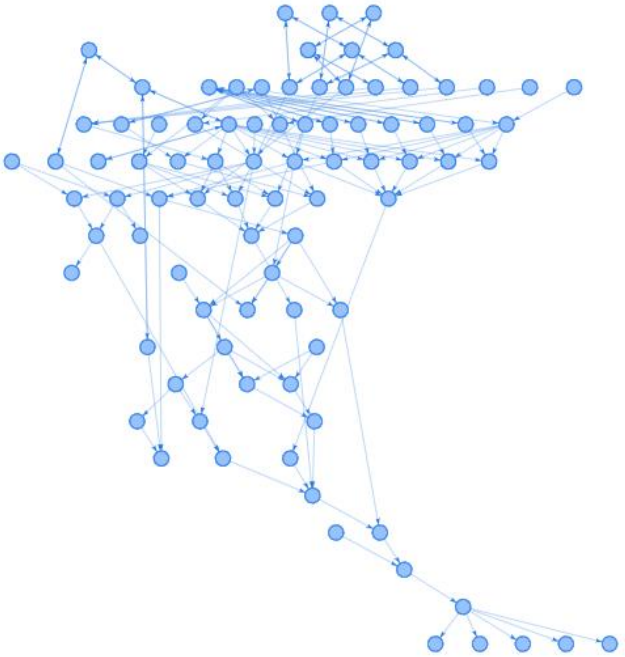

Figura 47 CPDAG red Bayesiana grande

A continuación, pasamos a el aprendizaje de las redes basándonos en la muestra de datos dada.

El número de parámetros de la red Bayesiana es de 912, este cálculo se basa en el grado de las variables y el número de nodos. El número de parámetros para cada nodo es el siguiente:

Tabla 12 Parámetros de la red Bayesiana grande

Nodo	Parametros
PCLRfacet3	12
AbilityToCope	6
Stress	8
ProblematicLifeEvents	2
Victimisation	2
ViolentThoughts	4
...	...

Una vez conocidos los parámetros, aplicamos los algoritmo de aprendizaje de estructuras.

Comenzamos con los algoritmos basados en pruebas de independencia condicional. Los algoritmos que aplicamos son el *PC* y *Grow Shrink* con los tests de independencia información mutua y *Pearson* X^2. Y estas son las estructuras aprendidas por cada algoritmo (*figuras 47 y 48*)

Figura 48 Estructuras aprendidas por el algoritmo PC con pruebas de independencia Pearsons's x2 e Información Mutua (de arriba abajo

Figura 49 Estructuras aprendidas por el algoritmo GS con pruebas de independencia Pearsons's x2 e Información Mutua (de arriba a abajo)

Seguimos con los algoritmos basados en puntuación que son *Hill-Climbing* y *Tabu Search* usando como medidas de puntuación BIC, AIC, BDeu (*figuras 49 y 50*)

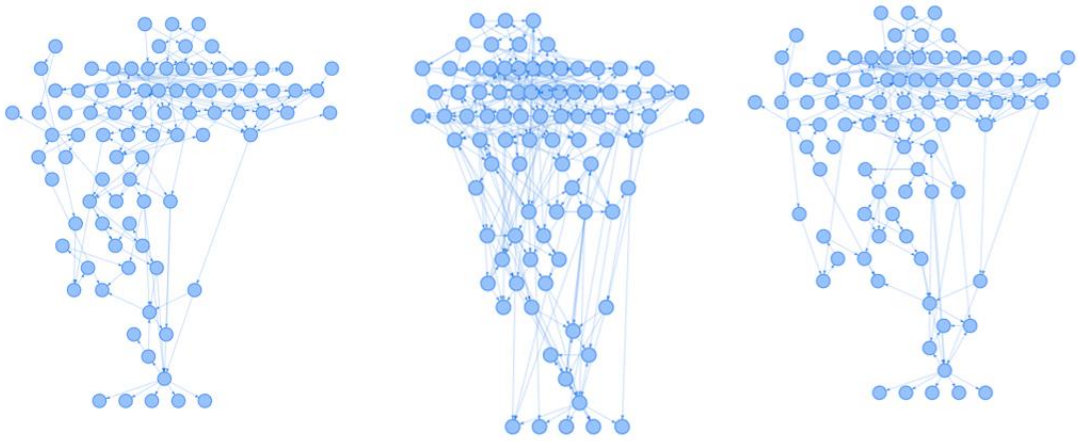

Figura 50 Estructuras aprendidas por el algoritmo HC con las puntuaciones BIC, AIC y BDeu (de izq a drch)

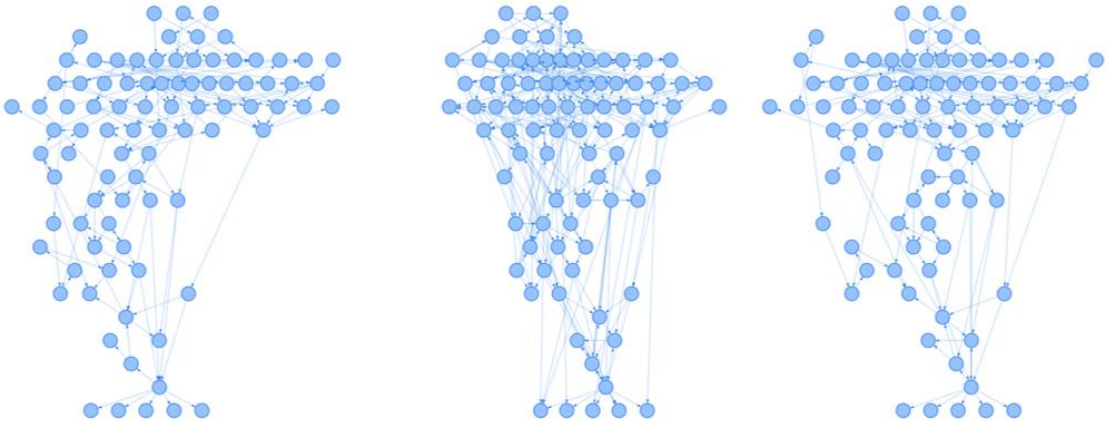

Figura 51 Estructuras aprendidas por el algoritmo TABU con las puntuaciones BIC, AIC y BDeu (de izq a drch)

Tabla 13 Puntuación algoritmos aprendizaje red Bayesiana grande

Puntuación		
	Hill-Climbing	Tabu Search
BIC	-421750.3	-421682.4
AIC	-424186	-424210.7
BDeu	-422185.9	-422118

La puntuación máxima para el algoritmo *Hill-Climbing* y *Tabu Search* es AIC, por lo tanto es la puntuación que mejor ayudará al algoritmo a capturar la estructura de red correcta.

Por último implementaremos el algoritmos hibrido *Max-Min Hill Climbing,* que aprende la siguiente estructura:

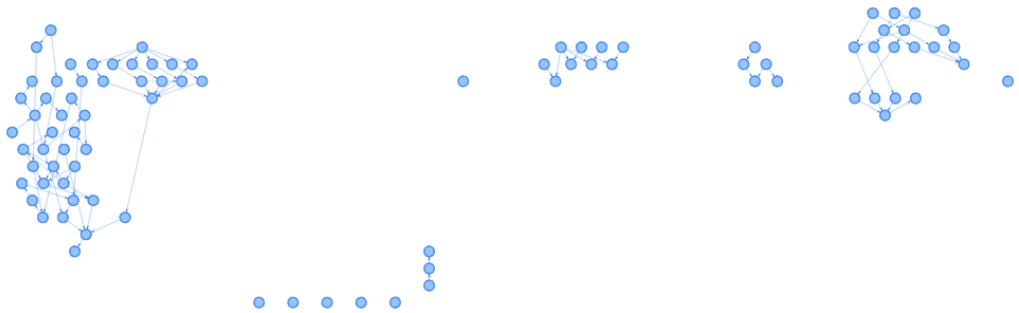

Figura 52 Estructuras aprendidas por el algoritmo MMHC

Solo los grafos aprendidos con algoritmos de puntuación han captado las relaciones con mayor significatividad que anteriormente calculamos a través de la fuerza de los arcos.

A continuación, se aprenderán los parámetros de todos algoritmos usando el algoritmo de aprendizaje de parámetros frecuentitas *máximum likelihood* y obtendremos las distribuciones de probabilidad condicional de cada nodo.

Los métodos basados en aprendizaje de pruebas de condición independiente incluyen ciclos en la red y por lo tanto se ha elegido de manera aleatoria una dirección del arco para poder proceder con el aprendizaje de parámetros.

4 Conclusiones

En este último punto del estudio, se analizarán los resultados de los experimentos mediante las métricas elegidas y se seleccionarán aquellos algoritmos que son más adecuados para el aprendizaje de estructuras y parámetros en las redes Bayesianas.

Además se expondrán los puntos para mejorar un siguiente trabajo relacionado con el aprendizaje de redes Bayesianas y su alineación con los objetivos de desarrollo sostenible (ODS).

4.1 Resultados de los experimentos

El mejor metodo de aprendizaje será aquel que maximice la tasa de verdaderos positivos y minimice los falsos arcos positivos y negativos. Además aquel que tengan la mínima distancia estructural de *Hamming* y cuyo valor de la función de puntuación equilibrada sea más cercano a 1. Por último, aquel que menor distancia de *Kullback-Leibler* obtenga tras el aprendizaje de la distribucción de probabilidad.

En primer lugar, en la *tabla 14* se observan los valores dados por las tasas de verdaderos positivos, falsos positivos y falsos negativos:

Tabla 14 Resultados de los arcos aprendidos para cada tipo de red

	Red pequeña			Red mediana			Red grande		
Algortimos	VP	FP	FN	TP	FP	FN	TP	FP	FN
PC Pearson's chi-cuadrado	2	7	13	10	9	21	47	46	91
PC Información Mutua	2	7	13	11	8	20	48	45	90
Grow Shrink Pearson's chi-cuadrado	4	4	11	5	8	26	22	40	116
Grow Shrink Información Mutua	4	4	11	2	9	29	18	41	120
Hill Climbing BIC	9	0	6	15	20	16	98	40	40
Hill Climbing BDeu	9	0	0	17	19	19	104	45	45
Hill Climbing AIC	15	0	0	16	23	23	109	117	117
Tabu Search BIC	8	2	7	16	17	15	98	42	40
Tabu Search BDeu	8	2	7	18	17	13	104	47	34
Tabu Search AIC	15	0	0	18	21	13	109	118	29
Max-Min Hill Climbing	9	0	6	11	7	20	73	17	65

- Para la Red Pequeña:

  Hill Climbing AIC y Tabu Search AIC aprenden el 100% de los arcos presentes en la red original y no generan ningún arco inexistente ni con una dirección incorrecta.

  El algoritmo *Max-Min Hill Climbing* no genera ningún arco inexistente, pero su rendimiento en las otras métricas es peor.

- Para la Red Mediana:

  Los algoritmos *Tabu Search* con las puntuaciones BDeu y AIC, aprende el 50% de los arcos presentes en la red original aunque genera 17 arcos nuevos y 13 arcos con la dirección incorrecta.

  El algoritmo *Max-Min Hill Climbing* es el que menos arcos inexistentes crea (7), pero su rendimiento en las otras métricas es peor.

- Para la Red Grande:

  Los algoritmos *Hill Climbing AIC* y *Tabu Search AIC* aprenden un 78% de los arcos presentes en la red original, aunque ambos generan aproximadamente 110 arcos más que no existen en la red

original. Además el algoritmo *Hill Climbing AIC* aprende 117 arcos con la dirección incorrecta, por lo que el algoritmo *Tabu Search* es un mejor algoritmo ya que solo genera 29 arcos con la dirección incorrecta.

En este caso también tenemos que el algoritmo *Max-Min Hill Climbing* es el algoritmo que menos arcos adicionales aprende (17 arcos) en comparación con *Hill Climibing AIC* y *Tabu Search AIC* (117 arcos y 118 arcos).

Como se ve que hay mucha variabilidad en el rendimiento de cada algoritmo con respecto a las métricas anteriores. Se va a usar una medida más restrictiva que es SHD que calcula cuantos arcos difieren entre las CPDAGs y por lo tanto nos dice cuantos arcos tienen que ser transformados para que la red se parezca a la original (*figura 52*)

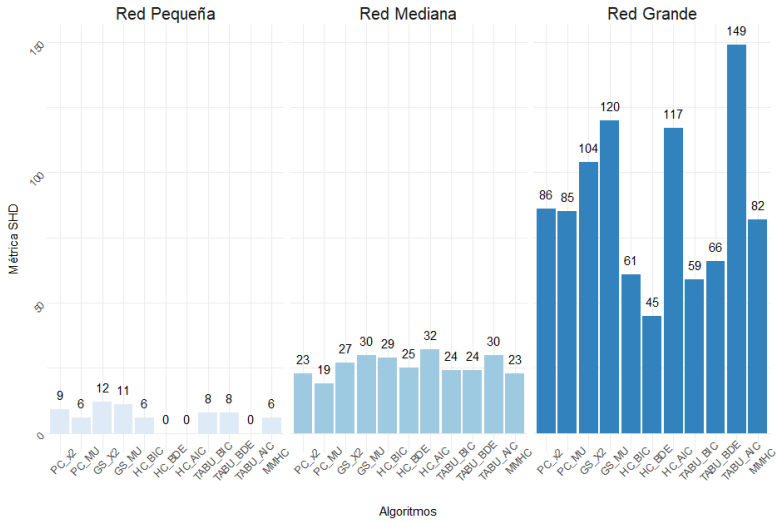

Figura 53 Comparativa Distancia Estructural de Hamming

Los algoritmos que menos distancia estructural de Hamming presentan son:

- Para la Red Pequeña

  Los algoritmos *Hill Climbing* con las puntuaciones AIC y BDeu y el algoritmo *Tabu Search* con la puntuación AIC, no necesitan que ningún arco cambie para que las redes aprendidas sean iguales a las originales.

- Para la Red Mediana

  El algoritmo *PC* con la prueba de independencia de información mutua es aquel que menos cambios necesita en sus arcos (19) para que se parezcan a la red original, aunque anteriormente se ha visto que sus métricas no eran los suficientemente correctas.

- Para la Red Grande:

  El algoritmo *Hill Climbing* con la puntuación BDeu solo necesita 45 arcos transformados para tener la misma estructura que la red original.

La métrica SHD está sesgada y no tiene tan en cuenta las dependencias directas correctas aprendidas pero la función BSF equilibra esto dándole la misma importancia a las dependencias directas aprendidas y aquellas con la correcta dirección.

En la *figura 53* se ve el resultado de la función BSF para cada red:

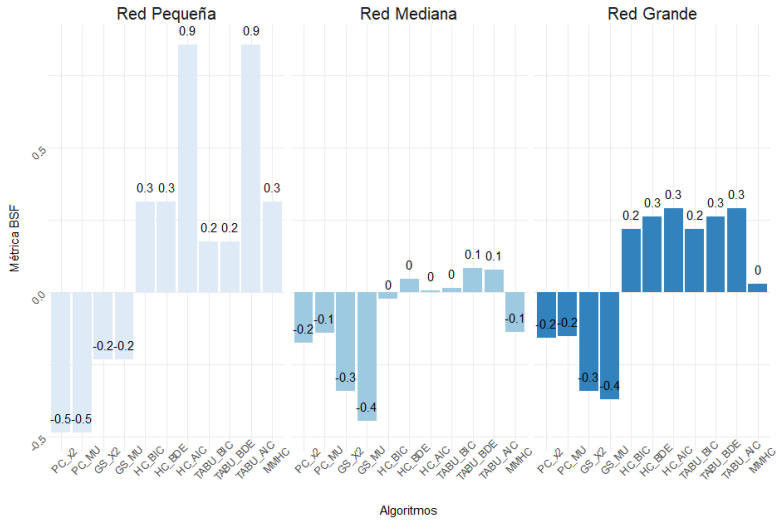

Figura 54 Comparativa función de puntuación equilibrada

- Para la Red Pequeña

  Los algoritmos *Hill Climbing* con las puntuaciones AIC y el algoritmo *Tabu Search* con la puntuación AIC aprenden todas las dependencias directas con la correcta dirección.

- Para la Red Mediana

  Ninguno de los algoritmos aprende de manera correcta las dependencias directas puesto que ninguno supera el 0.1.

- Para la Red Grande:

  Todos los algoritmos tienen una evaluación pobre para esta función, aunque podemos destacar que los mejores algoritmos son *Hill Climbing Bdeu, AIC y Tabu Search Bdeu y AIC*.

Por último, se observa la distancia de *Kullback-Leibler* que compara las distribuciones de probabilidad condicionadas aprendidas por el método de maximización de verosimilitud para el aprendizaje de parámetros con los parámetros de la red original.

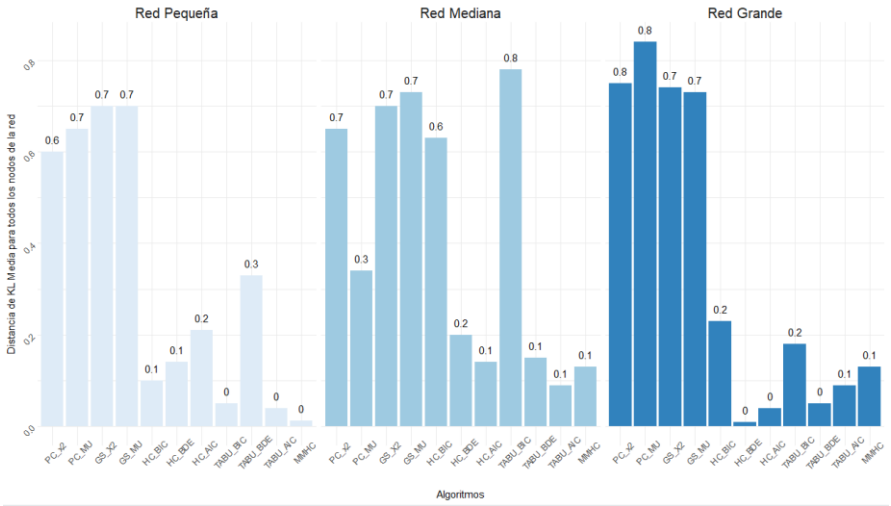

Figura 55 Comparación distancias de Kullback-Leibler

Las estructuras que mejor aprenden los parámetros son los aprendidas por los siguientes algoritmos:

- Para la Red Pequeña: Todos los algoritmos de aprendizaje de puntuaciones e híbridos, destacando *Hill Climbing y Tabu Search* con la puntuación AIC y MMHC.
- Para la Red Mediana: Destacan los algoritmos *PC* con la prueba de información mutua, los algoritmos *Hill Climbing* con las puntuaciones BIC AIC. Y los algoritmos *Tabu Search* con las puntuaciones BDeu y AIC y el algoritmo híbrido MMHC.
- Para la Red Grande: Todos los algoritmos de aprendizaje de puntuaciones e híbridos, destacando *Hill Climbing* con las puntuaciones BDeu y AIC y *Tabu Search* con las puntuaciones BDeu y AIC y el algoritmo híbrido *MMHC.*

Finalmente, en la *tabla 15* se visualizará cual es le mejor algoritmo dependiendo de las métricas elegidas y del objetivo, aprender una correcta estructura, distribución de probabilidad o ambas:

Tabla 15 Evaluación de las redes según su tamaño.

	Red Pequeña						Red Mediana						Red Grande					
Algoritmos	VP	FP	FN	SHD	BSF	KL	VP	FP	FN	SHD	BSF	KL	VP	FP	FN	SHD	BSF	KL
PC X2																		
PC MU										X		X						
GS X2																		
GS MU																		
HC BIC						X						X						X
HC BDeu				X		X						X				X	X	X
HC AIC	X	X	X	X	X	X							X				X	X
TS BIC						X						X						X
TS BDeu						X	X		X			X					X	X
TS AIC	X	X	X	X	X	X	X		X			X	X		X		X	X
MMHC		X				X		X				X		X				X

- Para una Red Pequeña:

Los algoritmos que mejor aprenden la estructura de la red son los basados en aprendizaje de puntuaciones en concreto *Hill Climbing y Tabu Search* con la puntuación AIC.

Los algoritmos que mejor aprenden la distribución de probabilidad son todos los algoritmos basados en puntuaciones e híbridos son adecuados.

Por lo tanto, para ambos aprendizajes se eligen los algoritmos *Hill Climbing y Tabu Searc*h con la puntuación AIC.

- Para una Red Mediana:

Todos los algoritmos tienen algún fallo a la hora de aprender la estructura completa, por lo tanto, basándonos en el número de arcos verdaderos y falsos negativos los algoritmos elegidos son *Tabu Search* con las puntuaciones BDeu y AIC. Y basándonos en la distancia estructural de Hamming el algoritmo *PC* con la prueba de información mutua.

Los algoritmos que mejor aprenden la distribución de probabilidad son casi todos los basados en puntuaciones y el algoritmo *PC* basado en pruebas de independencia.

Por lo tanto, para ambos aprendizajes se eligen los algoritmos *Hill Climbing y Tabu Search* con la puntuación AIC y BDeu y el algoritmo *PC* con información mutua.

- Para una Red Grande:

Los algoritmos que mejor aprenden la estructura de la red son los basados en aprendizaje de puntuaciones en concreto *Hill Climbing* con la puntuación BDeu y Tabu Search con las puntuaciones AIC y BDeu.

Los algoritmos que mejor aprenden la distribución de probabilidad son todos los algoritmos basados en puntuaciones e híbridos son adecuados.

Por lo tanto, para ambos aprendizajes se eligen los algoritmos *Hill Climbing y Tabu Search* con las puntuaciones AIC y BDeu.

Por lo tanto, tras analizar todas las métricas se llega a la conclusión que los métodos de aprendizaje basados en puntuación y el método híbrido son los más adecuados a la hora de usar redes o conjuntos de datos con diferentes números de variables o nodos.

Estos métodos aprenden correctamente los parámetros de la red, aunque algunos de ellos no aprendan correctamente la estructura. No podemos generaliza, debido a la variabilidad de conjuntos de datos existentes y a la heterogeneidad del grado de las variables

El principal inconveniente de los métodos de aprendizaje por puntuación es su convergencia al máximo global o la mejor estructura, ya que no siempre está garantizada en muestras finitas y puede encontrar el máximo local.

Una de las principales ventajas es que son métodos más estables que los basados en pruebas de independencia. Además la mayoría de las puntuaciones tienen parámetros de ajuste, mientras que las pruebas de independencia condicional en su mayoría no cuentan con parámetros de ajuste. Por lo tanto, con los métodos de puntuación podemos ajustar el aprendizaje a nuestros objetivos.

Para las pruebas híbridas, la principal ventaja es que podemos mezclar las pruebas de independencia condicional y las puntuaciones, y por lo tanto podemos usar pruebas tanto frecuentitas como Bayesianas. Cogerán lo bueno de ambos métodos, la rapidez de los métodos de pruebas de independencia y la estabilidad de los métodos de puntuación. Aunque como inconveniente no es tan fácil configurar los parámetros de las pruebas y las puntuaciones.

4.2 Trabajo futuro

Como trabajo futuro, este estudio podría realizarse con datos continuos y por lo tanto aplicar discretización y además usar diferentes pruebas de independencia o puntuaciones que mejoren los algoritmos de aprendizaje.

En nuestro caso hemos usado variables ya discretizadas pero en un futuro para seguir con esta comparación de algoritmos estructurales sería buena opción usar variables continuas. Cuando contamos con variables continuas hay dos opciones podríamos usar Redes Bayesianas Gaussianas o discretizar nuestras variables.

El uso de técnicas de discretización de variables continuas provoca que solo se capten características aproximadas de las variables continuas. Sin embargo, induce a modelos que pueden utilizarse eficazmente para la inferencia probabilística y la toma de decisiones óptimas. [26]

El principal problema de la discretización de variables es encontrar un umbral que divida los valores para cada variable continua X, en un número finito de intervalos. Estos intervalos son los valores de la parte discretizada de X. La aproximación de este problema se basa en el principio de la longitud de descripción mínima (MDL). La puntuación MDL de una red se compone de dos partes. La primera parte mide la "complejidad" de la red, mientras que la segunda mide lo buena que es la red como modelo para los datos. Siendo el grafo óptimo aquel que minimice la puntuación MDL. Además el método MDL regula el número de parámetros aprendidos y evitar el sobreajuste.

Otro de los objetivos futuros de este trabajo es usar varias medidas de puntuación para comparar los algoritmos basados en puntuación y diferentes tests de independencia condicional para los algoritmos basados en condiciones, ya que podría mejorar la precisión de los modelos. Para el caso de las pruebas de independencia condicional usamos test paramétricos, es decir, se asume que la distribución sigue una Normal. Pero podríamos usar otras alternativas como los test semiparamétricos de Monte Carlo o los test de permutaciones, ya que mejorarían el aprendizaje de las redes. [27]

Los test de *Pearson' X2* y de información mutua se pueden implementar como pruebas semiparamétricas, los grados de liberad de X2 se estimarán mediante permutaciones. Las pruebas paramétricas utilizan las permutaciones para generar la distribución nula empírica y el valor de significación observado.

Estos procedimientos paramétricos son más eficientes en muestras o sub-muestras de pequeño tamaño, permiten al algoritmo aprender de múltiples conjuntos de datos distribuidos de forma no idéntica, por lo que disminuye el error de rechazar la hipótesis nula, es decir, que exista una relación de independencia en los datos y que realmente no exista.

Para las puntuaciones de los algoritmos de aprendizaje por puntuaciones existe un dilema; cada una de las puntuaciones mide de manera diferente y por lo tanto no es posible usarlas como medidas de ajuste iguales. Para solucionar este problema existe la opción de usar puntuaciones multiobjetivo.

4.3 Objetivos de Desarrollo Sostenible

Por último, se analizará como impactan los Objetivos de Desarrollo Sostenible en el estudio de los modelos de aprendizaje de Redes Bayesianas.

- Salud y Bienestar

Las redes Bayesianas ayudan a modelar sistemas como el rendimiento de servicios de emergencias médicas. Estos servicios presentan mucha incertidumbre debido a la gran variedad de problemas de gestión en el servicio público. Gracias a el uso de algoritmos de Redes Bayesianas se pueden captar relaciones de dependencia entre factores e incorporar información externa al análisis, como conocimientos del gestor del hospital. [28]

- Industria, innovación e infraestructura

Para reducir el impacto de la industria de la construcción en el medio ambiente, se han realizado análisis del ciclo de vida de puentes usando Redes Bayesianas. El uso de estos métodos matemáticos ha confirmado la eficacia de tratar con factores inciertos en la evaluación del desarrollo sostenible en puente. [26]

- Educación de calidad

Se han elaborado herramientas de apoyo en la toma de decisiones del tutor en el sistema educativo usando Redes Bayesianas. El uso de estos algoritmos incentiva a la innovación educativa mejorando el sistema de información en el ámbito educativo. [30]

5 Bibliografía

[1] Schrater, *Gibbs Sampling for Approximate Inference in Bayesian Networks,* 2011.

[2] K. P. Murphy, Machine Learning: A Probabilistic Perspective, Londres.

[3] R. C. Team, «R: A Language and Environment for Statistical Computing,» R Foundation for Statistical Computing, 2021. [En línea]. Available: https://www.R-project.org/.

[4] M. Scurati, «Learning Bayesian Networks with the bnlearn R Package,» *Journal of Statistical Software,* vol. 35, nº 3, pp. 1-22, 2010.

[5] S. Højsgaard, «Graphical Independence Networks with the gRain Package for R,» *Journal of Statistical Software,* vol. 46, nº 10, pp. 1-26, 2012.

[6] P. Aguilera, A. Fernández, R. Fernández, R. Rumí y A. Salmerón, «Bayesian networks in environmental modelling,» *Environmental Modelling & Software,* vol. 26, nº 12, pp. 1376-1388, Julio 2011.

[7] T. Verma y J. Pearl, «Causal networks: Semantic and Expressiveness,» *Machine Learning and Pattern Recognition,* vol. 9, nº 1990, pp. 69-76, Junio 2014.

[8] R. Neapolitan, Probabilistic reasoning in expert systems: theory and algorithms, John Wiley & Sons, Inc., 1990.

[9] J. Pearl, Causality: Models, Reasoning and Inference, Cambridge University Press, 2009.

[10] N. Friedman, M. Goldsszmidt y A. Wyner, «Data Analysis with Bayesian Neworks: A Bootstrap Approach,» *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence,* pp. 196-201, 2013.

[11] D. Maxbell Chickering, «A transformation characterization of equivalent Bayesian network structures,» de *Proceeding of the Eleventh conference on Uncertainty in artificial intelligence*, Montréal, 1995.

[12] I. M. Wortel, J. Textor y I. M. Wortel, «Testing Graphical Causal Models Using the R Package "dagitty",» *Current Protocols,* vol. 1, 2021.

[13] A. L. Madsen, F. Jensen, A. Salmerón, H. Langseth y T. D. Nielsen, «A parallel algorithm for Bayesian network structure learning from large data sets,» *Knowledge-Based System,* vol. 117, pp. 46-55, 2017.

[14] D. Koller y N. Friedman.

[15] P. &. G. Spirtes y C. &. S. Richard, Causal, Prediction, and Search, vol. 81, Londres: The MIT Press, 1993.

[16] C. a. Z. Glymour y K. a. S. Peter, «Review of Causal Discovery Methods Based on Graphical Models,» *Frontiers in Genetics,* vol. 10, 2019.

[17] D. Margaritis, *Learning Bayesian Network Model Structure from Data,* Pittshburgh, 2003.

[18] R. Blanco, P. Larrañaga y I. Inza, «Learning Bayesian networks in the space of structures by estimation of distribution algorithms,» *Int. J. Intell. Syst.,* vol. 18, pp. 205-220, 2003.

[19] L. M. de Campos, «A Scoring Function for Learning Bayesian Networks based on Mutual Information and Conditional Independence Tests,» *Journal of Machine Learning Research,* vol. 7, pp. 2149-2187, 2006.

[20] I. Tsmardinos, L. E. Brown y C. F. Aliferis, «The max-min hill-climbing Bayesian netwotk structure learning algorithm,» *Mach Learn,* nº 65, pp. 31-78, 2006.

[21] L. de Campos, J. M. Fernandez-Luna y J. Miguel Puerta, «Local Search Methods for Learning Bayesian Networks Using a Modified Neighborhood in the Space of DAGs,» vol. 2527, pp. 182-192, 2022.

[22] D. Koller y N. Friedman, Probabilistic Graphical Models: Principless and Techniques, Massachusetts, 2009.

[23] A. C. Constantinou, Y. Liu, K. Chobtham, Z. Guo y N. K. Kitson, «Large-scale empirial validattion of Bayesian Network structure learning algorithms with noisy data,» *International Journal of Approximate Reasoning,* vol. 131, pp. 151-188, 2021.

[24] G. Xiaoguang, X. Liu, Z. Wang y X. Ru, «mproved Local Search with Momentum for Bayesian Networks Structure Learning,» *Entropy,* vol. 23, nº 6, p. 750, 2021.

[25] S. Moral, A. Cano y M. Gomez-Olmedo, «Computation of Kullback-Leibler Divergence in Bayesian Networks,» *Entropy,* vol. 23, nº 9, p. 1122, 2021.

[26] A. C. Constantinou, Y. Liu, K. . Chobtham, Z. Guo y N. K. Kitson, «The Bayesys data and Bayesian Network repository,» Queen Mary University of London, 2020. [En línea]. Available: http://bayesian-ai.eecs.qmul.ac.uk/bayesys/.

[27] N. Friedman y M. Goldszmidt, «Discretizing Continuous Attributes While Learning Bayesian Network,» de *ICML*, 1996.

[28] I. Tsmardinos y G. Borboudakis, «Permutation Testing Improves Bayesian Network Learning,» de *Machine Learning and Knowledge Discovery in Databases*, Berlín, 2010, pp. 322-337.

[29] «A comparison of learning algorithms for Bayesian networks: a case study based on data from an emergency medical service,» *Artificial Intelligence in Medicine,* vol. 30, pp. 215-232, 2004.

[30] «Life Cycle Assesment of Bridges Using Bayesian Networks and Fuzzy Mathematics».

[31] «Innovación educativa. Sistemas de información y redes bayesianas en apoyo a la tutoría de alumnos de licenciatura.,» de *La innovación y el desarrollo sustentable en las organizaciones.*, Tresguerras, Instituto Politécnico Nacional, 2013, p. 335.

[32] D. Colombo y M. H. Maathuis, Order-Independent Constraint-Based Causal Structure Learning, Zurich: Peter Spirtes, 2014.

[33] L. Jianhua, «Divergences measure based on the shannon entropy,» *IEEE Transactions on Information Theory,* vol. 37(1), nº 86, pp. 145-151, 1991.

[34] S. Wasserkrug, R. Marinescu, S. Zeltyn, E. Shindin y Y. Feldman, «Learning the Parameters of Bayesian Networks from Uncertain Data,» *AAAI,* vol. 35, nº 13, pp. 12190-12197, 2021.