Universidad Nacional Autónoma de México

Facultad de Ciencias

REDES BAYESIANAS EN EL DIAGNÓSTICO MÉDICO

T E S I S

PARA OBTENER EL TÍTULO DE:

Licenciado en Matemáticas

PRESENTA:

Martín Jiménez García

TUTORA

Dra. Lizbeth Naranjo Albarrán

Ciudad Universitaria 2020

Agradecimientos

Quiero expresar mi agradecimiento a las personas que me han brindado su apoyo y cariño a lo largo de esta etapa de mi vida. Hay personas a las cuales no me alcanzara la vida para terminar de agradecerles su ayuda, compañía y cariño, este ultimo esfuerzo es por ustedes.
A mis padres Elia y Martin por su esfuerzo y por su cariño, por su guía y sus consejos es que hoy puedo estar aquí dando lo mejor de mi, de la manera que ustedes inculcaron en mí. Su ejemplo me sigue motivando a alcanzar mis metas y mejorar cada día.
A mis abuelos Enedelia, Mario, Elena y Raymundo, por siempre brindarme un consejo, un abrazo y una cálida historia, gracias por todo el amor que me han brindado, su compañía para mi es invaluable, los recuerdos con ustedes son tesoros que guardare toda mi vida.
A mi familia Alma, Guadalupe, Carolina, Roberto y Moisés, Por ser una guía para mi, por apoyarme y escucharme cuando la vida era compleja y necesitaba de ayuda. Agradezco su calidez y gran compañía en buenos y malos momentos, la mayor lección que recibo de ustedes es que la familia puede ser muy unida.
A mi hermano Andres, un compañero del que sigo aprendiendo todos los días, a pesar de ser el menor, por el entiendo el significado de hermano.
A mi asesora de tesis Lizbeth Naranjo Albarrán, gracias por tenerme paciencia y compartir conmigo sus conocimientos, por brindarme apoyo para la realización de mi tesis, sus consejos siempre me ayudaron a entender y elaborar mi tesis, gracias por todo su trabajo y su apoyo. Gracias a ustedes, a su ejemplo, consejos y compañía que me tienen es que hoy les dedico este trabajo el cual culmina una etapa en mi vida.
De manera muy especial quiero agradecer a todos los que siguen compartiendo experiencias y tiempo conmigo, a las maravillosas personas que han vivido y compartido momentos importantes conmigo, gracias por su cariño y amistad, gracias por pertenecer a este conjunto el cual considero mi familia, gracias a ustedes entiendo que la familia no es solo sangre, son los vínculos que generamos y alimentamos día a día, gracias a ustedes mis amigos Mariana, Rodrigo, Ariadna, Karen Sánchez, Karen Aguirre, Geovana y Lorena, hemos tenido grandes experiencias y momentos que guardare en mi memoria.
Gracias a todas aquellas personas con las que he compartido experiencias, se me hace justo hacer una mención a todas aquellas personas que he conocido, porque he aprendido algo de ellas laboralmente o personalmente, gracias a todos ustedes.
Mi gratitud para la facultad de ciencias, la cual moldeo mi pensamiento, me enseño mi camino profesional y me permitió conocer a las grandes personas que me formaron académicamente, profesores y doctores a los cuales admiro, mi eterna gratitud a la Universidad Nacional Autónoma de México, mi alma mater, por brindarme la mejor educación desde el bachillerato hasta la universidad, allí conocí y pace momentos los cuales apreciare toda mi vida y también que me moldearon. Conocí muchas personas y todo gracias a los espacios que la UNAM nos permitió ocupar.

Índice

1. Introducción **1**

2. Preliminares **3**

Teoría de Gráficas . . . 3

Estructura de un Grafo . . . 4

Forma de Árbol . . . 7

Conceptos de Probabilidad . . . 8

Teorema de Bayes . . . 9

La Regla de Bayes Combinando Evidencias . . . 10

Independencia Condicional . . . 11

Clasificadores . . . 11

Indicadores de un Buen Clasificador . . . 11

Clasificadores Bayesianos . . . 12

Algoritmo Naive Bayes . . . 13

Aproximación Naive Bayes . . . 13

3. Redes Bayesianas **14**

¿Qué son las Redes Bayesianas? . . . 14

Definición Formal y Propiedades . . . 15

Parámetros . . . 15

Estructura de las Redes Bayesianas . . . 16

D-Separación . . . 17

Axiomas de Independencia . . . 18

Propagación de Probabilidades . . . 19

Algoritmo de Propagación de Redes en Forma de Árbol . . . 19

Inferencia: Razonamiento probabilístico . . . 21

Propagación en Árboles . . . 22

Propagación en Redes Multiconectadas . . . 24

Aprendizaje de Clasificadores Bayesianos . . . 24

Aprendizaje de Redes Bayesianas . . . 25

Aprendizaje Paramétrico . . . 26

Aprendizaje Estructural . . . 27

4. Redes Bayesianas en Diagnóstico Médico **29**

El Diagnóstico Médico . . . 29

Toma de Decisiones . . . 29

Conocimiento Experto . . . 30

Cardiopatía y Enfermedad Cardiovascular . . . 30

Hipótesis . . . 31
Estudiando la Información . . . 32
Descripción de la Base de Datos . . . 33
Análisis Exploratorio de la Información . . . 34
Implementación de los Algoritmos . . . 38
Aprendizaje de las Redes Bayesianas . . . 38
Red Bayesiana con Naive Bayes . . . 39
Red Bayesiana Naive Bayes Aumentado a Árbol TAN . . . 41
Red Bayesiana de Conocimiento Experto . . . 45
Árboles de Decisión . . . 46
Naive Bayes . . . 50
Máquinas de Soporte Vectorial . . . 55
Máquina de Soporte Vectorial (SVM) en nuestra data . . . 58

5. Concluciones **61**
Interpretación de los resultados . . . 62
Mira a un Futuro . . . 62

Capítulo 1

Introducción

La estadística, es la rama de las matemáticas que estudia la variabilidad, donde la recopilación, la interpretación y la validación de datos son un conjunto de técnicas estadísticas que buscan elaborar diferentes modelos que ayuden a entender el fenómeno estudiado, a este conjunto de técnicas lo nombramos *análisis estadístico*. El análisis estadístico nos permite detectar comportamientos no evidentes y basándose en los mismos, establecer predicciones. El resultado de estos permite el estudio de datos que a simple vista no muestran un orden o variables con las cuales poder discriminarlos, es por ello que el análisis estadístico es ampliamente usado en diferentes diciplinas. En diversas áreas del conocimiento como la medicina, en donde no se tiene la certeza plena del sistema, aunque las relaciones entre el estado interno y las observaciones externas puedan dar ciertas pistas de cuál es el posible síntoma del sistema, proceso que puede ser optimizado con el uso de las herramientas estadísticas. Por ejemplo, un médico sabe que no siempre se presentan los mismos síntomas para una misma enfermedad, sin embargo, existe cierta probabilidad de que cada síntoma para una misma enfermedad aparezca en todos los pacientes, así mismo, podemos poner un ejemplo para la industria, supongamos que una máquina falla, eso no quiere decir que todas las máquinas fallarán de la misma manera o que presentarán cuadros de fallos similares, y aun así, en ambos casos podemos obtener diagnósticos acertados en forma eficiente. Este proceso se llama **razonamiento bajo incertidumbre** y es hoy en día una de las áreas de gran actividad en las ciencias computacionales. Ésta en particular ha tenido mucha atención porque ha logrado buenos resultados y porque apoya a muchas áreas que suponen un impacto en el desarrollo de la modernidad como lo es la medicina, la economía y la industria.
Es por esto, que hoy en día las **redes Bayesianas** son una herramienta poderosa para la comprensión de la información bajo la incertidumbre, a pesar de problemas, como lo pueden ser: **datos faltantes** o **ruido** dentro de los **datos**, por comodidad le diremos **data** a la información que vamos a analizar. Las redes Bayesianas actualmente es un algoritmo que va en aumento su uso e implementación, esto se debe a su facilidad de interpretación y diversos métodos de inferencia y aprendizaje. El cometido de esta tesis es el poder modelar el diagnóstico de una enfermedad de corazón, bajo una red bayesiana habiendo ausencia de la información y manejando un error estándar.

Modelos Gráficos Probabilísticos

Los Modelos Gráficos Probabilísticos son básicamente una representación gráfica de la incertidumbre entre las variables de nuestro conjunto de información o que en la literatura también podemos encontrar como **data**. Ayuda a determinar probabilísticamente el estado de una variable con rela-

ción a su dependencia con una o más variables del conjunto de datos, así es como podríamos definir los modelos gráficos probabilísticos. El objetivo de la tesis será poder representar gráficamente el modelo probabilístico de la información que tenemos para que el médico tenga una toma de decisión basada en el método científico. Esto se puede hacer representando la información disponible en la forma de un modelo gráfico.

Redes Bayesianas aplicadas a la medicina

Mencionamos anteriormente que las redes Bayesianas son usadas más como una herramienta para sostener algún hecho con evidencias, ya que estas evidencias tienen un significado al interpretarlo a la realidad. A esto se le denomina **Medicina basada en la evidencia (MBE)** que es explicada como *el uso consciente, explícito y juicio de la mejor evidencia científica disponible para la toma de decisiones sobre los pacientes*. En base a diversas pruebas podemos hacer uso de la información arrojada para encontrar la debida clasificación y así con ello ahorrar tiempo, básicamente lo que la red bayesiana hace es aprender de manera supervisada de nuestra información y nuestras variables aplicadas a un modelo. El principal objetivo de la *MBE* es poder ofrecer un punto de vista mucho más interdisciplinario para la problemática de la salud con una mayor evidencia científica y poder sustentar en ella una decisión médica importante.

Red bayesiana para diagnósticos médicos

El objetivo de esta tesis será que mediante la información de una base de datos, obtenida de *Medical Center, Long Beach and Cleaveland Clinic*, analizaremos las variables que contenga el *data frame*. Posteriormente con el algoritmo de *Naive Bayes* clasificaremos la información en subconjuntos para el análisis de la información mediante la Red Bayesiana y conforme a los resultados poder establecer una matriz de confusión para la predicción, con la finalidad de poder clasificar debidamente a los pacientes y en caso de tener más pacientes o pacientes externos poder establecer una debida clasificación. Determinando en el camino, parámetros que nos permitan la inferencia de nuestra red, exponiendo los motivos que nos llevaron a escoger el algoritmo de Naive Bayes y porqué las redes Bayesianas sobre otros modelos de árboles nos ofrecen ciertos beneficios. Parte importante de que la inferencia en la red funcione con un error bajo, será el conocimiento experto vertido en la red, ya que las dependencias estarán fuertemente ligadas al conocimiento de médicos, que a lo largo del tiempo han establecido ciertos criterios de relación y correlación entre las variables expuestas.

Capítulo 2

Preliminares

En este capítulo daremos la introducción a los conceptos básicos de la **teoría de gráficas** para poder entender el modelo de estudio de este trabajo.

Teoría de Gráficas

En esta sección definiremos los conceptos básicos de la teoría de gráficas, así como algunas propiedades de las mismas, con la finalidad de apoyar algunos resultados posteriores.
Las **redes Bayesianas** obtienen su estructura de la teoría de gráficas y se apoya de muchos resultados de la misma.

Definición 1 (Grafo) *Un grafo es un objeto matemático definido como una dupla* $\mathbb{G} = (V, A)$*, donde* $V = (V_1, V_2, \ldots, V_n)$ *es un conjunto finito y no vacío, y* A *un conjunto finito, que puede ser vacío. Donde a los elementos de* V *se les llamarán vértices o nodos, y a los elementos de* A *aristas o arcos, descritos como* $a = (u, v)$ *que conectan a un par de nodos* $u, v \in V$*.*

Definición 2 (Vértices adyacentes) *Decimos que los vértices* u *y* v *en* V *del grafo* $\mathbb{G} = (V, A)$ *son adyacentes si están conectados mediante una arista* $a = (u, v) \in A$*.*

Dependiendo de la relación de orden que existe entre los nodos de un grafo, se puede hablar de dos tipos de arcos **dirigidos y no dirigidos** (figura 2.1) donde:

- **Dirigidos:** Si existe un arco de u a v, donde (u, v) es un par ordenado.
- **No Dirigido:** Si el arco (u, v) no es ordenado.

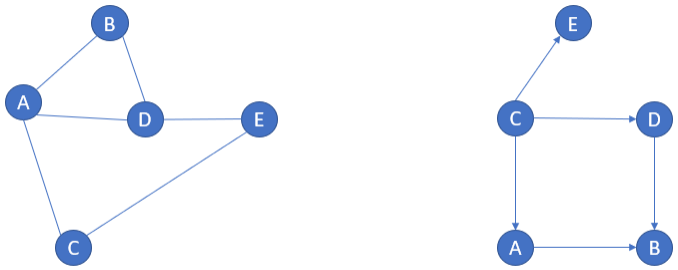


Figura 2.1: Grafo no dirigido (izquierda) y grafo dirigido (derecha).

Representamos a los arcos dirigidos como $u \to v$ y a los arcos no dirigidos como $u - v$; esta clasificación de los arcos nos induce una clasificación de grafos que mencionaremos a continuación.

Definición 3 (Grafos dirigidos) *Decimos que un grafo es dirigido cuando está definido por un par de conjuntos* $\mathbb{G} = (V, A)$ *donde:*

- $V \neq \varnothing$*, un conjunto no vacío de objetos simples llamados vértices o nodos.*
- *Es un conjunto de pares ordenados de elementos de* V *denominados aristas o arcos, donde por definición un arco va del primer nodo* u *al segundo nodo* v *dentro del par.*

Definición 4 (Grafos no dirigidos) *Para cuestiones prácticas definimos un grafo no dirigido como aquel grafo en los que* (u, v) *son nodos que no tienen una dirección entre ellos.*

Estructura de un Grafo

La estructura de un grafo es la configuración en que se encuentran las aristas de la misma. Recordemos que un grafo es la interpretación de la información por lo cual es de suma importancia conocer la estructura de la información, por lo que es fundamental estudiarla, ya que existen propiedades interesantes y valiosas en la estructura.

Definición 5 (Camino) *Sea* $\mathbb{G} = (V, A)$ *un grafo. Decimos que una sucesión de aristas* $w = a_1, a_2, a_3, \ldots, a_n$ *es un camino en* $\mathbb{G}$ *si* $w \subseteq A$*, y si* $a_i = (v_{i-1}, v_i)$*, decimos que entonces* v_i *es adyacente a* v_{i+1} *(figura 2.2).*

Ahora es importante que mencionemos en esta sección lo que es un ciclo ya que mucha de la información se anida en ciclos y su definición nos dirá cómo nos ayuda a interpretarla.

Figura 2.2: Camino.

Definición 6 (Ciclo) *Un camino* $w = v_0 - v_1 - \ldots - v_n$ *es un ciclo si el vértice inicial del camino coincide con el vértice final del misma es decir* $v_n = v_0$ *(figura 2.3).*

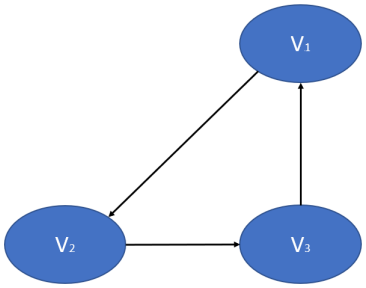


Figura 2.3: Ciclo.

El trabajo de esta tesis está orientado en su mayor parte a los grafos dirigidos por lo que ahondaremos más en este tipo de grafos, ya que existen agrupaciones y relaciones entre los vértices que definen conjuntos específicos como los que a continuación se muestran.

Definición 7 (Ancestro y descendiente) *Existen dos términos para definir a los nodos (figura 2.4):*

- *Un ancestro del vértice v_i, es cualquier vértice que tiene un camino hasta v_i.*
- *Un descendiente del vértice v_i es cualquier vértice al que podamos extender una trayectoria desde v_i.*

Definición 8 (Padres, Hijos) *A nivel de grafo (figura 2.4) definimos los siguientes términos para denotar elemento o conjuntos de elemento propios del grafo y así en futuros resultados explicar su importancia:*

- ***Nodo Padre:** Lo definimos como un camino entre los vértices v_i y v_j tal que en la trayectoria $v_i \to v_j$ v_i es el padre.*
- ***Nodo Hijo:** Lo definimos como un camino entre los vértices v_i y v_j tal que en la trayectoria $v_i \to v_j$ v_j es el hijo.*

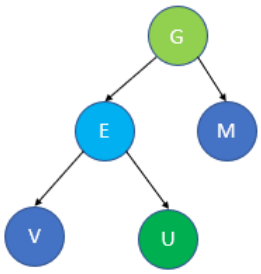


Figura 2.4: El nodo G es nuestro Ancestro, el nodo E es padre, el nodo u es hijo. Tenemos como nodo padre-hijo al nodo E.

Anteriormente definimos un grafo cíclico, por lo que nos referiremos como grafos acíclicos, V como un subconjunto de grafos tal que $T \subseteq V$, donde decimos que T no contiene ningún grafo con algún ciclo. Por lo que formalmente definiremos los DAG.

Definición 9 (Grafo dirigido acíclico) *Se dice que un grafo dirigido $\mathbb{G} = (V, A)$ es acíclico cuando no contiene ningún ciclo (figura 2.5).*

Con el fin de formalizar más la idea de los grafos daremos algunas definiciones que complementan a los anteriores como lo son **Grafos completos**, donde usaremos las definiciones anteriores.

Definición 10 (Grafo completo) *Decimos que el grafo G_C es completo si para cada par de nodos adyacentes existe un arco que los conecte (figura 2.6).*

Definición 11 (Grafo conexo) *Un grafo es conexo si cada par de vértices está conectado por un camino, es decir, si para cualquier par de vértices (a, b), existe al menos un camino posible desde a hacia b. Decimos que es no conexo cuando no está conectado el grafo y es fácil separarlo en 2 o más grafos (figura 2.7).*

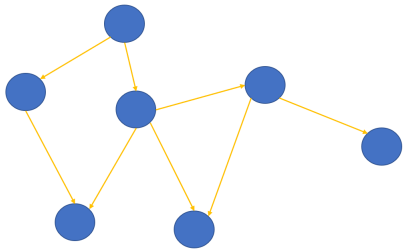


Figura 2.5: Grafo dirigido acíclico.

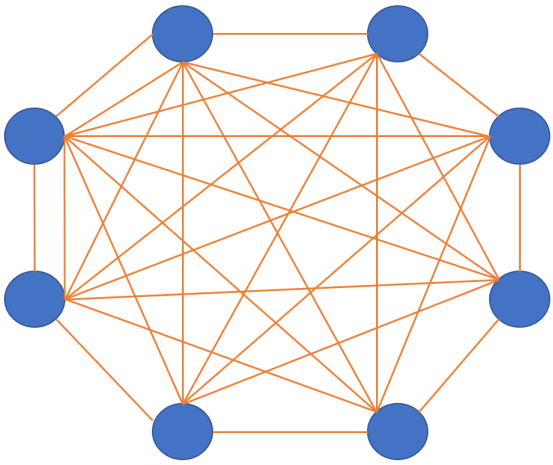


Figura 2.6: Grafo completo.

Esto quiere decir que para cada nodo le corresponde un arco tal que no existe nodo que no esté conectado entre sí. Lo que nos lleva a los **conjuntos completos** W_C, tal que el conjunto W_C es sub conjunto de G_C esto induce un subgrafo completo G, esto nos dice que para cada nodo en $A \in W_C$ existirá un arco que lo conecte.

Definición 12 (Cliques) *Sea el subconjunto C de G tal que es un grafo completo máximo. Es decir, no hay otro conjunto completo en G que contenga a C. Identificamos a los cliques $C_1, C_2, \ldots, C_k$ del grafo completo máximo como subconjuntos máximos de nodos en los que cada elemento es adyacente a todos los demás (figura 2.8).*

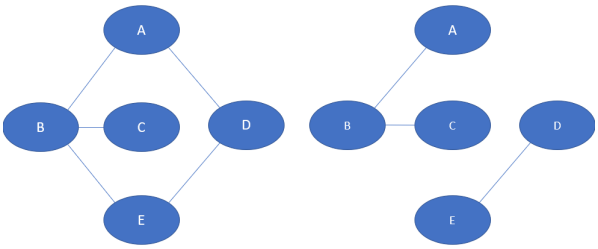


Figura 2.7: A) Grafo Conexo, B) Grafo no Conexo.

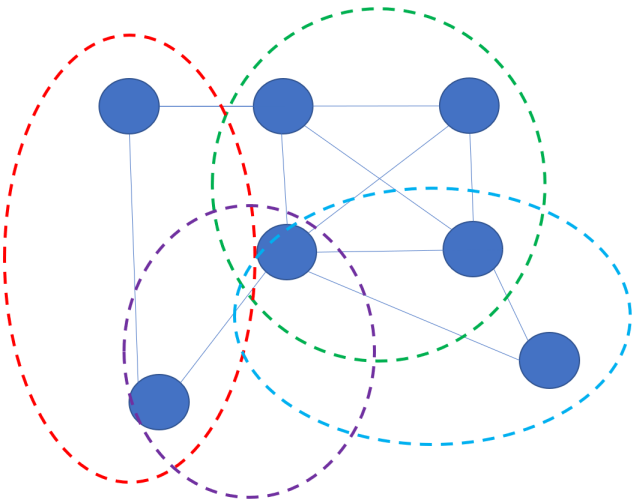


Figura 2.8: Clique.

Forma de árbol

En capítulos posteriores explicaremos lo que es el algoritmo de los árboles, en esta sección lo que nos interesa es explicar su topología.
Los árboles son un tipo de gráfico muy importante en la informática en general y para los modelos probabilísticos en particular son importantes, en particular nos permite visualizar las dependencias y ponderarlas. Discutiremos dos tipos de grafos tipo árbol en esta sección que son los **dirigidos y no dirigidos**, que como habíamos definido anteriormente obedecen a la definición de **grafos dirigidos y no dirigidos**, pero primero definamos formalmente qué es un árbol.

Definición 13 (Árbol) *Decimos que G es árbol si cumple lo siguiente:*

- *Todo par de vértices de G está unido por un único camino simple.*
- *Tiene un total de $n-1$ aristas.*
- *G es acíclico* ***(no tiene ciclos)****.*

Las anteriores condiciones son todas equivalentes, por lo que, si una de ellas se cumple, se cumplen todas las anteriores. Los árboles también tienen ciertas propiedades como son:

- Hay una trayectoria simple entre cada par de vértices.
- El número de vértices $|V|$ es igual al número de arcos más 1: $|V| = |A| + 1$.
- Un árbol con dos o más vértices tiene al menos dos nodos de hoja.

Los árboles se caracterizan por tener grados [1], el grado es explicado como el nivel de su padre más uno. Por definición el nodo raíz tiene nivel 0.
Ahora procederemos a definir los tipos de árboles que ocuparemos:

- **Árbol no dirigido:** Es un grafo conectado que no tiene circuitos.

[1] Nivel de un árbol: Es el "nivel" en el que se encuentra el nodo máximo.
Decimos que el nodo padre tiene nivel 1 y sus descendientes nodo 2, así sucesivamente.

- **Árbol dirigido:** Es un grafo dirigido conectado de tal manera que solo hay una única dirección dirigida.

Explicaremos brevemente lo que cada uno de estos árboles significa.

Árboles no Dirigidos

Dentro de los **árboles no dirigidos** (figura 2.9) existen dos tipos de nodos:

- **Nodos Hoja**, su grado es 1.
- **Nodos Internos**, su grado es mayor que uno.

Árboles Dirigidos

Hay dos tipos de árboles dirigidos (figura 2.9):

- Árboles arraigado o simplemente un árbol.
- Poli árbol.

El **Poli árbol** es un árbol especial, ya que es un árbol que puede tener más de un nodo raíz, lo que provoca que otros nodos puedan tener grado más de 1, a estos nodos los llamamos **multi padre**.

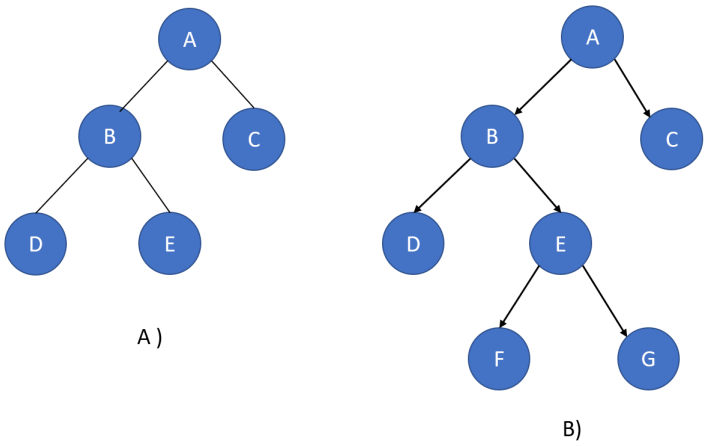


Figura 2.9: A) Árbol no dirigido, B) Árbol Dirigido.

Conceptos de Probabilidad

Para poder entender más a fondo las redes bayesianas debemos tener bien presentes algunos conceptos, principalmente las redes Bayesianas tienen un fuerte vínculo con la **teoría de probabilidad** y con ello muchos de sus resultados, para evitar un repaso de los resultados de probabilidad daremos por hecho que el lector tiene conocimientos de probabilidad pero enunciaremos los que consideramos más importantes.
Si A es un evento de interés, cuya probabilidad es $\mathbb{P}(A)$ y en paralelo tenemos que un evento B ha ocurrido, suponiendo que el evento A ocurre dado B tenemos la siguiente definición.

Definición 14 (Probabilidad condicional) *Para los eventos A y B, tal que $\mathbb{P}(B) \neq 0$, la probabilidad de A dado B es:*

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}. \tag{2.1}$$

Una definición que ocuparemos mucho será el término de independencia, esto quiere decir de manera más formal.

Definición 15 (Independencia) *Tenemos dos eventos A y B decimos que son independientes si se cumple que*

$$\mathbb{P}(A, B) = \mathbb{P}(A)\mathbb{P}(B). \tag{2.2}$$

Observacion 1 *Equivalentemente, A y B son independientes si $\mathbb{P}(A|B) = \mathbb{P}(A)$ con $\mathbb{P}(A) \neq 0$ y $\mathbb{P}(B) \neq 0$.*

A los eventos que no son independientes decimos que son dependientes. Los conceptos de dependencia e independencia se refieren a dos subconjuntos de variables, pero se puede generalizar el concepto cuando hay implicados más de dos conjuntos. Si X, Y y Z son tres conjuntos disjuntos de variables, entonces decimos que si X es condicionalmente independiente de Y dado Z, $I(X, Y|Z)$ sí y solo sí:

$$\mathbb{P}(x|z, y) = \mathbb{P}(x|z) \Leftrightarrow \mathbb{P}(x, y|z) = \mathbb{P}(x|z)\mathbb{P}(y|z). \tag{2.3}$$

Para todos los valores posibles de x, y, z.

Teorema de Bayes

Uno de los resultados más importantes en la teoría de probabilidad es el término de **probabilidad condicional**, resultados que conocemos y por lo tanto solo daremos los resultados sin meternos tanto en las demostraciones. De esta definición nace lo que conocemos como **teorema de Bayes** que normalmente se utiliza cuando no se puede determinar la probabilidad condicional de interés directamente. Mientras se cumplan los supuestos de sucesos disjuntos y exhaustivos, el teorema es valido.

Teorema 1 (Teorema de Bayes) *Si A es cualquier evento con probabilidad $\mathbb{P}(A) > 0$ y $B_1, B_2, \ldots, B_n$ es una partición de Ω , tal que $\mathbb{P}(B) \neq 0$ para todo $i = 1, \ldots, n$ $(1 \leq i \leq n)$, entonces:*

$$\mathbb{P}(B_j|A) = \frac{\mathbb{P}(A|B_j)\mathbb{P}(B_j)}{\mathbb{P}(A)} = \frac{\mathbb{P}(A|B_j)\mathbb{P}(B_j)}{\sum_{i=1}^{n} \mathbb{P}(A|B_i)\mathbb{P}(B_i)} \tag{2.4}$$

Donde tenemos los siguientes elementos:

- $\mathbb{P}(B_j|A)$ es la probabilidad a *posteriori* de B_j. Probabilidad de que sea cierta después de observar a A.
- $\mathbb{P}(A)$ es la probabilidad a *priori* de A, la probabilidad de observar A, sin saber qué hipótesis se verifica.
- $\mathbb{P}(B_j)$ es la probabilidad a *priori* de la hipótesis B_j.
- $\mathbb{P}(A|B_j)$ es la probabilidad a *posteriori* de A.

Como hemos dicho la red bayesiana se apoya de muchas otras disciplinas, como métodos computacionales que se someten a factores de optimización, lo que a grandes rasgos nos permite **diagnosticar** en función de nuestro conocimiento de relaciones **causales**. Una variante es cuando manejamos variables booleanas, para eso nos apoyaremos en un pequeño ejemplo para dejarlo todo más claro.

Ejemplo: 1 *Sabemos que la probabilidad de que un paciente de meningitis tenga el cuello hinchado es de 0.5 (denotado por la letra h es nuestra relación causal), por otra parte tenemos que la probabilidad (denotado por m es la relación no condicional) de tener meningitis ($\frac{1}{50000}$). Estas probabilidades provienen del conocimiento y la experiencia de los médicos, mediante esta información es que aplicaremos lo antes enunciado ya que el* ***Teorema de Bayes*** *nos permite diagnosticar la probabilidad de tener meningitis una vez que se ha observado que el paciente tiene el cuello hinchado, procedemos a aplicar tal cual la fórmula.*

$$\mathbf{P}(m|h) = \frac{\mathbf{P}(h|m)\mathbf{P}(m)}{\mathbf{P}(h)} = 0.0002. \tag{2.5}$$

Por lo que tenemos nuestro resultado.

Ahora conforme al ejemplo anterior podemos hacer una modificación. Como habíamos dicho queremos el caso en el que las variables son del tipo booleanas (1 ó 0), la ecuación cambia conforme al requerimiento y ahora incorporamos una α que sería nuestro factor de normalización, si M es una variable aleatoria booleana:

$$\mathbb{P}(m|h) = \alpha\mathbb{P}(Y|X)\mathbb{P}(Y). \tag{2.6}$$

Donde X, Y son variables aleatorias, esto genera la siguiente pregunta ¿Por qué calcular el diagnóstico en función del conocimiento causal y no al revés?, el motivo es sencillo es más fácil y robusto disponer de probabilidades causales que de probabilidades de diagnóstico. Usaremos la **regla de Bayes** para calcular $\mathbb{P}(causa|efecto)$ por lo que es importante definir la **regla de Bayes combinando evidencias**.

La Regla de Bayes Combinando Evidencias

Cuando manejamos varias variables para representar distintas evidencias, que es justamente nuestro caso, el uso de la regla de Bayes puede necesitar una cantidad exponencial de probabilidades de tipo $\mathbb{P}(efecto|causa)$. Por lo que si tenemos n variables booleanas de evidencia, deberíamos tener entonces 2^n probabilidades condicionales, esto traducido en hilos de procesamiento supone un gran reto computacional, es por eso que se desarrollan algoritmos de clasificación como **Naive Bayes** así mismo hablaremos de nociones de independencia entre variables para simplificar la cuestión. Supongamos dos v.a. X y Y, tal que ambas son independientes dada una v.a. Z si:

$$\mathbf{P}(X, Y|Z) = \mathbf{P}(X|Z)\mathbf{P}(Y|Z) \tag{2.7}$$

Solo será necesario saber las probabilidades causales de cada variable **por separado** y así reducimos la exponencialidad. Es importante indagar de manera muy general, lo definiremos como la **Independencia Condicional**.

Independencia Condicional

La independencia condicional entre algunas variables es esencial para un almacenamiento eficiente de las probabilidades, ahora solo enunciaremos el resultado importante de las independencias condicionales. Supongamos que tenemos una $Causa$ con n efectos, E_i independientes entre sí, dada la $Causa$ se tiene que:

$$\mathbb{P}(Causa|E_1, E_2, \ldots, E_n) \propto \mathbb{P}(Causa) \prod_i \mathbf{P}(E_i|Causa) \tag{2.8}$$

Resaltamos el hecho de que no siempre se dan las condiciones de independencia como para poder llevar a cabo lo que mencionamos anteriormente, por lo que la supondremos.

Clasificadores

Consiste en asignar clases (categoría) o capas a los objetos (instancia o dato), es una habilidad natural de nosotros los humanos, básicamente esto nos permite abstraer la información llevándola a una representación más adecuada para la toma de decisiones por ejemplo podemos clasificar a un animal según sea el caso de supervivencia, esto quiere decir que existen dos clasificaciones, el animal puede ser *presa* o *depredador*.
El uso de los clasificadores es muy basto y tiene un papel muy importante en el diseño de algoritmos computacionales que nos ayuden a clasificar de manera óptima la información que tenemos, es por eso que recae una gran importancia en diseñar clasificadores ya sea en *software* o *hardware* que nos ayuden a resolver dichos problemas.
Visto desde el punto matemático decimos que la clasificación consiste en asignar un vector $X = (x_1, x_2, x_3, \ldots, x_n) \in \mathbb{R}^n$ a una de las r_0 clases de la variable C. Las clases verdaderas se denotan por c y toman valores en $1, 2, 3, \ldots, r_0$. Se puede contemplar el clasificador como una función γ que asigna etiquetas a observaciones, es decir:

$$\gamma : (x_1, x_2, \ldots, x_n) \to \{1, 2, \ldots, r_0\} \tag{2.9}$$

Subyacente a las observaciones suponemos la existencia de una distribución de probabilidad conjunta:

$$\mathbb{P}(x_1, x_2, \ldots, x_n) \propto \mathbb{P}(c|x_1, x_2, \ldots, x_n) = \mathbb{P}(x_1, x_2, \ldots, x_n|c)\mathbb{P}(c) \tag{2.10}$$

$$\mathbb{P}(C|A) \propto \mathbb{P}(C)\mathbb{P}(A|C) \tag{2.11}$$

Existen dos grandes maneras de agrupar a los clasificadores que son:

- **No supervisados:** En este caso las clases son desconocidas, y el problema consiste en dividir un conjunto de n objetos en $k - clases$, de forma que a objetos *similares* se les asigna la misma clase.
- **Supervisados:** Las clases se conocen a *priori*, y el problema consiste en encontrar una función que asigne a cada objeto su clase correspondiente.

Indicadores de un Buen Clasificador

Para comparar las diferentes técnicas de clasificación necesitamos evaluar varios aspectos, los evaluaremos dependiendo de la aplicación ya que los factores a evaluar pueden tener pesos distintos según la necesidad de la aplicación. Los principales aspectos a considerar son:

- Precisión
- Tiempo de clasificación
- Tiempo de entrenamiento
- Número de muestras
- Claridad

El objetivo de esta tesis es enfocarnos en la clasificación supervisada, el problema estará en encontrar una función que realice un mapeo de los atributos del objeto a su clase correspondiente, en general es difícil construir dicha función por lo que emplearemos técnicas de aprendizaje computacional para obtener la función a partir de los datos.

Clasificadores Bayesianos

Desde un enfoque bayesiano, el problema de clasificación supervisada consiste en asignar a un objeto descrito por un conjunto de atributos o características $X_1, X_2, X_3, \ldots, X_n$ a una de m clases posibles $C = c_1, c_2, c_3, \ldots, c_m$, tal que la probabilidad de la clase dados los atributos se maximiza:

$$Arg_c[Max\mathbb{P}(C|X_1, X_2, \ldots, X_n)]. \tag{2.12}$$

Los atributos son denotados como $\mathbb{X} = X_1, X_2, X_3, \ldots, X_n$, tal que se puede escribir como $Arg_c[Max\mathbb{P}(C|\mathbf{A})]$. Existen muchas maneras de construir un clasificador, entre las que están incluidas muchas y muy famosas, por ejemplo **árboles de decisión, redes neuronales y máquinas vectoriales**, pero en la tesis nos enfocaremos en las **redes Bayesianas** y nos apoyaremos en los **árboles de decisión, bayesianos, TAN** con la finalidad de comparar su desempeño contra las redes Bayesianas. Utilizaremos el *software* libre R que cuenta con algunas librerías que en otra sección explicaremos más de ello.
La formulación de los clasificadores bayesianos se apoya fuertemente de la **Regla de Bayes** para estimar la probabilidad de cada clase dado sus atributos:

$$\mathbb{P}(C|A_1, A_2, A_3, \ldots, A_n) = \frac{\mathbb{P}(C)\mathbb{P}(A_1, A_2, A_3, \ldots, A_n|C)}{\mathbb{P}(A_1, A_2, A_3, \ldots, A_n)}. \tag{2.13}$$

Que puede ser escrita de manera más compacta como:

$$\mathbb{P}(C|\mathcal{A}) = \frac{\mathbb{P}(C)\mathbb{P}(\mathbf{A}|C)}{\mathbb{P}(\mathbf{A})} \tag{2.14}$$

Podemos expresar la ecuación anterior en términos de cualquier función que varíe momentáneamente con respecto a $\mathbb{P}(C|A)$, por ejemplo:

- $Arg_c[Max[\mathbb{P}(C)\mathbb{P}(\mathbf{A}|C)]]$.
- $Arg_c[Max[\log(\mathbb{P}(C)\mathbb{P}(\mathbf{A}|C))]$.
- $Arg_c[Max[\log\mathbb{P}(C) + log\mathbb{P}(\mathbf{A}|C)]]$.

Consideraremos una constante para la maximización a $\mathbb{P}(\mathbf{A})$ pues ésta no varía conforme a la clase en la que esté.
Ahora formulamos las equivalencias para maximizar la clasificación, pero realmente no hemos resuelto el problema por el que empezamos a clasificar, nos referimos a la estimación, intentamos estimar de la mejor manera la $\mathbb{P}(C)$ que es llamada la probabilidad a *priori* de las clases, y la $\mathbb{P}(\mathbf{A}|C)$ como la probabilidad o en la literatura encontrado como *verosimilitud*, por otra parte también tenemos la probabilidad a posteriori $\mathbb{P}(C|\mathbf{A})$ la cual se calcula multiplicando la probabilidad a *priori* por la probabilidad que depende de cada clase, el costo computacional de aplicar directamente el **Teorema de Bayes** y si el número de atributos es demasiado grande entonces el costo en memoria para el equipo se eleva exponencialmente, esto hace que este algoritmo funcione solo con *data* relativamente pequeñas en términos de atributos. Por lo que una manera de simplificar este costo será usar un algoritmo diferente.

Algoritmo Naive Bayes

El algoritmo Naive Bayes es uno de los clasificadores más utilizados por su simplicidad y rapidez, anteriormente mencionamos que el clasificador Bayesiano conlleva un gran consumo computacional que se resume en cálculos que ocupan memoria y procesador dependiendo la cantidad de operaciones que debe de manejar, este caso es distinto ya que se trata de una técnica de clasificación y predicción supervisada que construye modelos que calculan las probabilidad lo cual nos permite predecir posibles resultados, está basada fundamentalmente en el **Teorema de Bayes** el cual explicamos brevemente en la sección anterior.
El clasificador Naive Bayes o ingenuo se basa en el supuesto de que todos los atributos son independientes dada la variable de clase; es decir, cada atributo A_i es condicionalmente independiente de todos los demás atributos de la clase $\mathbb{P}(A_i|A_j, C) = \mathbb{P}(A_i|C)\ \forall j \neq i$. Bajo este supuesto retomamos la ecuación del clasificador bayesiano donde la podemos reescribir como:

$$\mathbb{P}(C|A_1, A_2, A_3, \ldots, A_n) = \frac{\mathbb{P}(C)\mathbb{P}(A_1|C)\mathbb{P}(A_2|C), \ldots, \mathbb{P}(A_n|C)}{\mathbb{P}(A)}. \tag{2.15}$$

Donde podemos considerar a $\mathbb{P}(A)$ como una constante de normalización. La formulación de Naive Bayes reduce drásticamente la complejidad computacional a la que el clasificador bayesiano nos sometía, ya que solo necesitamos la probabilidad a *priori* y el valor de los n atributos, esto también ayuda mucho al cálculo de la probabilidad a *posteriori* pues ayuda a un cálculo aún más ágil.

Aproximación Naive Bayes

Como bien sabemos lo que hace **Naive Bayes** es suponer que los valores de los atributos son condicionalmente independientes dado el valor de la clasificación:

$$\mathbb{P}(a_1, a_2, a_3, \ldots, a_n|c_j) = \prod_i \mathbb{P}(a_i|c_j) \tag{2.16}$$

Capítulo 3

Redes Bayesianas

El Teorema de Bayes fue desarrollado por el reverendo Thomas Bayes (figura 3.1) (1702 - 1761), el cual tiene un gran impacto en la inferencia estadística debido a que obtiene la probabilidad de una causa mediante la observación del efecto de la misma. El término **red Bayesiana** es establecido por el informático filósofo Judea Pearl en 1985, al extender el Teorema de Bayes a modelos gráficos de las relaciones probabilísticas entre muchas variables causalmente relacionadas. Las redes Bayesianas han tenido un gran impacto en la inferencia estadística y como aplicación sus logros y cobertura va desde el diagnóstico médico hasta la toma de decisiones para una empresa hotelera, por mencionar algunas. El objetivo de este capítulo será dar una breve introducción a las redes Bayesianas y de cómo se junta la información (*sintaxis*) y de cómo interpreta dicha información en la red (*semántica*), para posteriormente en el siguiente capítulo entender las intenciones de nuestro modelo, así como su procedimiento.

Figura 3.1: Thomaas Bayes (1702-1761).

¿Qué son las Redes Bayesianas?

Las redes Bayesianas son una manera práctica de representar la incertidumbre basada en probabilidades. Las redes Bayesianas o *red de Creencia*, es un modelo probabilístico multivariado que relaciona un conjunto de variables aleatorias mediante un grafo dirigido que indica explícitamente la influencia causal. Gracias a su motor de actualización de probabilidades, el **Teorema de Bayes**, las redes Bayesianas son una herramienta extremadamente útil en la estimación de probabilidades ante nuevas evidencias. Adicionalmente las redes Bayesianas también modelan el peso cualitativo de las conexiones entre las variables, permitiendo que las **creencias probabilísticas** sobre ellas se actualicen automáticamente, a medida que se disponga de nueva información. Al construir la

red Bayesiana nos hacemos muchas preguntas por lo que el objetivo de este capítulo será aclarar las más importantes, iniciando con definir qué es una red Bayesiana formalmente.

Propiedades de las redes Bayesianas

Definiremos lo necesario para conocer matemáticamente a una red Bayesiana. Fundamentando matemáticamente la red Bayesiana y dándole unión a los conceptos previos.

Definición 16 (Red Bayesiana) *Es un modelo probabilístico multivariado que relaciona un conjunto de variables aleatorias mediante un grafo dirigido que indica explícitamente influencia causal. Formalmente se definen como grafos dirigidos acíclicos cuyos nodos representan variables y los arcos que los unen representan dependencias condicionales entre las variables. Los nodos representan la distribución conjunta de* ***n*** *variables:*

$$X_1, X_2, X_3, \ldots, X_n$$

. Dichas variables las representan como un grafo acíclico dirigido (DAG), mientras que las dependencias arcos *construirán un conjunto de tablas condicionales de probabilidad, de tal manera que si existe un nodo A con otro nodo B, A es denominado un padre de B, y B es llamado un hijo de A. Por lo que si la distribución conjunta de los valores del nodo puede ser escrita como el producto de las distribuciones locales de cada nodo y sus padres:*

$$\mathbb{P}(X_1, \ldots, X_n) = \prod_{1=1}^{n} \mathbb{P}(X_i | padres(X_i)). \tag{3.1}$$

Parámetros

Para completar las especificaciones de una **red Bayesiana** necesitaremos definir cuáles son los parámetros en los que nos vamos a apoyar para construirla, nos referimos a las probabilidades condicionales de cada nodo dadas las probabilidades adyacentes a ese nodo, consideremos lo siguiente, bajo el supuesto de que las variables son discretas:

- **Nodo Raíz:** Es el vector al cual asignamos las probabilidades marginales.
- **Otros Nodos:** Armando las probabilidades condicionales generamos una CPT *Conditional Probability Table* de las variables dados los padres.

En el caso de que nuestras variables sean continuas necesitaremos especificar la función que relacione la función de densidad de cada variable a la densidad de sus padres. En el caso de que las variables sean discretas, tendremos que el número de los parámetros se verá en aumento de manera exponencial. Una forma de reducir tal cantidad de datos es utilizar ciertos modelos para representar las tablas sin requerir especificar todas las probabilidades, utilizando lo que se conoce como **Modelos Canónicos**. Los principales tipos de modelos canónicos son:

- Modelo de interacción disyuntiva **(noisy OR)**.
- Modelo de interacción conjuntiva **(noisy AND)**.
- Compuerta **max (noisy Max gate)**.

- Compuerta **min (noisy Min gate)**.

El modelo más común es el **Noisy OR** que se aplica cuando varias causas pueden ocasionar un efecto cada una por sí sola, y la probabilidad del efecto no disminuye si se presentan varias causas. Tomemos el ejemplo de una enfermedad, si se tiene que un modelo en donde tenemos varias enfermedades las cuales pueden producir el mismo síntoma esto quiere decir que se especifica un solo parámetro para cada nodo *padre* por lo que se consideran variables binarias. Una forma compacta de representar las CPT es con los árboles de decisión y redes neuronales, aparte de las redes Bayesianas.

Estructura de las Redes Bayesianas

La estructura ó topología de la red debe captar las relaciones cualitativas entre las variables, es decir, que las afirmaciones de independencia condicional implicadas por la estructura de una red Bayesiana deben corresponder a las relaciones de independencia condicional de la distribución de probabilidad conjunta y viceversa. Estos se representan generalmente utilizando la siguiente notación. Si **X es condicionalmente independiente de Z dado Y:**

- En la distribución de probabilidad: $P(X|Y,Z) = P(X|Y)$.
- En el grafo lo podemos expresar como: $X \perp\!\!\!\perp Y|Z$[1].

Existen estructuras dentro de las redes Bayesianas que nos dejan ver el tipo de condición de las variables, consideraremos 3, en una red Bayesiana con 3 (figura 3.2)variables y 2 arcos:

- **Secuencial**: $X \rightarrow Y \rightarrow Z$
- **Divergente**: $X \leftarrow Y \rightarrow Z$
- **Convergente**: $X \rightarrow Y \leftarrow Z$

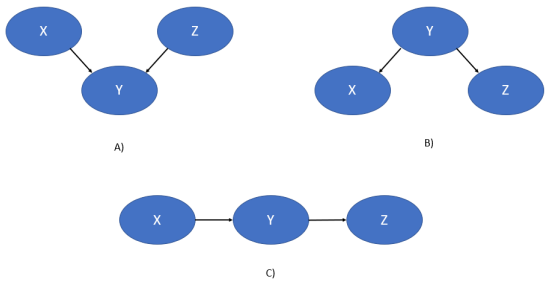


Figura 3.2: Estructuras de una red: A)Convergente , B)Divergente, C)Secuencial.

En el primer y segundo caso anteriormente mencionados tenemos que tanto X como Z son condicionalmente independientes dado Y, sin embargo, en el tercer caso esto no sucede. Este último caso corresponde a tener dos causas con un efecto en común. Es deseable construir redes Bayesianas lo más compactas posibles por tres razones:

- Mientras más compacto es el modelo, más fácil de manejar se vuelve.

[1] $\perp\!\!\!\perp$ Hace referencia a la independencia condicional en grafos, en la literatura es definida de manera general: $I < X|Y|Z >$. esta versión puede cambiar dependiendo de la estructura de la red

- Cuando las redes se vuelven demasiado densas fallan en representar la independencia de manera explícita.
- Las redes excesivamente densas no suelen representar las dependencias causales del dominio.

Es por lo que se opta por refinar lo más posible la red Bayesiana, con la cual nos apoyaremos de diferentes técnicas para visualizar o eliminar dicha saturación de información y no volver tan densa la red Bayesiana.

D-Separación

El método de la ***D-Separación*** nos permite comprobar si una red Bayesiana ha sido construida adecuadamente y se corresponde con la realidad que tratamos de modelar o al menos de forma significativa. El supuesto de Bayes nos dice: Cada variable es condicionalmente independiente de sus no descendientes, dados sus padres. Es ciertamente posible razonar sobre la independencia usando esta declaración, podemos usar la D-separación como un procedimiento más formal para determinar la independencia. Comenzaremos haciéndonos algunas preguntas con el fin de pensar en la independencia de alguna de estas formas:

- ¿Son X y Y condicionalmente independientes, dado Z?
- ¿Son X y Y marginalmente independientes?

Estas preguntas nos permiten seguir las independencias, con el fin de encontrar y mejorar nuestro grafo en cuestión de independencia. El método de **D-Separación** consiste en los siguientes pasos, una vez definido el grafo revisamos los siguientes incisos para corroborar las independencias de los nodos.

1. Dibuja la gráfica ancestral. Construya el *gráfico ancestral* de todas las variables mencionadas en la expresión de probabilidad. Esto es una versión reducida de la red original, que consiste solo en las variables mencionadas y todos sus ancestros (padres, padres de los padres, etc.)
2. *Moralizar* el gráfico ancestral al *casarse* con los padres. Para cada par de variables con un hijo común, dibuje un borde no dirigido (línea) entre ellas. (Si una variable tiene más de dos padres, dibuje líneas entre cada par de padres).
3. *Desoriente* el grafo reemplazando los bordes dirigidos (flechas) con bordes no dirigidos (líneas).
4. Eliminar los *givens* y sus bordes. Si la pregunta de independencia tenía alguna variable dada, borre esas variables del grafo y borra todas sus conexiones. Tenga en cuenta que **variables dadas** como se usa aquí se refiere a la pregunta "¿Son A y B condicionalmente independientes, dados D y F?", la ecuación $\mathbb{P}(A|BDF) = \mathbb{P}(A|DF)$, y por lo tanto no incluye B.
5. Lee la respuesta del grafo.
   a) Si las variables están desconectadas en este gráfico, se garantiza que son independientes.
   b) Si las variables están conectadas en este gráfico, no se garantiza que sean independientes.

Tengamos en cuenta que están conectados eso significa que tiene una ruta entre ellos, por lo que si tenemos una ruta X - Y - Z. Se considera que X y Z están conectados, incluso si no hay bordes entre ellos.

- Si falta una o las dos variables (porque fueron dadas, y fueron por lo tanto eliminados), son independientes.
- Podemos decir las variables son dependientes, en lo que respecta a la red de Bayes no requiere que las variables sean independientes, pero no podemos garantizar la dependencia usando la *D-separación* sola, porque las variables aún pueden ser numéricamente independientes (por ejemplo, si $\mathbb{P}(\mathbb{A}|\mathbb{B})$ y $\mathbb{P}(\mathbb{A})$ son iguales para todos los valores de A y B).

Dado un gráfico G, un conjunto de variables A es condicionalmente independiente de un conjunto B dado un conjunto C si no hay una trayectoria en G entre A y B dado:

- Todos los nodos convergentes son o tienen descendientes en C.
- Todos los demás nodos están fuera de C.

Manta de Márkov

Una vez que hemos dado el algoritmo de la **D-Separación** ahora definiremos un método gráfico para la independencia de los nodos padres e hijos, que formalmente se define de la siguiente manera.

Definición 17 (Manta de Márkov) *La manta de Márkov de un nodo X es el conjunto de nodos $M(X)$ compuesta por los nodos padres e hijos de X, así como también los nodos padres de los hijos de X. Esto quiere decir que la* ***Manta de Márkov*** *de un nodo es el conjunto de nodos que lo hacen independiente del resto de la red (figura 3.3).*

$$\mathbb{P}(X|M(X), Y) = \mathbb{P}(X|M(X)) \; \forall \, Y \; en \; la \; Red \tag{3.2}$$

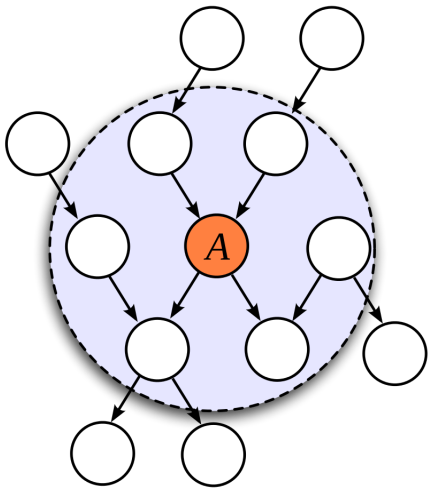


Figura 3.3: Manta de Markov.

Axiomas de Independencia

Definimos la independencia de eventos, ahora daremos algunos axiomas acerca de la independencia. A partir de ciertas relaciones de independencia se pueden derivar otras, sin necesidad de evaluar las probabilidades para eso se pueden utilizar ciertas reglas. Las reglas básicas se conocen como **Axiomas de Independencia**:

- **Simetría:**
$$I(X, Z, Y) \leftrightarrow I(Y, Z, X). \tag{3.3}$$
- **Descomposición:**
$$I(X, Z, Y \cup W) \rightarrow I(X, Z, Y) \wedge I(X, Z, W). \tag{3.4}$$
- **Unión débil:**
$$I(X, Z, Y \cup W) \rightarrow I(X, Z \cup W, Y). \tag{3.5}$$
- **Contracción:**
$$I(X, Z, Y) \wedge I(X, Z \cup Y, W) \rightarrow I(X, Z, Y \cup W). \tag{3.6}$$
- **Intersección:**
$$I(X, Z \cup W, Y) \wedge I(X, Z \cup Y, W) \rightarrow I(X, Z, Y \cup W). \tag{3.7}$$

Propagación de Probabilidades

Explicaremos la estructura de las *redes Bayesianas*, nos interesa ahora realizar consultas sobre las variables incluidas en las mismas. En el campo de los sistemas expertos el principal interés se centra en ver cómo los valores que toman ciertas variables interactúan con las probabilidades del resto. Es importante observar si intentáramos hacer estos cálculos con las herramientas que actualmente contamos, como lo son: *Teorema de Bayes, Independencia Condicional y los Teoremas de Probabilidad*, los cálculos crecerían de manera exponencial así como la relación entre las variables, esto se convertirá en un problema computacional, ya que se requerirían más recursos computacionales de los habituales, hasta volverse intratables. Por lo que definiremos los **algoritmos de propagación de probabilidades**, que utilizan las relaciones de independencias implícitas en la estructura de una red Bayesiana para calcular las probabilidades de cada uno de los nodos dada la evidencia disponible de una forma más eficiente, una vez calculadas estas probabilidades podemos extender su uso a un carácter más abductivo o predictivo según sea el caso. Para entender el funcionamiento nos tomaremos la libertad de tomar el algoritmo más simple que existe, nos referimos a el algortimo para *redes con forma de árbol*, que a su vez es el algoritmo que hemos usado para esta tesis.

Algortimo de Propagación de Redes en Forma de Árbol

El primer método de propagación para redes Bayesianas que se desarrolló es el algoritmo de propagación en árboles (Pearl, 1982). La idea consiste en modificar la información adecuada a un nodo, el cual traspasa la información a sus nodos vecinos mediante un conjunto de mensajes. Estos nodos a su vez procesan la información recibida junto con la que ellos poseen y la pasan a sus nodos vecinos (aún no modificados) y así sucesivamente hasta que todos los nodos han actualizado su información. El algoritmo consta de dos fases:

- **Fase de Inicialización**.
- **Fase de Actualización**.

La principal limitación del algoritmo es que solo se puede aplicar a redes con estructura de árbol, describiremos brevemente el desarrollo de cada una de las fases del algoritmo.

Fase de Inicialización

En esta fase se obtienen las probabilidades a *priori* de todos los nodos de la red, obteniendo un estado inicial de la red que denotaremos por S_0. Para uso de esta tesis usamos el algoritmo de **Naive Bayes** para generar tablas con las probabilidades condicionales, que para fines de la tesis vamos a ocupar como nuestras probabilidades a priori.

Fase de Actualización

Cuando tenemos una variable la cual se inicializa, conlleva a la actualización del estado de la red, obteniéndose las probabilidades a posteriori de las variables de la red basadas en la evidencia considerada adoptando la red un estado que denotaremos por S_1. Este paso se repite cada vez que una variable se inicializa, así es como obtenemos los estados de la red. Cada vez que una variable se inicializa, informa a sus nodos vecinos mediante el peso de lo que llamaremos *mensajes*, de la siguiente forma:

- La variable envía a su padre un mensaje que llamaremos $\lambda - mensaje$, para informarle de que ha cambiado su valor.
- La variable envía a todos sus hijos un mensaje, que llamaremos el $\pi - mensaje$, para informarles que ha cambiado su valor.

Así es como finalmente la información va propagándose en la red. Estos mensajes asignan a cada variable unos valores que llamaremos $\lambda - valor$ y $\pi - valor$. Multiplicando estos valores obtendremos las probabilidades a *posteriori* de cada una de las variables de la red.

Instanciar las variables

Los **valores** y los **mensajes** los definimos como vectores. Por ejemplo, supongamos que tenemos el modelo más básico que es: en el que la variable A toma tres valores posibles que denotaremos a_1, a_2 y a_3, por otra parte la variable B tomará los valores de b_1 y b_2. Tendríamos que:

- Al instanciar[2] B, se enviará un $\lambda - mensaje$ a A, $\lambda_B(A) = (\lambda_B(a_1), \lambda_B(a_2), \lambda_B(a_3))$.
- Al instanciar A, se enviará un $\pi - mensaje$ a B, $\pi_B(A) = (\pi_B(a_1), \pi_B(a_2), \pi_B(a_3))$.

En función de los valores tendremos un $\lambda - valor$ y un $\pi - valor$ para A:

- $\lambda(\mathbf{A}) = (\lambda(\mathbf{a_1}), \lambda(\mathbf{a_2}), \lambda(\mathbf{a_3}))$.
- $\pi(\mathbf{A}) = (\pi(\mathbf{a_1}), \pi(\mathbf{a_2}), \pi(\mathbf{a_3}))$.

Y también un $\lambda - valor$ y un $\pi - valor$ para B:

- $\lambda(\mathbf{B}) = (\lambda(\mathbf{b_1}), \lambda(\mathbf{b_2}))$.
- $\pi(\mathbf{B}) = (\pi(\mathbf{b_1}), \pi(\mathbf{b_2}))$.

Multiplicando los valores y normalizando, obtendremos las probabilidades asociadas a A o B, según sea el caso.

[2] Con *Instanciar* nos referimos al proceso de leer o especificar información, como los valores y el tipo de almacenamiento de un campo de datos, usado en su mayoría en el área de la informática mayormente usada en programación orientada a objetos.

Formas para Calcular los Mensajes y los Valores de P*

Dependiendo el suceso que tengamos en la red, tendremos una manera en la cual podemos proceder, enlistaremos los posibles casos, así como su manera de calcular los mensajes y valores para $P*$.

1. Si B es un hijo de A, B tiene k valores posibles y A tiene m valores posibles, entonces para $j = 1, \ldots, m$ el $\lambda - mensaje$ de B a A viene dado por:

$$\lambda_b(a_j) = \sum_{i=1}^{k} \mathbb{P}\left(\frac{b_i}{a_j}\right) \lambda(b_i).$$

2. Si B es hijo de A y A tiene m valores posibles, entonces $j = 1, ..., m$, el $\pi - mensaje$ de A a B viene dado por:

$$\pi_b(a_j) = \begin{cases} \pi(a_j) \prod \lambda_c(a_j) & \text{si } A \text{ no ha sido inicializada} \\ 1 & \text{si } A = a_j \\ 0 & \text{si } A \neq a_j \end{cases}$$

3. Si B tiene $k - valores$ posibles entonces para $i = 1, ..., k$, el $\lambda - valor$ de B viene dado por:

$$\pi_b(a_j) = \begin{cases} \prod_{C \in S(B)} \lambda_c(b_i) & \text{si } B \text{ no ha sido inicializada} \\ 1 & \text{si } B = b_i \\ 0 & \text{si } B \neq b_i \end{cases}$$

4. Si A es padre de B, B tiene k valores posibles y A tiene m valores posibles, entonces para $i = 1, \ldots, k$ donde le $\pi - valor$ de B viene dado por:

$$\pi(b_i) = \sum_{j=i}^{m} \mathbb{P}\left(\frac{b_i}{a_j}\right) \pi_B(a_j).$$

5. Si B es una variable con k posibles valores, entonces para $i = 1, \ldots, k$ la probabilidad a posteriori basada en las variables instanciadas se calcula como:

$$\mathbb{P}^*(b_i) \propto \lambda(b_i)\pi(b_i).$$

Para el caso general, se han desarrollado otros algoritmos. El problema de la propagación en redes Bayesianas es **NP-Duro**, lo que siginifica que no es posible obtener un algoritmo de complejidad polinomial para el problema de la propagación en redes Bayesianas con una topología general.

Inferencia: Razonamiento probabilístico

La principal utilidad de las redes Bayesianas consiste en hallar de forma eficiente la probabilidad de cualquier nodo (o conjunto de nodos), dada una cierta información (evidencia). Este proceso de razonamiento probabilístico permite cuantificar la incertidumbre de las distintas variables y eventos del problema a medida que se va teniendo nueva información o evidencia.
El razonamiento probabilístico o propagación de probabilidades consiste en propagar los efectos de la evidencia a través de la red para conocer la probabilidad a *posteriori* de las variables. Es decir, se le dan valores a ciertas variables (*evidencia*), y se obtiene la probabilidad posterior de las demás variables dadas (el conjunto de variables conocidas puede ser vacío, en este caso se obtienen las probabilidades a *priori*). Existen diferentes tipos de algoritmos para calcular las probabilidades posteriores, dependerá mucho del tipo de grafo del que estemos hablando, los principales tipos de algoritmos son:

- Algoritmo de eliminación.
- Algoritmo de propagación de Pearl.
- Agrupamiento *(junction tree)*.

Empezaremos definiendo la propagación por los árboles y poli árboles, revisaremos posteriormente los algoritmos de agrupamiento o árboles de unión.

Propagación en Árboles

Este algoritmo se aplica a estructuras del tipo árbol, y se puede extender a poli árboles (grafos sencillamente conectados en que un nodo puede tener más de un padre, figura 3.4). Dada cierta evidencia E, representada por la instancia de ciertas variables, la probabilidad posterior de cualquier variable B, por el **teorema de Bayes**:

$$\mathbb{P}(B_i|E) = \frac{\mathbb{P}(B_i)\mathbb{P}(E|B_i)}{\mathbb{P}(E)}. \tag{3.8}$$

Donde decimos que B_i será un nodo con el cual tomaremos de referencia para iniciar el algoritmo:

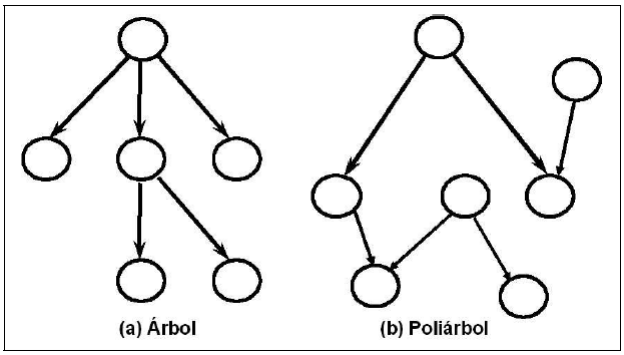


Figura 3.4: Árbol y Poli Árbol.

Ya que la estructura es un árbol y en la figura 3.5 se aprecia que el nodo B la separa de dos subárboles, por lo que podemos dividir la evidencia en dos grupos.

- **E- :** Datos en el árbol que cuya raíz es B.
- **E+ :** Datos en el resto del árbol.

A partir de esto modificamos la probabilidad de la siguiente manera:

$$\mathbb{P}(B_i|E) = \frac{\mathbb{P}(B_i)\mathbb{P}(E-, E+|B_i)}{\mathbb{P}(E)}. \tag{3.9}$$

Dado que ambos son independientes y aplicando nuevamente el **Teorema de Bayes:**

$$\mathbb{P}(B_i|E) = \alpha\mathbb{P}(B_i|E+)\mathbb{P}(E-|B_i). \tag{3.10}$$

Donde α es una constante de normalización como habíamos dicho anteriormente, por lo que definimos los siguientes dos términos más:

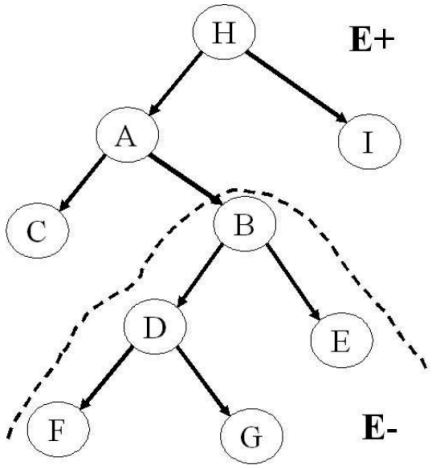


Figura 3.5: Propagación de Árboles, grafo dividido en 2 subgrafos.

$$\lambda(B_i) = \mathbb{P}(E - |B_i) \tag{3.11}$$

$$\pi(B_i) = \mathbb{P}(B_i|E+) \tag{3.12}$$

Entonces,

$$\mathbb{P}(B_i|E) = \alpha\pi(B_i)\lambda(B_i). \tag{3.13}$$

Con base en la ecuación anterior, se puede integrar un algoritmo distribuido para obtener la probabilidad de un nodo dada cierta evidencia. Para ello, se descompone el cálculo en dos partes:

- Evidencia de los Hijos (λ).
- Evidencia de los demás nodos (π).

Cada nodo guarda los valores de los vectores π y λ, así como las matrices de probabilidad $\mathbb{P}$. Para ello, se descompone el cálculo en dos partes: *(i) Evidencia de los hijos* (λ), y *(ii) Evidencia de los demás nodos* (π). La propagación se lleva a cabo por un mecanismo de *paso de mensajes*, en donde cada nodo envía los mensajes correspondientes a su padre e hijos. Mensaje al padre (hacia arriba, figura 3.5), nodo B a su padre A:

$$\lambda_B(A_i) = \sum_j \mathbb{P}(B_j|A_i)\lambda(B_j). \tag{3.14}$$

Mensaje a los hijos (hacia abajo, figura 3.5), nodo B a su hijo S_k

$$\pi_k(B_i) = \alpha\pi(B_i)\prod_{l\neq k}\lambda_l(B_j). \tag{3.15}$$

Al instanciarse ciertos nodos, estos envían mensajes a sus padres e hijos, y se propagan hasta llegar a la raíz u hojas, o hasta encontrar un nodo instanciado, al final de la propagación, cada nodo tiene un vector π y un vector λ.
Este algoritmo se puede extender fácilmente para poli árboles, pero no se aplica en redes múlticonectadas. En este caso hay varios algoritmos, es aquí donde entran los famosos *árboles de unión*.

Propagación en Redes Multiconectadas

En esta sección describiremos los métodos de agrupamiento o el algoritmo más famoso **Árboles de unión**, el cual consiste en transformar la estructura de la red para obtener un árbol, mediante agrupación de nodos usando la **Teoría de grafos**. Para ello se hace una transformación de una red a un árbol de uniones mediante el siguiente procedimiento (figura 3.6):

- Eliminar la direccionalidad de los arcos.
- Ordenamiento de los nodos por máxima cardinalidad.
- Moralizar el grafo (dibujar un arco entre nodos con hijos comunes).
- Triangular el grafo.
- Obtener los cliques y ordenar.
- Construir árboles de cliques.

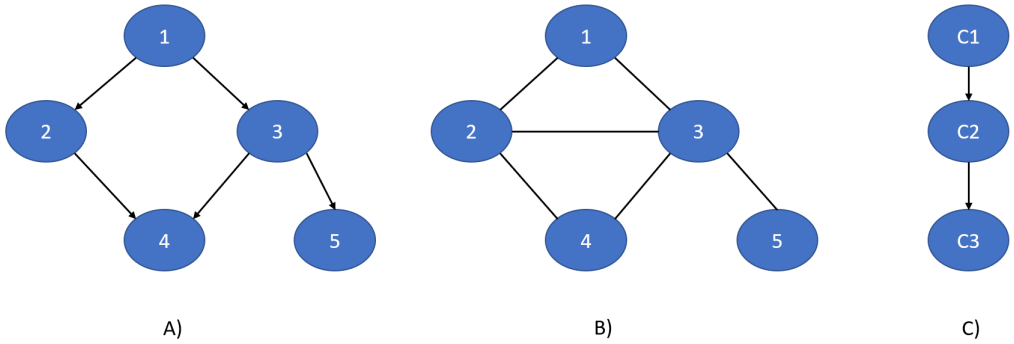


Figura 3.6: Transformación de una red: A) Red Original, B) Red Moralizada, C) Árbol de Uniones.

Siguiendo los pasos en el orden recomendado se transformará en un grafo el cual su propagación es mediante el envío de mensajes en el árbol de uniones o *cliques*. Dicho grafo calcula la probabilidad conjunta de cada *clique* y la probabilidad condicional dado el padre, posteriormente se recalcularán las probabilidades de cada *clique*, bajo cierta evidencia, la probabilidad individual de cada variable se obtiene mediante el producto del *clique* por la marginalización. Esto hace que el algoritmo se complique a cada cálculo de las probabilidades. Lo que puede llevarnos a que el problema sea **NP-duro**[3].

Aprendizaje de Clasificadores Bayesianos

Como vimos en el capítulo anterior existen muchos tipos de clasificadores y en especial existe uno que tiene una relación estrecha con las **redes Bayesianas** y nos referimos al algoritmo **Naive Bayes o simple**, así es llamado en la literatura. Esto nos motiva a construir una red conforme a este algoritmo de clasificación.

[3]Hace referencia a la clasificación de problemas computacionales, tiempo polinómico y su dificultad aumenta con cada iteración.

Aunque el clasificador bayesiano simple o Naive Bayes funciona muy bien en muchos dominios, en ocasiones su rendimiento decrece debido a que los atributos no son condicionalmente independientes como se asume. Aquí partimos de la idea de que se tienen atributos dependientes, una forma de considerar estas dependencias es extendiendo la estructura básica y agregando arcos entre dichos atributos. Existen dos formas de hacerlo:

- **TAN**: Clasificador Bayesiano simple aumentado con un árbol.
- **BAN**: Clasificador Bayesiano simple aumentado con una red.

En ambos casos se agrega una estructura de dependencia entre los atributos:

- En el TAN se agrega una estructura de árbol entre los atributos, de forma que se tienen en principio *pocas* conexiones y no aumenta demasiado la complejidad de la estructura.
- En el BAN se agrega una estructura general de dependencias entre atributos, sin limitaciones.

Dichas estructuras (figura 3.7) se pueden aprender mediante los algoritmos de aprendizaje estructural.

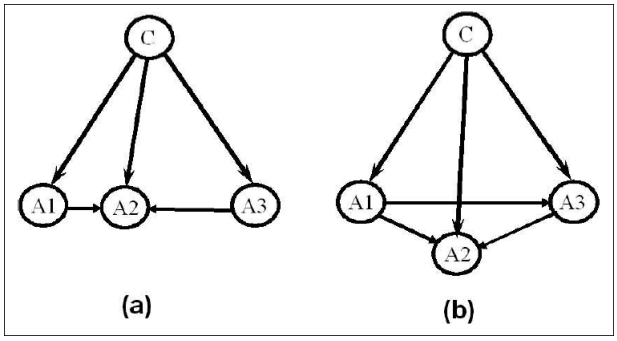


Figura 3.7: TAN y BAN.

Aprendizaje de Redes Bayesianas

El aprendizaje de manera general consiste en inducir un modelo, estructura y parámetros asociados, a partir de datos. Este puede dividirse en dos partes:

- **Aprendizaje estructural:** Obtener la estructura o topología de la red.
- **Aprendizaje paramétrico:** Dada la estructura, obtener las probabilidades asociadas.

Comenzaremos definiendo ambos tipos de aprendizaje. Pero antes de iniciar con los tipos de aprendizaje de las redes Bayesianas, vale la pena considerar los escenarios bajo los cuales se busca hacer aprender a una red.

1. Buscamos construir un modelo que nos permita responder *querys* generales (mismo objetivo que se eligió la red con conocimiento experto).
2. Buscamos predecir nuevas observaciones. Predecir variables objetivo y a partir de observaciones. Un ejemplo es en clasificación de imágenes o procesamiento de lenguaje.

3. No es de nuestro interés una tarea de inferencia particular, sino el descubrir conocimiento o estructura. Distinguir entre dependencias directas e indirectas y posibles direccionalidades de los arcos.

Sin embargo mencionamos que los objetivos anteriores se pueden satisfacer usando otras técnicas. Algunas de las razones o situaciones por las que se utilizan modelos gráficos son:

- Se busca predicción de objetos estructurados (explorar las correlaciones sobre varias variables).
- Se desea incorporar conocimiento experto al modelo.
- Tenemos un modelo unificado para múltiples variables.
- Es un marco para descubrir conocimiento.

Por ello para definir **aprendizaje de estructura** será necesario definir dos tipos generales de algortimos:

a) **Aprendizaje basado en restricciones:** Son algoritmos basados en pruebas de hipótesis de independencia entre variables. En este caso, el algoritmo se enfoca en las relaciones de independencia y dependencia, darles una explicación práctica al modelo.

b) **Aprendizaje basado en scores:** Son algoritmos que consideran las posibles estructuras gráficas como distintos modelos, de tal manera que el problema se convierte en uno de maximizar algún *Score*, el cual califica los distintos modelos. Definimos primero score (G, p), donde G es una gráfica y p es una distracción de probabilidad conjunto que se factoriza sobre G, e intentamos resolver (o aproximar) el problema.

$$\max_{G,p} scores(G, p). \tag{3.16}$$

El objetivo de esta tesis será enfocarnos más en el aprendizaje basado en *Scores*. Para esto tenemos que definir una función apropiada de *score*, y una manera de aproximar la solución del problema de maximización. Nuestro enfoque será heurístico, pues el problema de encontrar una solución extra (máximo global) rápidamente se vuelve en un problema intratable desde el punto de vista de tiempo y costo computacional. Por lo que si tenemos k variables, y consideramos un solo ordenamiento $X_1, \ldots, X_k$, por lo tanto hay un total de $2^1(2^2)\ldots(2^{k-1}) = 2^{\frac{k(k-1)}{2}}$ redes distintas que satisfacen el ordenamiento.

Aprendizaje Paramétrico

Cuando se tienen datos completos y suficientes para todas las variables en el modelo, es relativamente fácil obtener los parámetros, suponiendo que la estructura está dada. El método más común es el llamado **estimador de Máxima Verosimilitud**, bajo el cual se estiman las probabilidades en base a la frecuencia de los datos. Para una red Bayesiana se tienen los casos:

- **Nodos raíz:** Se estima la probabilidad marginal. Por ejemplo: $\mathbb{P}(A_i) \sim NA_i|N$, donde NA_i es el número de ocurrencias del valor i de la variable A_i y N el número total de registros.
- **Nodos Hoja:** Se estima la probabilidad condicional de la variable dados sus padres.

Dado que normalmente no se tienen suficientes datos, se tiene incertidumbre en las probabilidades estimadas. Esta incertidumbre se puede representar mediante una distribución de probabilidad, de forma que se considere en forma explícita la incertidumbre sobre las probabilidades. Para el caso de variables binarias se modela con una distribución Beta y para las variables multivariadas mediante su extensión que es la distribución Dirichlet. Esta representación puede utilizarse para modelar la incertidumbre cuando se tienen estimaciones de expertos, cambiando los valores de $a + b$, con el mismo valor estimado.

Aprendizaje Estructural

El aprendizaje estructural consiste en encontrar las relaciones de dependencia entre las variables, de forma que se pueda determinar la topología o estructura de la red Bayesiana. De acuerdo al tipo de estructura, podemos dividir los métodos de aprendizaje estructural en:

- Aprendizaje de árboles.
- Aprendizaje de poli árboles.
- Aprendizaje de redes multiconectadas.

Para el caso más general, que es el de redes multiconectadas, existen dos clases de métodos:

- Métodos basados en medidas y búsqueda.
- Métodos basados en relaciones de dependencia.

A continuación veremos el método para aprendizaje de árboles y sus extensión a poliárboles, para después ver los dos enfoques para aprender redes multiconectadas.

Aprendizaje de Árboles

El aprendizaje de árboles se basa en el algoritmo desarrollado por **Chow y Liu** para aproximar una distribución de probabilidad por un conducto de probabilidades de segundo orden (árbol). La probabilidad de n variables se expresa así:

$$\mathbb{P}(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_{j(i)}). \tag{3.17}$$

donde $X_{j(i)}$ es el padre de X_i. Para obtener el árbol se plantea el problema como uno de optimización: obtener la estructura de árbol que más se aproxime a la distribución *real*. Esto se basa en una medida de la diferencia de información entre la distribución real ($\mathbf{P}$) y la aproximada $\mathbf{P}*$:

$$\mathbf{DI}(\mathbb{P}, \mathbb{P}*)) = \sum_{X} \mathbb{P}(X) \log \left(\frac{\mathbb{P}(X)}{\mathbb{P} * (X)} \right). \tag{3.18}$$

El objetivo es minimizar **DI**. Se puede definir dicha diferencia en función de la información mutua entre pares de variables, que se define como

$$I(X_i, X_j) = \sum_{X_i X_j} \log \left(\frac{\mathbb{P}(X_i, X_j)}{\mathbb{P}(X_i)\mathbb{P}(X_j)} \right). \tag{3.19}$$

Se puede demostrar que la diferencia de información es una función del negativo de la suma de las informaciones mutuas (pesos) de todos los pares de variables que constituyen el árbol. Entonces, encontrar el árbol más próximo equivale a encontrar el árbol con mayor peso. Mediante el algoritmo de *maximum weight spanning tree:*

1. Calculando la información mutua entre todos los pares de variables (que para n variables, son $\frac{n(n-1)}{2}$).
2. Ordenar las informaciones mutuas de mayor a menor.
3. Seleccionar la rama de mayor valor como árbol inicial.
4. Agregar la siguiente rama mientras no forme un ciclo, si es así se desecha.
5. Repetir el paso 4 hasta que se cubran las variables ($n-1$ ramas).

El algoritmo no da direcciones para los nodos por lo que estos se pueden asignar de forma arbitraria o utilizando semántica externa (conocimiento experto).

Aprendizaje de Poli Árboles

Una forma de darle direcciones al 'esqueleto' aprendido con el algoritmo de **Chow y Liu**, es mediante pruebas de independencias no sólo entre dos variables, sino entre grupos de tres variables o tripletas. Mediante este esquema se genera un algoritmo de aprendizaje para poli árboles, ya que al asignar las direcciones puede ser que la estructura generada sea un árbol o un poli árbol (en realidad, un árbol es un caso especial de poli árbol). El algoritmo parte del esqueleto (estructura sin direcciones) obtenido con el algoritmo de **Chow y Liu**. Después se determinan las direcciones de los arcos utilizando pruebas de dependencia entre tripletas de variables. Dadas tres variables, existen tres casos posibles:

- Arcos secuenciales: $X \rightarrow Y \rightarrow Z$.
- Arcos divergentes: $X \leftarrow Y \rightarrow Z$.
- Arcos convergentes: $X \rightarrow Y \leftarrow Z$.

Los primeros dos casos son indistinguibles basado en las pruebas de independencias; es decir, son equivalentes. En ambos, X y Z son independientes dado Y. Pero el tercero es diferente, ya que las variables X y Z son marginalmente independientes. Este tercer caso lo podemos usar para determinar entonces las direcciones de los dos arcos que unen estas tres variables, y a partir de éstos, es posible encontrar las direcciones de otros arcos utilizando pruebas de independencia. De acuerdo a lo anterior, se establece el siguiente algoritmo para aprendizaje de poli árboles:

1. Obtener el esqueleto utilizando el algoritmo de **Chow y Liu**.
2. Recorrer la red hasta encontrar una tripleta de nodos que sean convergentes, donde la variable a la que apuntan los arcos la llamaremos nodo multi padre.
3. A partir de un nodo múlti padre, determinar las direcciones de otros arcos utilizando la prueba de dependencia de tripletas, hasta donde sea posible (base causal).
4. Repetir los pasos 2-3 hasta que ya no se puedan descubrir más direcciones.
5. Si quedan arcos sin direccionar, utilizar semántica externa para obtener su dirección.

Capítulo 4

Redes Bayesianas en Diagnóstico Médico

La medicina es una ciencia que tiene como objeto de estudio a la vida, la salud, las enfermedades y la muerte del ser humano. El campo de estudio es muy amplio pues las afecciones humanas pueden tener muchos orígenes, así como también consecuencias. Es por ello que la medicina actual tiene que apoyarse en el conocimiento de otras ciencias, entre ellas y las que nos causan mayor interés, es el campo de las ciencias exactas, ya que ellas ofrecen un punto de vista diferente a lo que otras ciencias pueden dar, es por ello que para este enfoque nace un tipo diferente de medicina, una complementaria al punto de vista y enfoque clásico de la medicina, ya que en esta medicina se trabajará con la incertidumbre que puede generar alguna de las variables que se modelarán.
Definimos a la **Medicina basada en evidencia (MBE)** como *el uso conciente, explícito y juicioso de la mejor evidencia científica disponible para tomar decisiones sobre los pacientes*. Estas palabras son del profesor **David Sackett** [Moreno Rodríguez, 2005]. En su propósito por obtener la mejor evidencia posible, el ejercicio de la *MBE* se ha fundamentado en el método científico, al que considera el mejor instrumento para comprender la realidad y expresarla de manera sistemática e inteligible.

El Diagnóstico Médico

El enfoque de esta tesis es principalmente hacia las enfermedades cardíacas (estado de la enfermedad). Nos basamos en los fundamentos de la *MBE (Medicina basada en evidencias)*, cuyo enfoque busca optimizar la toma de decisiones, haciendo énfasis en el uso de pruebas científicas. La *MBE* utiliza la clasificación de las pruebas científicas observadas y consensuadas, las cuales deben cumplir que sean firmemente establecidas, con esto nos referimos a que las pruebas tienen que tener algún fundamento que sustente dichas pruebas para que ellas puedan generar una recomendación médica. Como bien dijimos las MBE se sustentan en argumentos científicos por lo que la experiencia médica es integración necesaria para tomar la mejor decisión terapéutica, tomando en cuenta los valores y las preferencias de los pacientes, con esto en mente hemos mencionado los tres pilares de la medicina basada en evidencias que son: **evidencias, experiencias, valores y preferencias**.

Toma de Decisiones

El tema de las decisiones clínicas es muy complejo, es un proceso multidisciplinario y marcado de manera ineludible por la incertidumbre. Usualmente los médicos hacen el cálculo de los riesgos basado en su juicio particular, así como en sus experiencias médicas y conocimientos sobre el tema, el conjunto de estos elementos es lo que permite al médico recomendar el mejor tratamiento para

cada paciente en particular. Es por ello que nos vemos en la necesidad de crear un nuevo paradigma que sea capaz de incorporar la incertidumbre en sus procesos, para ello es que recurrimos a las matemáticas específicamente al campo de la **estadística y probabilidad**, las cuales trabajan con la incertidumbre, ocupamos dichas ramas de las matemáticas como sustento teórico para las redes Bayesianas, un sistema experto el cual nos ayuda a encontrar el diagnóstico óptimo para los pacientes basándonos en su **información médica como paciente** y así dar mayor certidumbre al diagnóstico.

Conocimiento Experto

Definiremos el conocimiento experto o también llamado **conocimiento de la materia**, como el conocimiento excelente de un tema, lo podemos interpretar como el camino de la experiencia, este conocimiento es difícil de plasmar como datos, su valor dependerá de la ponderación que le de el analista, particularmente en el área de la salud es el conocimiento experto es de suma relevancia, ya que son considerados mucho más veraces los diagnósticos de una medico veterano a un médico novato.
Es común que algún desorden en alguna característica médica de nuestro paciente puede desencadenar una serie de padecimientos, ya sean en el corazón como en otros órganos del paciente, esto variará según el estado del órgano, así como también dependerá del estado del paciente en sus otras variables. Con esto establecemos una cierta relación entre variables, ya que algunas son independientes completamente de otras como lo es la prueba de *fluoroscopia* con el *sexo*, la relación en este ejemplo es muy baja o nula. Una vez estableciendo esto podemos describir a manera de explicación las variables para generar el modelo más óptimo basándonos en conocimiento experto.
El término *enfermedad del corazón* engloba varios términos, entre ellos debemos tener en cuenta dos clases **la cardiopatía y la enfermedad cardiovascular**, esta clasificación ayuda al médico a interpretar de manera más óptima un diagnóstico. El propio término de *enfermedad cardíaca* aplica a varias enfermedades que como tal pueden generar algún síntoma ya que afectan no solo al corazón sino también al sistema circulatorio y los vasos sanguíneos.

Cardiopatía y Enfermedad Cardiovascular

El término como tal de *enfermedad cardiovascular* incluye una amplia gama de afecciones, las cuales afectan como actor protagónico al corazón y los vasos sanguíneos, por lo que la circulación de la sangre se ve afectada. ,tenemos varias afecciones al corazón de las cuales se desprenden ciertas dependencias o relaciones, diremos algunas para ilustrar su importancia.
El estrechamiento de las arterias coronarias da como resultado la reducción del suministro de sangre y oxígeno al corazón lo que nos conduce a afecciones coronarias. Los infartos de miocardio por otra parte pueden tener como síntomas el dolor en el pecho y las anginas que se formen en el corazón, muchas veces estas dos son confundidas o relacionadas inmediatamente con esta enfermedad del corazón. Un bloqueo repentino de una arteria coronaria, generalmente se debe a un coágulo de sangre, el coagulo puede tener muchas razones entre ellas la mala alimentación o índices de glucosa y colesterol altos. La hipertensión arterial, la enfermedad arterial coronaria, la enfermedad cardíaca valvular, el accidente cerebrovascular o la fiebre reumática también conocida como enfermedad cardíaca reumática, son las diversas formas de enfermedad cardiovascular. Pero como bien hemos dicho muchas de estas enfermedades tienen síntomas que el paciente puede tomar en cuenta para tomar la decisión de ir con un especialista o médico general:

- Mareos continuos o desmayos.

- Molestias después de las comidas, especialmente si se prolongan más de lo esperado.
- Poco consumo de alimentos, después de un ligero esfuerzo.
- Dolor u opresión en el tórax, un signo común de insuficiencia coronaria.
- Sentimiento de adormecimiento o dolor muy intenso en los brazos.
- Palpitaciones fuera de lo común.

El diagnóstico inicial de un ataque cardíaco se realiza mediante una combinación de síntomas clínicos y cambios característicos del electrocardiograma. Con la siguiente informacion podemos obtener una serie de factores de riesgo para un ataque al corazón:

- *Alta presión sanguínea.*
- *Diabetes.*
- *Alto colesterol.*
- *Historial familiar de ataques cardíacos a edades menores de 60 años, uno o más ataques cardíacos anteriores, sexo masculino.*
- *Obesidad.*
- *Las mujeres posmenopáusicas tienen un mayor riesgo que las mujeres pre menopáusicas. Se cree que esto se debe a la pérdida de los efectos protectores de la hormona estrógeno en la menopausia.*

Como hemos dicho el conocimiento experto se construye mediante tiempo y experiencia en este caso del médico, lo que el modelado de la información basado en algoritmos de *machine learning* ofrece al médico es modelar la incertidumbre, ofrecer una perspectiva computacional basada en estadística para ofrecer al tomador de decisión un camino óptimo.

Hipótesis

Supongamos que tenemos de manera particular un paciente que presenta un cuadro de *dolor en pecho*, el cual es un síntoma que nos alarma que algo puede no estar bien con nuestro corazón, claro no es el único factor que consideraremos en el modelo pero en primera instancia es uno que llama la atención fuertemente.
Como bien sabemos la *medicina basada en pruebas* tiene una fuerte base en el conocimiento experto. Aquí tendremos dos aspectos a considerar:

- **Considerando el conocimiento experto.**
- **Sin considerar el conocimiento experto.**

Ambas posturas tienen como objetivo ver la eficacia de utilizar las técnicas de *clasificación* de la información para darle un objetivo claro, que en este caso es la predicción de una enfermedad cardíaca.
Por lo que establecemos nuestras hipótesis para el análisis de datos, los cuales iremos explicando poco a poco a medida que avancemos tanto en los métodos como en los algoritmos que utilizamos. Por lo mientras diremos que nuestra hipótesis inicial es: *¿El diagnóstico mantiene una correlación directa o indirecta con todas las variables? y ¿cuáles variables son más significativas para el diagnóstico?* Con esto partimos al análisis de la *base de datos*.

Estudiando la Información

La base de datos fue obtenida de un repositorio web **UCI Machine Learning Repository**, el cual proporciona bases de datos para su análisis. Ya que el objetivo de la tesis es hablar sobre el potencial de las redes Bayesianas para el análisis de datos que contengan una gran cantidad de incertidumbre, elegimos la base de datos para el diagnóstico de enfermedades del corazón *(heart disease)*, de este repositorio escogimos la base de datos *"processed.cleveland"*, escogimos esta base de datos ya que nos facilita el trabajar con bases de datos que tengan el menor número de *"datos perdidos"* o mayormente conocido en la literatura como *"missing data"*. La base de datos contiene a una serie de pruebas que se realizaron a distintos pacientes, en total fueron 76 atributos que tomaron en cuenta los médicos para evaluar a los pacientes, de los cuales 14 de ellos describen de mejor manera el cuadro clínico de los pacientes enfermos del corazón o con antecedentes de enfermedades de corazón que presentan síntomas.
Estos 14 atributos provienen del conocimiento experto, ya que a consideración de los médicos involucrados son pruebas que definen con mayor certeza una posible enfermedad del corazón, en especial las *anginas de pecho*, estos atributos son los siguientes:

1. **age**: Edad del paciente.
2. **sex**: Sexo del paciente.
3. **cp**: Dolor en el pecho.
4. **trestbps**: Presión arterial en reposo.
5. **chol**: Colesterol sérico.
6. **fbs**: Glucemia en ayunas.
7. **restecg**: Electrocardiograma en reposo.
8. **thalach**: Frecuencia cardíaca máxima.
9. **exang**: Angina inducida por ejercicio.
10. **oldspeak**: Depresión ST inducida por el ejercicio.
11. **slope**: La pendiente del segmento ST.
12. **ca**: Número de vasos principales.
13. **thal**: Resultado de la prueba de estrés de Talio.
14. **num**: Estado de la enfermedad del corazón

Estos atributos, en su mayoría, representan pruebas médicas o mediciones de rutina como lo es el ritmo cardiaco, el conocimiento experto de los medicos considera necesarias estas pruebas para diagnosticar el estado de la enfermedad del corazón, pues cada atributo tiene una ponderación necesaria para el análisis al cual someteremos estas pruebas. Posteriormente a estas pruebas las llamaremos variables para comodidad del lector, ya que nuestro objetivo es volver una entrada a cada variable, llevaremos las variables a terreno computacional. Como bien hemos dicho a estas variables se les consideró una ponderación para su análisis, el cual describimos en la tabla 4.1.

Base Datos		
Variable	Tipo de variable	Valor
Edad	int	—
Sexo	binaria	0 = mujer 1 = hombre
Dolor en el Pecho (DL)	categórica	DL = 1 Angina Típica DL = 2 Angina Atípica DL = 3 Dolor no Anginal DL = 4 Asintomático
Presión Arterial en Reposo (PA)	continua	—
Colesterol Sérico (CS)	continua	—
Glucemia en Ayunas	binaria	Glucosa = 1 Verdad Glucosa = 0 Falso
Electrocardiograma en Reposo (Electro)	categórica	Electro = 0 normal Electro = 1 Alteración de onda Electro = 2 Hipertrofia
Frecuencia Cardíaca Máxima (RCM)	continua	—
Angina Inducida (Angina)	binaria	Angina = 1 Tiene angina Angina = 2 No tiene Angina
Depresión ST Inducida por Ejercicio (DIE)	continua	—
Pendiente del Segmento ST (PE)	categórica	DIE = 1 Upsloping DIE = 2 Flat DIE = 3 Downsloping
Número de vasos principales (V)	int	—
Resultados de la prueba de Talio (thal)	categórica	thal = 3 normal thal = 6 defecto detectado thal = 7 defecto reversible
Estado de la enfermedad del Corazón (diagnóstico)	int	diagnóstico >= 1 50 % de diámetro

Tabla 4.1: Información de nuestra *base de datos*.

Descripción de las Variables

La base de datos se compone principalmente de características médicas de los pacientes, con esto nos referimos a los estudios que los médicos consideraron pertinentes para hallar la existencia de alguna enfermedad del corazón y si se encontrará su estado y evolución.
Para ello los médicos mandaron los siguientes estudios a los pacientes como son: **electrocardiograma, depresión ST inducida por ejercicio, prueba de estrés de Talio y número de vasos coloreado por la fluoroscopia**, estos estudios los podemos considerar una manera de probar el rendimiento del corazón bajo el estrés de una actividad física. Para estas pruebas necesitamos la consideración médica, el estado en el que se encuentra el cuerpo y por ello también en la base de datos tenemos una descripción de las características de su cuerpo; de manera particular para cada paciente se hacen las mediciones, por estado del cuerpo nos referimos a los siguientes atributos: **edad, sexo, presión arterial, colesterol sérico, la glucosa y frecuencia cardíaca máxima**.
Todo esto es relacionado a algún padecimiento ya que alguna característica muy común en las enfermedades del corazón es el desarrollo de algún padecimiento que nos permite detectarlo, en este caso tenemos variables dentro de la base de datos que denotan padecimientos: **dolor en el pecho y angina inducida por ejercicio** normalmente crónicos. Con la siguiente información el médico debe tomar una decisión basada en su experiencia y conocimientos para poder dar un buen diagnóstico de la enfermedad del corazón, esto maneja un gran índice de incertidumbre, es donde entra en acción la **MBE** y proponemos a los métodos para el análisis de esta información y así poder apoyar de la manera más óptima al médico para que tome la mejor decisión de diagnóstico médico.

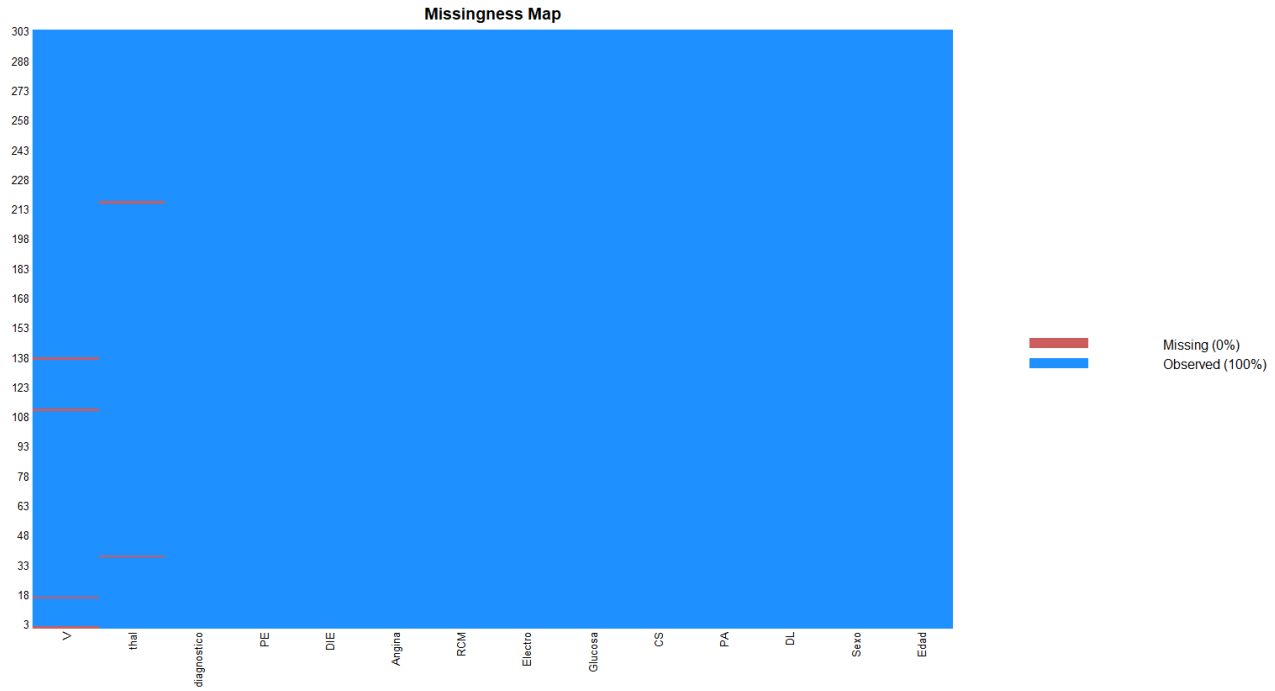


Figura 4.1: Missmap con Amelia.

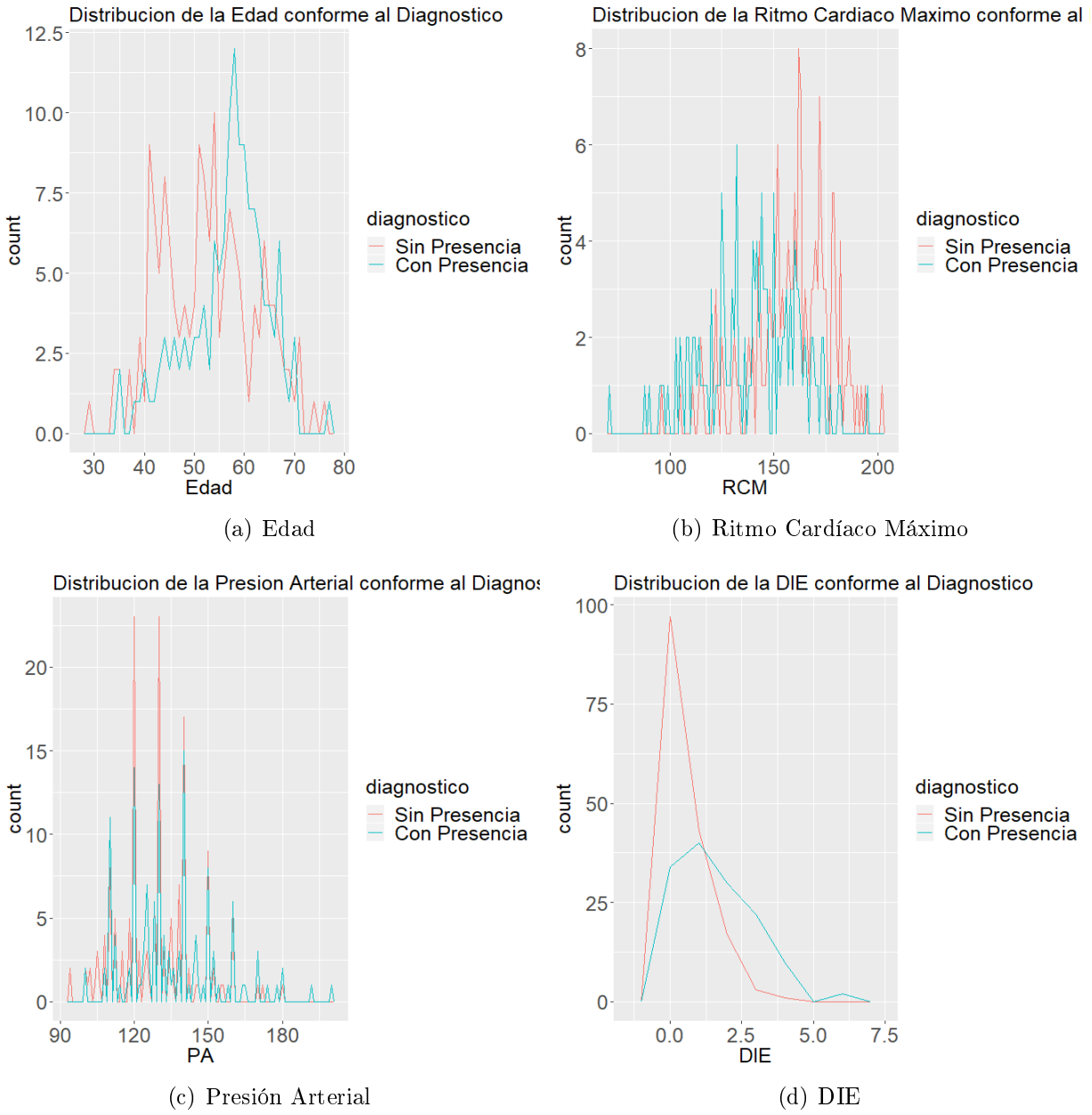
Análisis Exploratorio de la Información

Vamos a realizar una serie de visualizaciones para ver de mejor manera cada variable, esto nos ayudará a mejorar nuestro entendimiento de el significado de cada variable predictiva así con su interacción con la variable a predecir que en este caso particular es el *diagnóstico*.

Pérdida de Información o Valores Nulos

Debemos afrontar un hecho y es que existe una constante pérdida de información, este problema es muy habitual cuando tratamos con bases de datos ya que pueden surgir muchos problemas en la recolección de la información que van desde errores humanos hasta desastres naturales, comprobaremos aquí si es que existen valores nulos en nuestra *data*.
Después de un análisis exploratorio notamos que existen algunos valores registrados como símbolos[1] que no hace sentido a la información de nuestras variables *thal y V* (figura 4.1), así que procederemos a borrar el registro completo ya que no tenemos la certeza de cuál podría ser el valor y el completar los registros con valores nulos no entra en los objetivos de esta tesis, así que para fines prácticos vamos a eliminar el registro y procederemos al análisis.

[1] En la *data* encontramos el símbolo de interrogación (?).



(a) Edad

(b) Ritmo Cardíaco Máximo

(c) Presión Arterial

(d) DIE

Figura 4.2: Frecuencias Relativas.

Relación Entre Variables

La primera visualización que haremos será el análisis de frecuencias, como bien sabemos nuestra variable *diagnóstico* consta de 4 valores que va desde 0 al 4 siendo todos del tipo entero, pero según nuestra información esta variable representa el estado ***Angiográfico de la enfermedad*** por lo que tenemos dos valores:

- **Valor mayor al** $50\,\%$ **de diámetro de estrechamiento** = *Tiene presencia*
- **Valor mayor al** $50\,\%$ **de diámetro de estrechamiento** = *No tiene presencia*

Nos referiremos a *Tiene presencia* y *No tiene presencia* al estado de la enfermedad, en este caso la variable *diagnóstico* es de valores enteros por lo que lo resumimos a si *diagnstico* > 1, nuestro

valor será 1 y si *diagnstico* = 0 nuestro valor será 0. Así es como analizaremos la información de manera binaria.

Frecuencias

Ahora veamos cómo es que la información se distribuye según las frecuencias con respecto a la edad, en la (figura 4.2) se muestran algunas frecuencias de nuestra *data*.
Estas gráficas muestran diversas *evidencias* por ejemplo, que las personas con enfermedades cardíacas tienden a ser de edades mayores y de sexo masculino, tienen una presión arterial más alta, niveles más altos de colesterol y una frecuencia cardíaca máxima más baja medida mediante la prueba de estrés de Talio, todo esto comparado con las personas sin la enfermedad o en nuestro caso *sin presencia*. En la base de datos que estamos trabajando, la variable *edad* nos muestra un incremento en los pacientes con 60 años de edad, esta frecuencia nos indica que nuestra población mas propensa a desarrollar síntomas de enfermedad cardiaca será en su mayoría pacientes con este rango edad, con esto iniciamos el análisis de la base de datos.
Esto justo hace sentido al *conocimiento experto* del que hablamos con anterioridad, Ahora que lo vimos de manera gráfica y antes de pasar a los algoritmos debemos de usar herramientas estadísticas ya que necesitamos una manera de medir la discrepancia entre las variables, para ello ocuparemos la correlación estadística.

Variable	Correlación
Edad	0.22215589
Sexo	0.226797382
Dolor en Pecho	0.40424785
Presión Arterial	0.15962045
Colesterol Sérico	$6.644816e^{-02}$
Glucosa	0.0490398119
Electro	0.1841363
Ritmo Cardiaco Máximo	$4.206388e^{-01}$
Angina	0.3916131346
DIE	0.501460902
PE	0.37468927
thal	0.51337703
Número de Vasos	0.52117844

Tabla 4.2: Tabla de Correlaciones con *"diagnostico"*.

Test de Correlación

Mediante el software estadístico R, calcularemos la matriz de correlación de nuestras variables con respecto a nuestra variable *diagnóstico*, las cuales mostramos en la tabla 4.2.
En la tabla sintetizamos las correlaciones que queremos visualizar que es justamente la interacción que tiene la variable *diagnóstico* con las demás variables, aunque existe una manera menos simplificada llamada la matriz de correlación.

```
##                    Edad         Sexo           DL          PA
## Edad         1.00000000 -0.092399479  0.110470866  0.29047626
```

```
## Sexo        -0.09239948  1.000000000  0.008908026 -0.06634020
## DL           0.11047087  0.008908026  1.000000000 -0.03697975
## PA           0.29047626 -0.066340200 -0.036979748  1.00000000
## CS           0.20264355 -0.198089063  0.072088310  0.13153571
## Glucosa      0.13206199  0.038850300 -0.057663110  0.18085954
## Electro      0.14991651  0.033896828  0.063904695  0.14924228
## RCM         -0.39456288 -0.060496006 -0.339307624 -0.04910766
## Angina       0.09648880  0.143581250  0.377524789  0.06669107
## DIE          0.19712262  0.106567243  0.203243824  0.19124314
## PE           0.15940474  0.033344964  0.151078594  0.12117205
## V            0.36221034  0.091924800  0.235644123  0.09795376
## thal         0.12658600  0.383651748  0.268499548  0.13818322
## diagnostico  0.22215589  0.226797382  0.404247850  0.15962045
##                        CS       Glucosa     Electro           RCM
## Edad         2.026435e-01  0.1320619890  0.14991651 -3.945629e-01
## Sexo        -1.980891e-01  0.0388502996  0.03389683 -6.049601e-02
## DL           7.208831e-02 -0.0576631095  0.06390469 -3.393076e-01
## PA           1.315357e-01  0.1808595428  0.14924228 -4.910766e-02
## CS           1.000000e+00  0.0127082808  0.16504603 -7.456799e-05
## Glucosa      1.270828e-02  1.0000000000  0.06883111 -7.842359e-03
## Electro      1.650460e-01  0.0688311070  1.00000000 -7.228965e-02
## RCM         -7.456799e-05 -0.0078423590 -0.07228965  1.000000e+00
## Angina       5.933893e-02 -0.0008930821  0.08187392 -3.843675e-01
## DIE          3.859579e-02  0.0083106671  0.11372642 -3.476400e-01
## PE          -9.215240e-03  0.0478190123  0.13514058 -3.893067e-01
## V            1.159446e-01  0.1520858900  0.12902063 -2.687270e-01
## thal         1.085909e-02  0.0622090124  0.01879550 -2.748310e-01
## diagnostico  6.644816e-02  0.0490398119  0.18413630 -4.206388e-01
##                     Angina          DIE          PE           V        thal
## Edad         0.0964888046  0.197122616  0.15940474  0.36221034  0.12658600
## Sexo         0.1435812504  0.106567243  0.03334496  0.09192480  0.38365175
## DL           0.3775247891  0.203243824  0.15107859  0.23564412  0.26849955
## PA           0.0666910687  0.191243136  0.12117205  0.09795376  0.13818322
## CS           0.0593389323  0.038595794 -0.00921524  0.11594459  0.01085909
## Glucosa     -0.0008930821  0.008310667  0.04781901  0.15208589  0.06220901
## Electro      0.0818739197  0.113726420  0.13514058  0.12902063  0.01879550
## RCM         -0.3843675321 -0.347639972 -0.38930674 -0.26872698 -0.27483099
## Angina       1.0000000000  0.289309666  0.25057152  0.14823223  0.32692680
## DIE          0.2893096659  1.000000000  0.57903735  0.29445228  0.34497594
## PE           0.2505715154  0.579037353  1.00000000  0.10976112  0.27968774
## V            0.1482322256  0.294452277  0.10976112  1.00000000  0.25638250
## thal         0.3269268035  0.344975944  0.27968774  0.25638250  1.00000000
## diagnostico  0.3916131346  0.501460902  0.37468927  0.52117844  0.51337703
##              diagnostico
## Edad          0.22215589
## Sexo          0.22679738
## DL            0.40424785
## PA            0.15962045
```

```
## CS           0.06644816
## Glucosa      0.04903981
## Electro      0.18413630
## RCM         -0.42063880
## Angina       0.39161313
## DIE          0.50146090
## PE           0.37468927
## V            0.52117844
## thal         0.51337703
## diagnostico  1.00000000
```

Juntando la información proporcionada por las gráficas y por las correlaciones es que podemos establecer relaciones entre las variables, muchas de ellas muy evidentes como que la *Edad* tiene una relación con *el número de vasos* y el *diagnóstico*, como bien sabemos por el conocimiento experto, la edad es una fuerte influencia para los males del corazón. Pero también encontramos interesantes relaciones como son *Ritmo Cardíaco Máximo* no es tan significativo como pensamos, ya que al menos tiene índices muy bajos de correlación, por lo que nos lleva a descartar que *RCM* sea una variable que influya en nuestro modelo, ya sea que por conveniencia del algoritmo se elimine o pase a una unión con otra variable para ahorrar cálculos. Pero eso lo explicaremos en los algoritmos.

Implementación de los Algoritmos

Planteamos en la hipótesis de la tesis nuestra idea para esta *data* y como creemos que ciertas variables responden bajo ciertas circunstancias para provocar una afección al corazón, es por lo que en esta sección explicaremos cada uno de los algoritmos que utilizamos, esto con la finalidad de dotar a la tesis de una mayor explicación a la hora de programar el *script en R* y así poder establecer más claramente una conclusión.
El proceso de evaluación de las redes Bayesianas a partir de los datos médicos descritos anteriormente se llevó a cabo a través de la ejecución de algoritmos de clasificación, cuya precisión y rapidez de procesamiento son los más utilizados en este tipo de problemas, previamente definimos algunos por lo que nos tomaremos ciertas libertades.

Aprendizaje de las Redes Bayesianas

Conocemos como **aprendizaje** a la selección y ajuste de los modelos que tenemos; por lo general tenemos dos pasos para este proceso:

- Paso 1: **Aprendizaje estructural**, el cual consiste en el aprendizaje de la estructura gráfica a partir de la DATA.
- Paso 2: **Aprendizaje paramétrico**, conforme a la estructura del paso anterior, el aprendizaje empezará con las distribuciones locales basadas en la estructura.

Como bien sabemos el objetivo de la tesis es en el sentido Bayesiano, nuestra enfoque de trabajo será Bayesiano. Establecemos que X con θ será el conjunto de parámetros de las distribuciones globales, con estos elementos construiremos las redes Bayesianas que ocuparemos.

Red Bayesiana con Naive Bayes

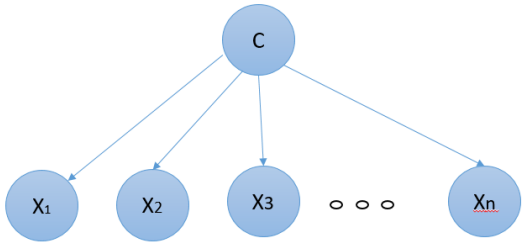


Figura 4.3: Red Bayesiana Naive Bayes.

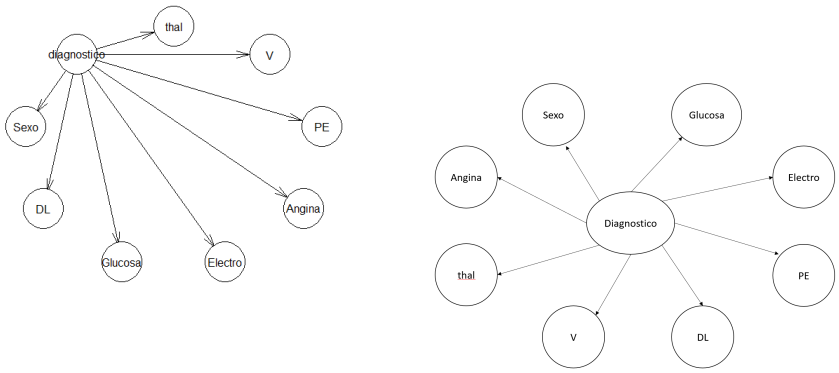
Implementación de Naive Bayes

Para la creación de las *redes Bayesianas Naive Bayes* (figura 4.3) y *TAN* recurriremos a las librerías de *bnlearn y Rgraphviz*, son dos librerías [Scutari and Ness, 2012] las cuales nos ayudarán a graficar y crear la debida inferencia, aprendizaje y cálculos que requeriremos para crear las redes.

```
library(bnlearn)
library(Rgraphviz)

part_bn <- Pacientes_Entrenamiento[,c(2,3,6,7,9,11,12,13,14)]
red_bayes <- naive.bayes(part_bn,"diagnostico")
red_bayes
graphviz.plot(red_bayes, layout = "fdp")
```

```
##
##   Bayesian network Classifier
##
##   model:
##    [diagnostico][Sexo|diagnostico][DL|diagnostico][Glucosa|diagnostico]
##    [Electro|diagnostico][Angina|diagnostico][PE|diagnostico][V|diagnostico]
##    [thal|diagnostico]
##   nodes:                                 9
##   arcs:                                  8
##     undirected arcs:                     0
##     directed arcs:                       8
##   average markov blanket size:           1.78
##   average neighbourhood size:            1.78
##   average branching factor:              0.89
##
##   learning algorithm:                    Naive Bayes Classifier
##   training node:                         diagnostico
##   tests used in the learning procedure:  0
```



(a) Estándar (b) Forma Estrella

Figura 4.4: Red Bayesiana Naive Bayes.

Ahora una vez que generamos la red *Naive Bayes* (figura 4.4[2]), vamos a ajustarla lo más posible para que su precisión sea mucho mayor.

```
fit_red <- bn.fit(red_bayes, part_bn)
```

```
bn.fit.barchart(fit_red$diagnostico)$
```

Ajustamos la red creada anteriormente, ahora con la (figura 4.5) podemos ver cómo se distribuye la clasificación por niveles del diagnóstico. Nos referimos a los valores de las probabilidades condicionales, en donde podemos observar una mayor distribución de los *pacientes sin presencia*, esto debido al ajuste necesario de las condiciones de nuestras variables que usamos para este modelo, que según lo anterior tienen una alta significancia en el modelo.
Ahora mediante la validación cruzada (la estándar de 10 elementos) vamos a comprobar su precisión a la hora de clasificar y de predecir, para el algoritmo de Naive Bayes, la validación cruzada arrojó lo siguiente:

```
cv_red = bn.cv(red_bayes, data = part_bn, loss = "pred-lw")
cv_red
```

```
##
##   k-fold cross-validation for Bayesian networks
##
##   target network structure:
##    [Naive Bayes Classifier]
##   number of folds:                        10
##   loss function:
##                                  Classification Error (Posterior, disc.)
##   training node:                          diagnostico
##   expected loss:                          0.1851852
```

[2]La forma de la red puede ser mostrada en forma de *Estrella* o *Estándar*, son formas de visualizar la información de la red, esto dependerá de quien analicé la información y cual forma lo ayude a ver las relaciones.

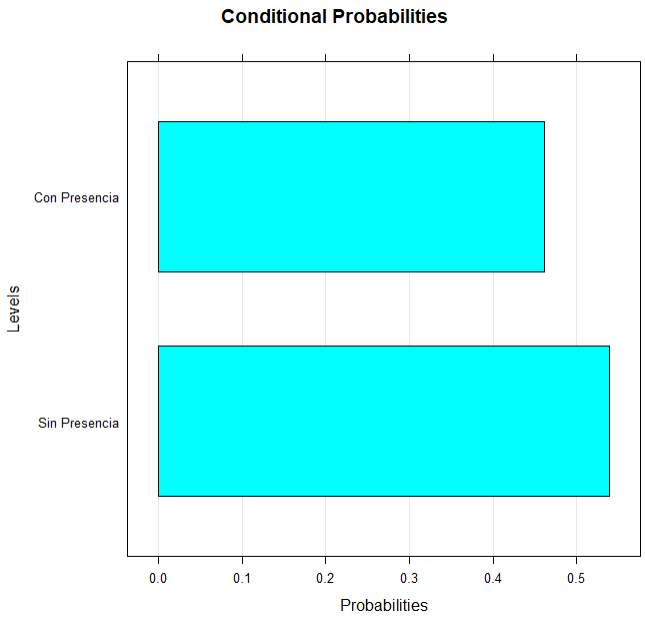


Figura 4.5: Probabilidad Condicional de Diagnóstico.

La validación cruzada es un algoritmo con el que nos apoyamos para probar de manera iterativa los resultados del ajuste a la red (no solo es aplicable a redes Bayesianas), utilizamos los elementos estándar para la prueba (10 elementos), por lo que R nos muestra con *expected loss* que se equivoca relativamente poco, por lo que podemos concluir que es un modelo estable, con posibles mejoras.

Red Bayesiana Naive Bayes Aumentado a Árbol TAN

Abreviado *TAN* por sus siglas en inglés *Tree Augmented Naive Bayes* (figura 4.6), esta estructura se obtiene al iniciar con una estructura de árbol convencional con las variables predictoras, para posteriormente conectar la variable clase con cada una de las variables predictoras.

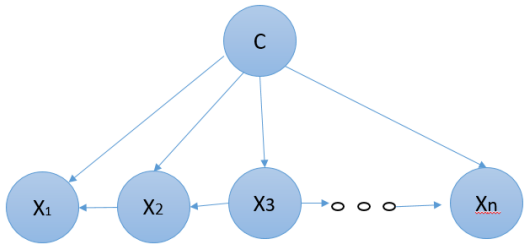


Figura 4.6: Red Bayesiana TAN.

Dicho algoritmo es una simplificación del trabajo de *Chow-Lin (1968)*, en dicho algoritmo se tiene en cuenta la cantidad de información mutua condicionada a la variable clase. La cantidad de información mutua entre las variables discretas X e Y condicionada a la variable C:

$$I(X,Y|C) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{r=1}^{w}\mathbb{P}(x_i,y_j,c_r)\log\left(\frac{\mathbb{P}(x_i,y_j|c_r)}{\mathbb{P}(x_i|c_r)\mathbb{P}(y_j|c_r)}\right). \tag{4.1}$$

El algoritmo se inicializa sencillo, para ello mostraremos un pseudocódigo y posterior como fue implementado en nuestra *data*.

```
Paso 1 Iniciar el modelo con Naive Bayes
Paso 2 Repetir en cada paso la mejor opcion entre
(a) Considerar remplazar dos de las variables usadas por el clasificador
por una nueva variable producto cartesiano entre ambas.
(b) Considerar eliminar una variable usada por el clasificador.
Evaluar cada posible opcion por medio de la estimacion del
porcentaje del clasificador.
Hasta que ninguna opcion produzca mejoras.
```

Después de la construcción de árbol, se calcula y almacena la probabilidad condicional de cada uno de los atributos condicionados en su etiqueta principal y de clase, además de la probabilidad condicional de la raíz. La variable condicionada en la clase se calcula y almacena, para posterior la probabilidad condicional de cada etiqueta de la clase $\mathbb{P}(C|X_1, \ldots X_n)$ se calculará como un producto de probabilidad. La etiqueta de clase con el valor de probabilidad posterior máximo, se asigna a la muestra de pruebas. La implementación de TAN utiliza las técnicas de corrección de *Laplace* al igual que *Naive Bayes* para redes, lo que deja la siguiente ecuación:

$$\mathbb{P}(C|X_1, \ldots, X_n) = \mathbb{P}(X_{raiz}|C) \prod_i \mathbb{P}(X_i|C, X_{parientes}). \tag{4.2}$$

Implementación de TAN

Para la creación de las *TAN* recurriremos a las librerías de *bnlearn y Rgraphviz*, son dos librerías [Scutari and Ness, 2012] las cuales nos ayudarán a graficar y crear la debida inferencia, aprendizaje y cálculos que requeriremos para crear las redes.

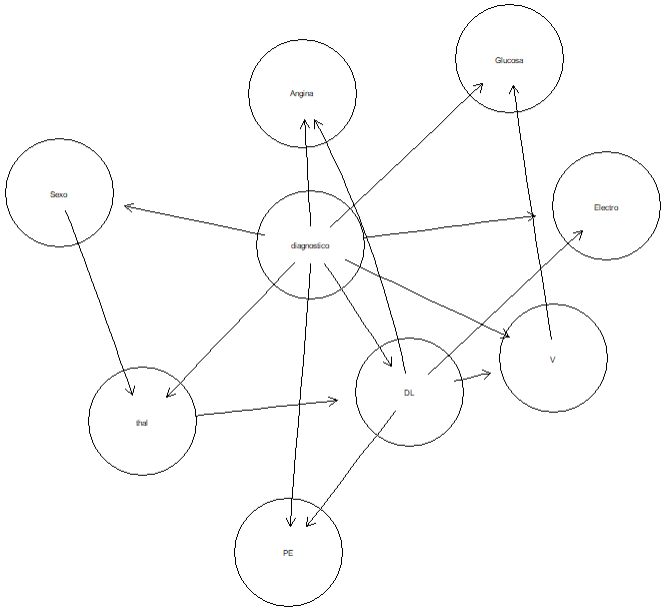
```
library(bnlearn)
library(Rgraphviz)
```

Nos enfocaremos en las variables con mayor correlación que previamente analizamos separándola en un conjunto aparte de nuestra *data*. Crearemos la red y analizaremos sus precisiones al momento de clasificar y predecir.

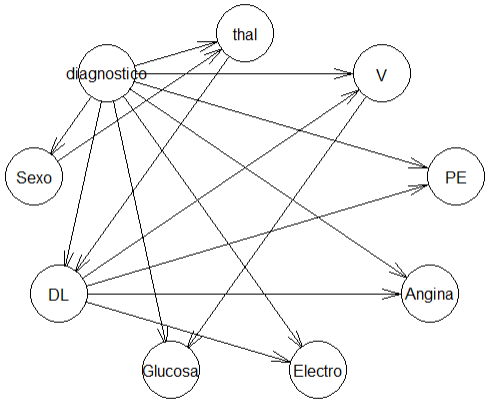
```
part_bn <- Pacientes_Entrenamiento[,c(2,3,6,7,9,11,12,13,14)]
red_TAN <- tree.bayes(part_bn, "diagnostico")
red_TAN
graphviz.plot(red_TAN, layout = "fdp")
```

```
##
##   Bayesian network Classifier
##
##   model:
##    [diagnostico][Sexo|diagnostico][thal|diagnostico:Sexo]
##    [DL|diagnostico:thal][Electro|diagnostico:DL][Angina|diagnostico:DL]
##    [PE|diagnostico:DL][V|diagnostico:DL][Glucosa|diagnostico:V]
##   nodes:                                 9
##   arcs:                                  15
##     undirected arcs:                     0
##     directed arcs:                       15
```

```
##   average markov blanket size:           3.33
##   average neighbourhood size:            3.33
##   average branching factor:              1.67
##
##   learning algorithm:                    TAN Bayes Classifier
##   mutual information estimator:          Maximum Likelihood (disc.)
##   training node:                         diagnostico
##   tests used in the learning procedure:  28
```



(a) Estándar



(b) Elaborada

Figura 4.7: Red Bayesiana TAN.

Ahora una vez que generamos la red *TAN* (figura 4.7), vamos a ajustarla lo más posible para que su precisión sea mucho mayor.

```
fit_redTAN = bn.fit(red_TAN,part_bn)
bn.fit.barchart(fit_redTAN$diagnostico)$
```

Con los anteriores comandos ajustamos nuestra red, pero queremos ver la manera en que las *probabilidades condicionales* se distribuyan según sea el caso de clasificación, como hicimos en el modelo de la *red Bayesiana con Naive Bayes*, elaboraremos una gráfica (figura 4.8) para ver cómo se distribuyen dichas probabilidades condicionales entre los niveles de clasificación del diagnóstico. Lo que podemos apreciar es una aumento en los casos de pacientes *sin presencia*.

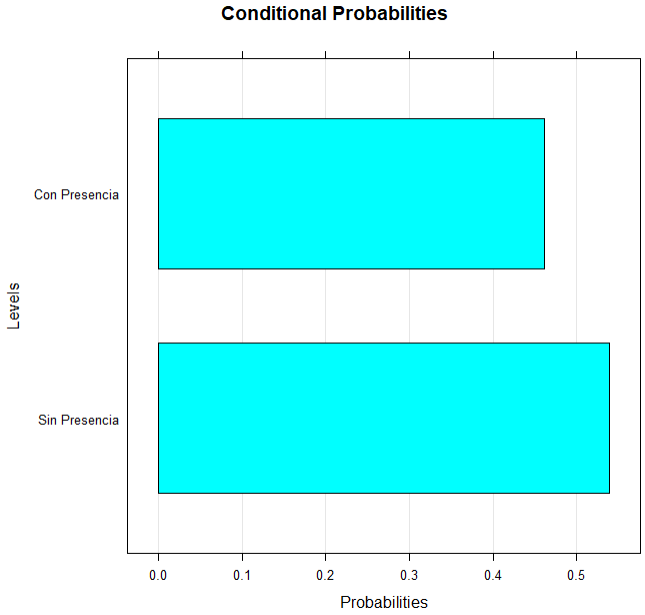


Figura 4.8: Probabilidad Condicional de diagnóstico.

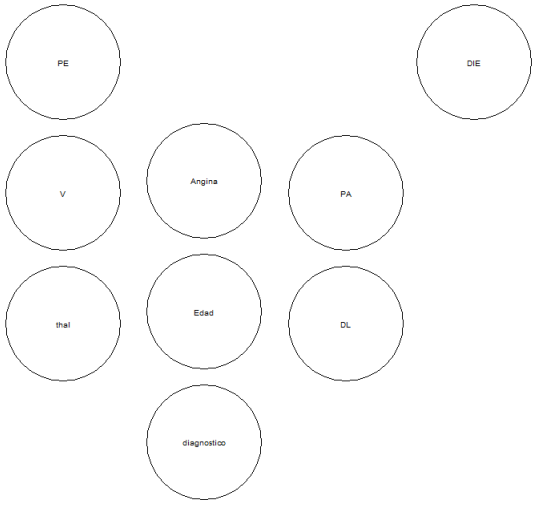
Ya ajustada nuestra red vamos a analizar con *validación cruzada* la red TAN, para saber la precisión con la que predice y clasifica, el resultado es el siguiente:

```
cv_TAN = bn.cv(red_TAN, data = part_bn, loss = "pred-lw")
cv_TAN
```

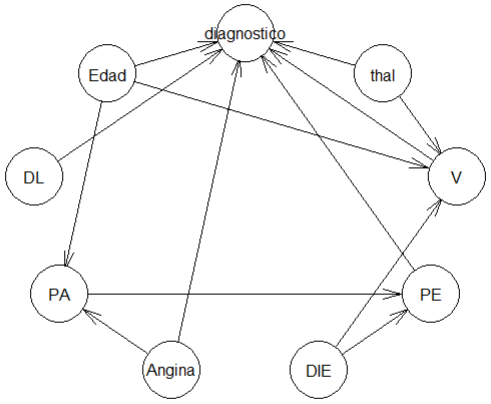
```
##
##   k-fold cross-validation for Bayesian networks
##
##   target network structure:
##    [diagnostico][Sexo|diagnostico][thal|diagnostico:Sexo]
##    [DL|diagnostico:thal][Electro|diagnostico:DL][Angina|diagnostico:DL]
##    [PE|diagnostico:DL][V|diagnostico:DL][Glucosa|diagnostico:V]
##   number of folds:                         10
##   loss function:
##                                  Classification Error (Posterior, disc.)
##   training node:                             diagnostico
##   expected loss:                             0.1885522
```

Lo que nos dice que falla al menos en clasificar a uno de los individuos. Tenemos que el error de clasificación es aceptable, basado en esto lo podemos considerar un buen modelo.

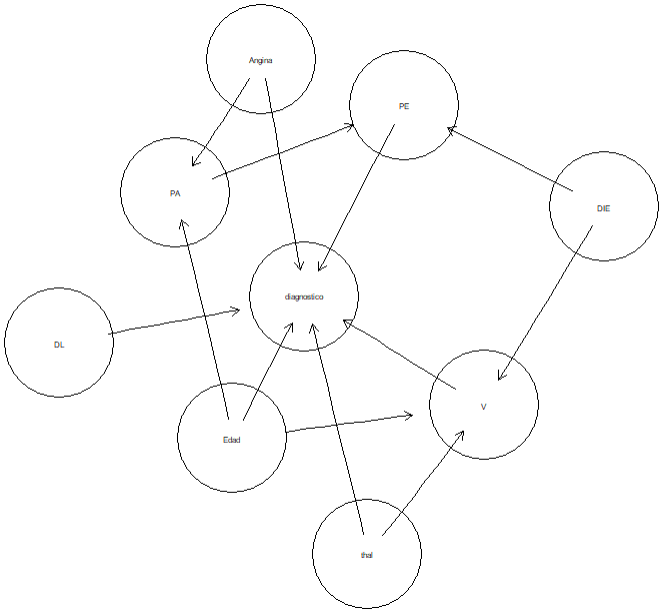
Red Bayesiana de Conocimiento Experto



(a) Sin Relaciones.



(b) Estándar.



(c) Elaborada.

Figura 4.9: Red Bayesiana Conocimiento Experto.

Como vimos anteriormente el conocimiento experto en las redes Bayesianas es muy importante pues mucha de la incertidumbre que puede contener el tema aplicando la red Bayesiana puede mitigarse si un experto colabora en la creación de la misma. Es por lo que uno de nuestros objetivos era crear una red Bayesiana con las cualidades que los expertos recomiendan y lo que tenemos es la red Bayesiana (figura 4.9):

- **Diagnóstico** La variable a predecir.

- **Presión Arterial.**
- **Dolor en el Pecho.**
- **Edad.**
- **thal.**
- **Número de Vasos.**
- **DIE.**
- **PE.**
- **Angina.**

La lista anterior nos muestra las variables que los médicos especialistas consideran relevantes en el seguimiento clínico de la enfermedad del corazón, lo anterior es un ejemplo de cómo sería una red Bayesiana y cuáles son sus relaciones en este caso de diagnóstico médico de enfermedades del corazón. Estas relaciones son echas y basadas en la mejor práctica para abordar los padecimientos clínicos según los médicos especialistas.

Árboles de Decisión

Se considera que el **Árbol de Decisión** es uno de los algoritmos de aprendizaje automático más útiles, ya que puede usarse para resolver una variedad de problemas. Aquí hay algunas razones por las que se usa el árbol de decisiones:

- Se considera que es el algoritmo de aprendizaje automático más comprensible y se puede interpretar fácilmente.
- Puede ser utilizado para problemas de clasificación y regresión.
- A diferencia de la mayoría de los algoritmos de aprendizaje automático, funciona de manera efectiva con datos no lineales.
- La construcción de un árbol de decisiones es un proceso muy rápido, ya que utiliza solo una función por nodo para dividir datos.

¿Qué es un Algoritmo de Árbol de Decisión?

Un árbol de decisión es un algoritmo de aprendizaje automático supervisado que se parece a un árbol invertido, donde cada nodo representa una variable predictiva (características), el enlace entre los nodos representa una decisión y cada nodo de hoja representa el resultado (variable respuesta). Un Árbol de decisión tiene la siguiente estructura:

- **Nodo Raíz:** El nodo raíz es el punto de inicio de un árbol. En este punto, se realiza la primera división.
- **Nodo Interno:** Cada nodo interno representa un punto de decisión (variable predictor) que eventualmente conduce a la predicción del resultado.

- **Hojas - Nodos Terminales:** Los nodos hoja representan la clase final del resultado y, por lo tanto, también se denominan nodos de terminación.
- **Ramas:** Las ramas son conexiones entre nodos, se representan como flechas. Cada rama representa una respuesta como *sí* o *no*.

Así que esa es la estructura básica de un árbol de decisión. Ahora tratemos de entender el flujo de trabajo de un árbol de decisión.

¿Cómo Funciona el Árbol de Decisión?

El algoritmo consta de los siguientes tres pasos:

Paso 1 Selecciona la entidad (variable predictiva) que mejor clasifica el conjunto de datos en las clases deseadas y asigne esa característica el nodo raíz.

Paso 2 Desplácese hacia abajo desde el nodo raíz, mientras toma decisiones en cada nodo interno, de modo que cada nodo interno clasifique mejor los datos.

Paso 3 Encamine de nuevo al *paso 1* y repita hasta que asigne una clase a los datos de entrada.

Los pasos mencionados anteriormente representan el flujo de trabajo general de un árbol de decisión utilizado para propósitos de clasificación. Ahora existen algoritmos para proceder con los árboles de decisión, aquí usaremos el **algoritmo ID3** el cual es uno de los más efectivos utilizados para construir un árbol de decisión. Utiliza el concepto de *Entropía* y *Ganancia* de información para generar un árbol de decisión para un conjunto dado de datos.

Entropía y Ganancia

Definimos la entropía y ganancia [Casas, 2012, página 55], que usualmente se usan para definir el mejor atributo, definiremos brevemente ambos. La **entropía** 1 mide la impureza o incertidumbre presente en los datos. Se utiliza para decidir cómo un árbol de decisión puede dividir los datos:

$$H(X) = E\{I(x)\} = \sum_{x} p(x) \log_2 p(x). \tag{4.3}$$

Se puede ver fácilmente $H(X) \geq 0$ esto corresponde al caso de menor incertidumbre. Por otro lado $H(X) \leq \log(|A|)$, donde $|A|$ es la cantidad de elementos de A, lo que corresponde al caso de mayor incertidumbre. La **ganancia** vista desde el punto de vista matemático es la ganancia de información, es la medida más significativa utilizada para construir un árbol de decisión. Indica la cantidad de *información* que una característica o variable particular nos da sobre el resultado final. La ganancia de información es importante porque solía elegir la variable que mejor divide los datos en cada nodo de un árbol de decisión. La variable con la *ganancia* más alta se usa para dividir los datos en el nodo raíz:

$$G(S, A) = E(S) - \sum_{v \in V(x)} \frac{|S_v|}{|S|} E(S_v). \tag{4.4}$$

Donde definimos a la función $G(S, A)$ como la ganancia, donde S es un conjunto de objetos y A los atributos, podríamos definirla mejor como: **La entropía del padre - el promedio ponderado por la entropía de los hijos**.
Después de esta breve introducción a los *árboles de decisión* aplicaremos estos conceptos a nuestra *data*.

Aplicando los Árboles de Decisión

Usaremos las siguientes librerías del software estadístico **R-project**.

```
library(caTools)
library(caret)
library(rpart.plot)
library(rpart)
```

Donde la librería de **caTools** nos servirá para crear 2 conjuntos de nuestra *data*, esos 2 conjuntos serán con el fin de *entrenamiento* y de *prueba*.

```
library(rpart)

set.seed(1200000)
divi = sample.split(Pacientes_Entrenamiento$diagnostico, SplitRatio = 0.63 )
test_arbol = subset(Pacientes_Entrenamiento, divi == FALSE)
training_arbol = subset(Pacientes_Entrenamiento, divi == TRUE)
training_arbol$V = as.numeric(training_arbol$V)
training_arbol[,c(1,4,5,8,10,12)] = scale(training_arbol[,c(1,4,5,8,10,12)])
test_arbol$V = as.numeric(test_arbol$V)
test_arbol[,c(1,4,5,8,10,12)] = scale(test_arbol[,c(1,4,5,8,10,12)])
arbol1 = rpart(formula = diagnostico ~.,
control = rpart.control(minsplit = 20, cp=0.01, maxdepth = 30), data = training_arbol)
arbol1$
```

Tenemos los dos conjuntos *test árbol* y *training árbol* mediante la división en un radio de 0.63. Ahora un detalle de los árboles de decisión, son malos bajo las variables tipo *factor*, es por eso que usamos lo que vimos en la matriz de correlaciones, aquellas que cumplen con las mejores variables para el árbol y las que elegimos fueron: **edad**, **presión arterial**, **colesterol**, **glucosa**, **frecuencia cardíaca máxima**, **DIE** y **número de vasos;** y esto fue lo que nos arrojó.

```
## n= 187
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 187 86 Sin Presencia (0.54010695 0.45989305)
##    2) DL=1,2,3 100 21 Sin Presencia (0.79000000 0.21000000)
##      4) thal=3,6 75  9 Sin Presencia (0.88000000 0.12000000)
##        8) Edad< 0.2100303 51  2 Sin Presencia (0.96078431 0.03921569) *
##        9) Edad>=0.2100303 24  7 Sin Presencia (0.70833333 0.29166667)
##         18) Edad>=0.5341761 17  2 Sin Presencia (0.88235294 0.11764706) *
##         19) Edad< 0.5341761 7  2 Con Presencia (0.28571429 0.71428571) *
##      5) thal=7 25 12 Sin Presencia (0.52000000 0.48000000)
##       10) RCM>=0.02917796 15  4 Sin Presencia (0.73333333 0.26666667) *
##       11) RCM< 0.02917796 10  2 Con Presencia (0.20000000 0.80000000) *
##    3) DL=4 87 22 Con Presencia (0.25287356 0.74712644)
##      6) V< -0.2059681 37 17 Sin Presencia (0.54054054 0.45945946)
##       12) thal=3,6 21  5 Sin Presencia (0.76190476 0.23809524)
```

```
##         24) Edad< 0.4261275 14  1 Sin Presencia (0.92857143 0.07142857) *
##         25) Edad>=0.4261275 7  3 Con Presencia (0.42857143 0.57142857) *
##       13) thal=7 16  4 Con Presencia (0.25000000 0.75000000) *
##      7) V>=-0.2059681 50  2 Con Presencia (0.04000000 0.96000000) *
```

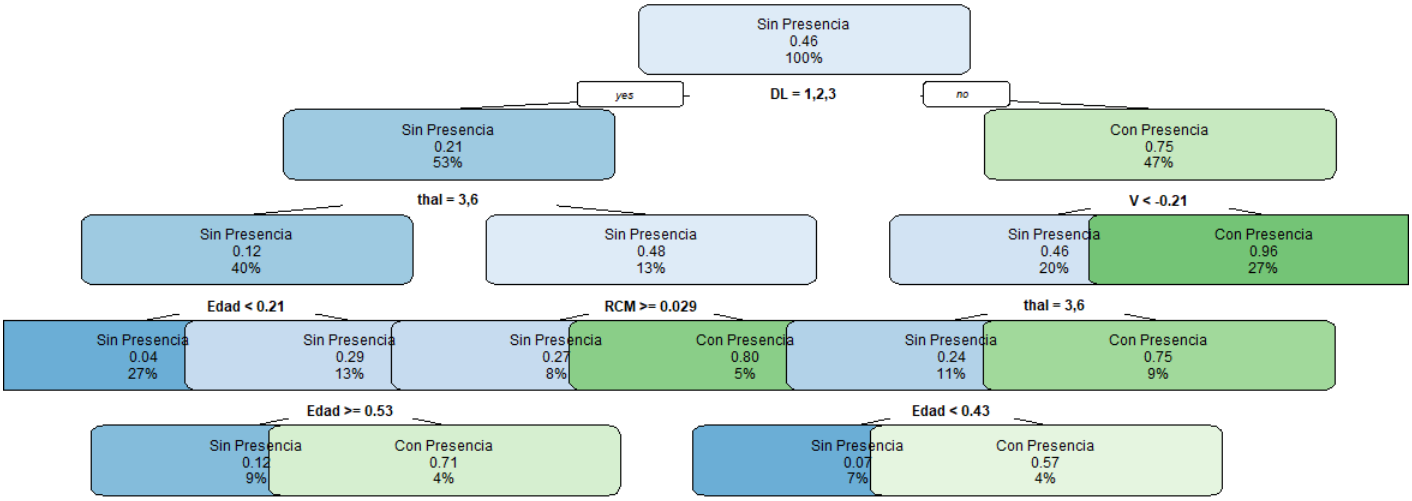


Figura 4.10: Árbol de Decisión.

Una vez generado el árbol, la librería **rpart.plot** [Therneau et al., 1997] nos ofrece la posibilidad de poder visualizarlo (figura 4.10), pero no hemos acabado aquí, tenemos que conocer su poder de predicción. Generaremos la *matriz de confusión*, la cual nos permite saber qué tan bien hemos clasificado.

```
predic_arbol =predict(arbol1, newdata = test_arbol[-14], type='class')
matriz_confucionarbol = table(test_arbol[,14], predic_arbol)
matriz_confucionarbol
```

```
##                 predic_arbol
##                  Sin Presencia Con Presencia
##   Sin Presencia             45            14
##   Con Presencia             11            40
```

Ahora mediante la librería ***caret*** [Kuhn et al., 2008] vamos a evaluar la matriz de confusión para saber el grado de exactitud al momento de clasificar evaluando nuestras variables.

```
conf_caretarbol <- confusionMatrix(predic_arbol,test_arbol[,14])
conf_caretarbol
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Sin Presencia Con Presencia
##   Sin Presencia            45            11
##   Con Presencia            14            40
##
##                Accuracy : 0.7727
##                  95% CI : (0.683, 0.8472)
##     No Information Rate : 0.5364
##     P-Value [Acc > NIR] : 2.465e-07
##
##                   Kappa : 0.5449
##  Mcnemar's Test P-Value : 0.6892
##
##             Sensitivity : 0.7627
##             Specificity : 0.7843
##          Pos Pred Value : 0.8036
##          Neg Pred Value : 0.7407
##              Prevalence : 0.5364
##          Detection Rate : 0.4091
##    Detection Prevalence : 0.5091
##       Balanced Accuracy : 0.7735
##
##        'Positive' Class : Sin Presencia
##
```

Naive Bayes

El algoritmo de Naive Bayes es de los más conocidos dentro de los algoritmos de clasificación y aprendizaje supervisado, en el capítulo anterior describimos brevemente su funcionamiento y su consistencia matemática aquí nos tocará describir su rendimiento en nuestra **data**. Este algoritmo se basa fuertemente en el ***Teorema de Bayes*** con el supuesto *ingenuo* de independencia condicional entre cada par de características dado el valor de la variable de clase. Bajo este supuesto ocupamos el siguiente resultado previamente mostrado en la sección 2:

$$\begin{aligned} \mathbb{P}(y|x_1,\cdots,x_n) \propto & \mathbb{P}(y)\prod_{i=1}^{n}\mathbb{P}(x_i|y). \\ \hat{y} = & \arg\max_y \mathbb{P}(y)\prod_{i=1}^{n}\mathbb{P}(x_i|y). \end{aligned} \tag{4.5}$$

Usamos la *estimación máxima a posteriori* o por sus siglas en inglés como (**MAP**) para estimar y, descrito brevemente para entender el clasificador *ingenuo*.
A pesar de sus supuestos aparentemente simplificados, los clasificadores ingenuos han funcionado bastante bien en la aplicación a problemas del mundo real, un ejemplo de éxito es el filtrado

de correos "no deseados". Para esto el algoritmo requiere una pequeña cantidad de datos de entrenamiento para estimar los parámetros necesarios, esto quiere decir que este clasificador puede ser extremadamente rápido en comparación a otros métodos más sofisticados.
Por otro lado, a pesar de estas bondades se conoce al clasificador *Naive Bayes* como un mal estimador, por lo que los resultados probabilísticos no deben considerarse para un estudio muy serio.
A continuación lo pondremos a prueba en nuestra ***data***:

Naive Bayes Aplicado

Explicamos en el capítulo anterior, el algoritmo de clasificación de Naive Bayes se basa en la idea de suponer todas las variables como independientes, por lo que los cálculos a nivel de costo computacional son menores y rápidos. A pesar de tener ciertos detalles es un clasificador que destaca por su buen rendimiento y más cuando la base de datos o información a analizar es pequeña.
Un detalle para entender aún más cómo funcionan los clasificadores Naive Bayes en R, es que la función **Naive Bayes** asume distribuciones tipo *Gaussianas* para las variables numéricas. Además, los valores a priori se calculan a partir de la porción de los datos de entrenamiento. Los prioris se muestran cuando se imprime el objeto. Los valores de Y son las medias y las desviaciones estándar de los predictores dentro de cada clase:

$$\mathbb{P}(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i-\mu)^2}{2\sigma_y^2}\right). \tag{4.6}$$

Para esto recurrimos al software estadístico **R**, con los paquetes:

```
library(e1071)
library(caTools)
library(caret)
```

Como vimos en *árboles de decisión* la librería *caTools* nos servirá de nuevo para la debida distribución y creación de particiones de nuestra *data*, así que volveremos a implementar estas dos librerías. La librería *e1071*([Dimitriadou et al., 2006]) la usaremos para la creación de nuestras tablas de probabilidad condicional que es con lo que arranca el algoritmo de **Naive Bayes**. Ahora explicaremos brevemente cómo es que aplicamos el algoritmo a nuestra *data*. Primeramente, hicimos un análisis previo para saber con qué estábamos tratando. Podemos ver que la variable a predecir de las 14 disponibles es el **diagnóstico**, en donde tomando en consideración que nuestras variables predictoras serán las 13 restantes.
En las pruebas de correlación podemos apreciar que para el "diagnóstico"tiene una correlación significativa con las variables de: *EDAD, SEXO, DL, PA, ELECTRO, ANGINA, DIE, V, THAL*. Las cuales nos ayudarán a comprender mucho mejor la distribución de las probabilidades.

```
set.seed(200)
particion_PE <- sample.split(diagnostico, SplitRatio = 0.80)
con_entre <- subset(Pacientes_Entrenamiento, particion_PE==TRUE)
con_test <- subset(Pacientes_Entrenamiento, particion_PE==FALSE)
```

Empezaremos creando dos conjuntos tal que uno de ellos será de *entrenamiento* y otro será de

prueba la partición será con una división de radio 0.80, esto nos permitirá entrenar [3] y evaluar el algoritmo. Ahora procederemos a la función:

```
nb_clasificacion <- naiveBayes(x = con_entre[-14],
                               y = con_entre$diagnostico)
nb_clasificacion$
```

La función *naiveBayes* está incorporada en la librería **e1071**, la cual supone la independencia de todas las variables como explicamos al inicio. Así como también supone la distribución Gaussiana. Lo que nos arroja la siguiente tabla de probabilidades condicionales.

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = con_entre[-14], y = con_entre$diagnostico)
##
## A-priori probabilities:
## con_entre$diagnostico
## Sin Presencia Con Presencia
##     0.5378151     0.4621849
##
## Conditional probabilities:
##                       Edad
## con_entre$diagnostico     [,1]     [,2]
##         Sin Presencia 53.09375 9.385395
##         Con Presencia 56.61818 7.672161
##
##                       Sexo
## con_entre$diagnostico         0         1
##         Sin Presencia 0.4609375 0.5390625
##         Con Presencia 0.1727273 0.8272727
##
##                       DL
## con_entre$diagnostico          1          2          3          4
##         Sin Presencia 0.10156250 0.25000000 0.39062500 0.25781250
##         Con Presencia 0.05454545 0.07272727 0.14545455 0.72727273
##
##                       PA
## con_entre$diagnostico     [,1]     [,2]
##         Sin Presencia 129.1094 16.47629
##         Con Presencia 135.7273 18.44402
##
##                       CS
## con_entre$diagnostico     [,1]     [,2]
```

[3] Entrenar: En los algoritmos, el termino entrenar se refiere a encaminar al algoritmo a una cierta meta, en base a prueba y error, cada error es una corrección dependiendo de los parámetros que tomemos en cuenta para determinar un acierto o un fallo.

```
##         Sin Presencia 244.2891 55.15721
##         Con Presencia 251.7545 47.82848
##
##                       Glucosa
## con_entre$diagnostico         0         1
##         Sin Presencia 0.8437500 0.1562500
##         Con Presencia 0.8454545 0.1545455
##
##                       Electro
## con_entre$diagnostico         0         1         2
##         Sin Presencia 0.5390625 0.0078125 0.4531250
##         Con Presencia 0.4000000 0.0000000 0.6000000
##
##                       RCM
## con_entre$diagnostico     [,1]     [,2]
##         Sin Presencia 158.2812 18.82114
##         Con Presencia 138.5818 22.78998
##
##                       Angina
## con_entre$diagnostico         0         1
##         Sin Presencia 0.8437500 0.1562500
##         Con Presencia 0.4636364 0.5363636
##
##                       DIE
## con_entre$diagnostico     [,1]      [,2]
##         Sin Presencia 0.606250 0.8258377
##         Con Presencia 1.607273 1.3329769
##
##                       PE
## con_entre$diagnostico         1         2         3
##         Sin Presencia 0.6484375 0.2890625 0.0625000
##         Con Presencia 0.2545455 0.6454545 0.1000000
##
##                       V
## con_entre$diagnostico         0         1         2         3
##         Sin Presencia 0.8125000 0.1171875 0.0468750 0.0234375
##         Con Presencia 0.3363636 0.3181818 0.2181818 0.1272727
##
##                       thal
## con_entre$diagnostico          3          6          7
##         Sin Presencia 0.82031250 0.03125000 0.14843750
##         Con Presencia 0.25454545 0.08181818 0.66363636
```

Usaremos estas probabilidades condicionales para clasificar y predecir debidamente. La librería *e1701* contiene una función llamada *predict* que justamente se encarga de hacer los cálculos de la predicción.

```
prediccion <- predict(nb_clasificacion, newdata = con_test[-14])
```

```
matriz_conf = table(con_test[,14],prediccion)
matriz_conf
```

Mediante la *matriz de confusión* evaluaremos la predicción.

```
##                prediccion
##                 Sin Presencia Con Presencia
##   Sin Presencia            27             5
##   Con Presencia             3            24
```

La librería *caTools* tiene una manera de hacer la verificación cruzada mediante su propio algoritmo y así podemos saber la exactitud con la que clasifica.

```
conf_caret <- confusionMatrix(prediccion,con_test[,14])
conf_caret
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Sin Presencia Con Presencia
##   Sin Presencia            27             3
##   Con Presencia             5            24
##
##                Accuracy : 0.8644
##                  95% CI : (0.7502, 0.9396)
##     No Information Rate : 0.5424
##     P-Value [Acc > NIR] : 1.458e-07
##
##                   Kappa : 0.7284
##  Mcnemar's Test P-Value : 0.7237
##
##             Sensitivity : 0.8438
##             Specificity : 0.8889
##          Pos Pred Value : 0.9000
##          Neg Pred Value : 0.8276
##              Prevalence : 0.5424
##          Detection Rate : 0.4576
##    Detection Prevalence : 0.5085
##       Balanced Accuracy : 0.8663
##
##        'Positive' Class : Sin Presencia
##
```

El algoritmo de clasificación, también nos permite predecir, mediante **R-project**, tenemos un modelo que clasifica y predice con una precisión de 0.8644, es uno de los más confiables pero que de manera teórica se toma muchas libertades.

Máquinas de Soporte Vectorial

Las Máquinas de Soporte Vectorial *(Support Vector Machine SVM)* es un método de aprendizaje supervisado que analiza los datos utilizados para la clasificación y el análisis de regresión. Se le asigna un conjunto de datos de entrenamiento, marcados como pertenecientes a cualquiera de las dos categorías. El modelo SVM es uno cuyo algoritmo asigna nuevas muestras a una categoría u otra, por lo que lo convierte en un clasificador no probabilístico. Un modelo de SVM es una representación de las muestras como puntos en el espacio, asignados de tal manera que las muestras de las categorías separadas se dividen por una brecha lo más clara y amplia posible. Esto se repite hasta que el algoritmo es capaz de predecir a qué categoría pertenecerán una serie de nuevas muestras que el algoritmo analizará. El objetivo de esta tesis no contempla el desarrollo argumental de este algoritmo, pero no podemos negar su poder predictivo, este algoritmo es un método de *clasificación - regresión* que fue desarrollado en la década de los 90's, dentro del campo de la ciencia computacional. Si bien originalmente se desarrolló como un método de clasificación binaria, su aplicación se ha extendido a problemas de clasificación múltiple y regresión. Se basa en el concepto de *híper plano*, comencemos por definir el híper plano. Aunque el objetivo de esta tesis no son los *sistemas de máquinas vectoriales* daremos una pequeña introducción a ellos de manera muy resumida, recomendamos *The Element of Stadistic Learning* [Friedman et al., 2001, pág 417].

Híperplano y Clasificador de Soporte Vectorial

En un espacio $p - dimensional$, un híperplano se define como un subespacio plano afín de dimensiones $p - 1$. El término afín significa que el subespacio no tiene porqué pasar por el origen. En un espacio de dos dimensiones, el híperplano es un subespacio de *1 - dimensión*, es decir, una recta. En un espacio tridimensional, un híperplano es un subespacio de dos dimensiones, un plano convencional. Para dimensiones $p > 3$ no es intuitivo visualizar un híperplano, pero el concepto de subespacio con $p - 1$ dimensiones se mantiene.
La definición matemática de un híperplano es bastante simple. En el caso de dos dimensiones, el híperplano se describe acorde a la ecuación de una recta:

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0. \tag{4.7}$$

Dados los parámetros $\beta_0, \beta_1, \beta_2$ todos los pares de valores $x = (x_1, x_2)$ para los que se cumple la igualdad, son puntos del hiperplano. Esta ecuación puede garantizarse para $p - dimensiones$:

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = 0. \tag{4.8}$$

Y de igual manera, todos los puntos definidos por el vector $x = x_1, x_2, \ldots, x_p$ que cumplen la ecuación pertenecen al híperplano. Cuando x no satisface la ecuación:

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p < 0. \tag{4.9}$$

o bien

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p > 0. \tag{4.10}$$

El punto x cae a un lado o al otro del híperplano. Así pues, se puede entender que un híperplano divide un espacio p-dimensional en dos mitades (figura 4.11). Para saber en qué lado del híperplano se encuentra un determinado punto x, solo hay que calcular el signo de la ecuación.

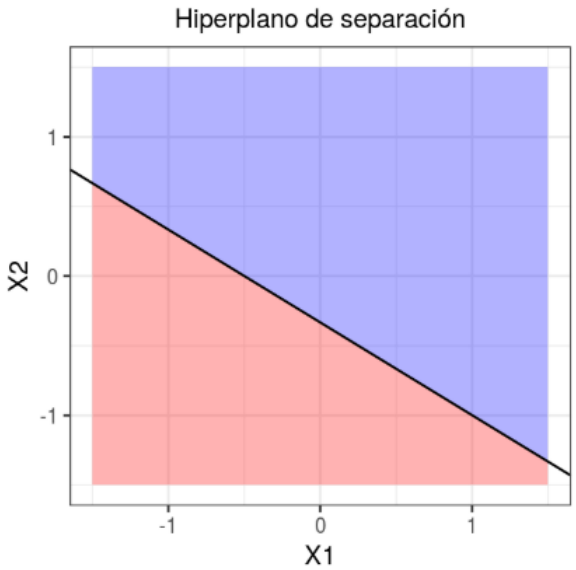


Figura 4.11: Híperplano en 1-dimensional.

Clasificación Binaria Empleando un Híperplano

Cuando se dispone de n observaciones, cada una con p predictores y cuya variable respuesta tiene dos niveles que identificaremos como $+1$ y -1, se pueden emplear híperplanos para construir un clasificador que permita predecir a qué grupo pertenece una observación en función de sus predictores. Este mismo problema puede abordarse también con otros métodos (regresión logística, árboles de clasificación, entre otros) cada uno con ventajas y desventajas. Para facilitar la comprensión, las siguientes explicaciones se basan en un espacio de dos dimensiones, donde un híperplano es una recta. Sin embargo, los mismos conceptos son aplicables a dimensiones superiores.

Casos Perfectamente Separables Linealmente

Supongamos que la distribución de las observaciones es tal que se pueden separar linealmente de forma perfecta en las dos clases $+1$ y -1, entonces un híperplano de separación cumple que:

$$\begin{aligned} \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p &> 0 \ si\ y_i = 1. \\ \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p &< 0 \ si\ y_i = -1. \end{aligned} \tag{4.11}$$

Al identificar cada clase como $+1$ o -1, y dado que multiplicar dos valores negativos resultan en un valor positivo, las dos condiciones anteriores pueden simplificarse en una única:

$$y_i(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p) > 0 \ para\ i = 1, \cdots, n. \tag{4.12}$$

Bajo este escenario, el clasificador más sencillo consiste en asignar cada observación a una clase dependiendo del lado del híperplano en el que se encuentre. Seleccionamos como clasificador óptimo al que se conoce como *híperplano óptimo de separación*, que se corresponde con el híperplano que se encuentra más alejado de todas las observaciones de entrenamiento. Para obtenerlo, se tiene que calcular la distancia perpendicular de cada observación a un determinado híperplano. La menor de estas distancias (conocida como margen) determina qué tan alejado está el margen de las observaciones de entrenamiento.

Clasificador de Soporte Vectorial

Anteriormente mencionamos una *máquina de soporte vectorial* se compone de híperplanos y de un algoritmo llamado *clasificador de soporte vectorial* el cual describiremos brevemente, la aplicación por separado de los híperplanos es muy poca ya que los requerimientos para que así se lleve a cabo son raros o difíciles de cumplir y aún en su manera ideal se presentan algunos inconvenientes, por esta razón es que nace la necesidad de crear un clasificador basado en un híperplano que, aunque no separe perfectamente las dos clases, sea más robusto y tenga mayor capacidad predictiva al aplicarlo a nuevas observaciones, esto es lo que justamente buscamos con este clasificador. Si bien la demostración matemática queda fuera del objetivo y de los alcances de esta tesis, pondremos el argumento matemático en la que este clasificador se basa en la siguiente ecuación que la llamamos *estándar* para el clasificador de soporte vectorial:

$$y_i(x_i^{\mathcal{T}}\beta + \beta_0) \geq M(1 - \xi_i) \tag{4.13}$$

Donde denotamos a ξ como las variables *flojas o slacks* $\xi = (\xi_1, \xi_2 \cdots \xi_N)$ y por otra parte tomamos a $M = \frac{1}{\|\beta\|}$ con las propiedades de ser máximo y finalmente a $x_i^{\mathcal{T}}$ como la variable x definida en el híperplano como:

$$\{x : f(x) = x^{\mathcal{T}}\beta + \beta_0 = 0\} \tag{4.14}$$

Máquinas de Soporte Vectorial

Describimos anteriormente se consiguen buenos resultados cuando el límite de separación entre clases es aproximadamente lineal. Si no lo es, su capacidad decae drásticamente. Una estrategia para enfrentarse a escenarios en los que la separación de los grupos es de tipo no lineal consiste en expandir las dimensiones del espacio original.
El hecho de que los grupos no sean linealmente separables en el espacio original no significa que no lo sean en un espacio de mayores dimensiones. Las imágenes de la figura 4.12 muestran cómo dos grupos, cuya separación en dos dimensiones no es lineal, sí lo es al añadir una tercera dimensión.
El método de Máquinas de Soporte Vectorial se puede considerar como una extensión de *Support Vector Classifier*, obtenida al aumentar la dimensión de los datos. Los límites de separación lineales generados en el espacio aumentado se convierten en límites de separación no lineales al proyectarlos en el espacio original.

Aumento de la Dimensión, Kernels

Una vez definido que las máquinas de soporte vectorial siguen la misma estrategia que el *Support Vector Classifier*, pero aumentando la dimensión de los datos antes de aplicar el algoritmo, la pregunta inmediata es: ¿Cómo se aumenta la dimensión y qué dimensión es la correcta?. La dimensión de un conjunto de datos puede transformarse combinando o modificado cualquiera de sus dimensiones. Por ejemplo, se puede transformar un espacio de dos dimensiones en uno de tres aplicando la siguiente función:

$$f(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2). \tag{4.15}$$

Ésta es solo una de las infinitas transformaciones posibles, ¿Cómo saber cuál es la adecuada?, Es aquí donde los *kernel* entran en juego. Un kernel K es una función que devuelve el resultado del *producto punto* entre dos vectores realizado en un nuevo espacio dimensional distinto al espacio original en el que se encuentran los vectores. Aunque no hemos entrado en detalle en las fórmulas matemáticas empleadas para resolver el problema de optimización, ésta contiene un *producto*

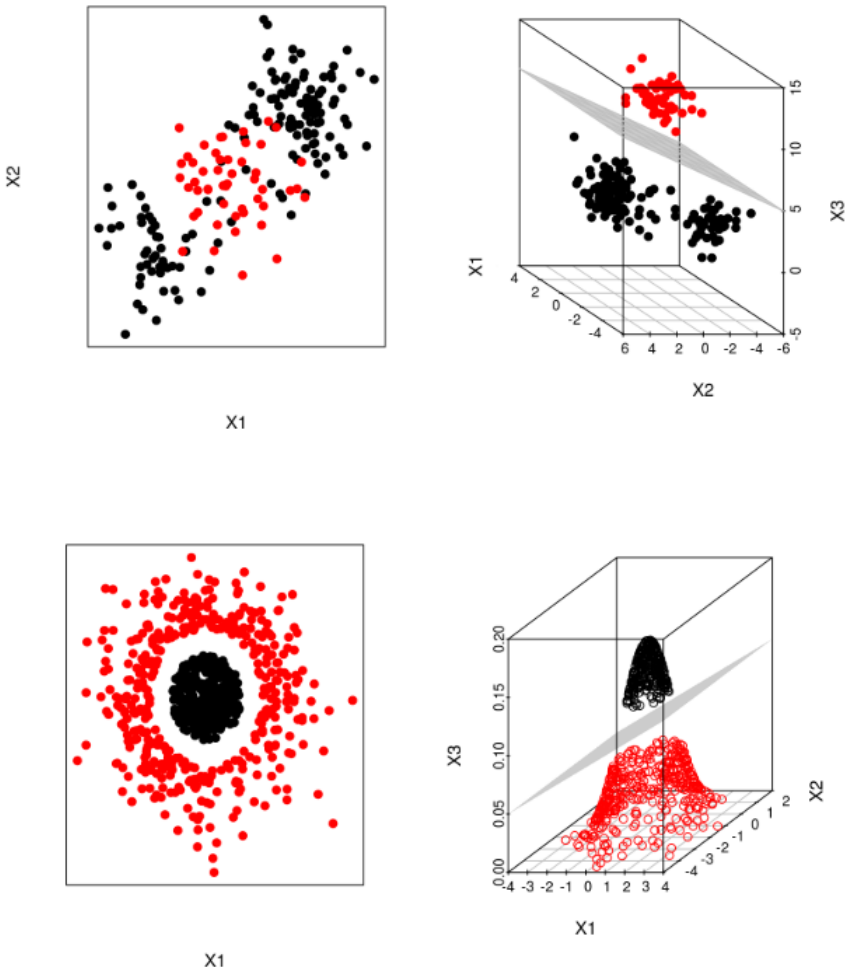


Figura 4.12: Clasificación del SVM.

punto. Si se sustituye este *producto punto* correspondiente al kernel. Ha esto se le conoce como *Kernel Trick*, porque con solo una ligera modificación del problema original, se puede obtener el resultado para cualquier dimensión, esto gracias a los *kernels*. Existe una gran variedad de *kernels* distintos, entre los más usados destacan:

- **Kernel lineal.**
- **Kernel polinómico.**
- **Kernel Gaussiano.**

Ahora que tenemos las nociones básicas de qué es una máquina de soporte vectorial, aplicaremos este algoritmo en nuestra *data* y evaluaremos su desempeño.

Máquina de Soporte Vectorial (SVM) en nuestra data

Empezaremos cargando las debidas librerías que ocuparemos para el análisis que son las mismas que hemos utilizado en anteriores algoritmos (árboles de decisión y Naive Bayes).

```
library(e1071)
library(caTools)
```

```
library(caret)
```

Ahora crearemos dos particiones de la *data*, que llamaremos: *entrenamiento y test*, las cuales nos servirán para construir la máquina de soporte vectorial.

```
set.seed(123)
particion_SVM = sample.split(diagnostico, SplitRatio = 0.85)
SVM_entre = subset(Pacientes_Entrenamiento, particion_SVM == TRUE)
SVM_test = subset(Pacientes_Entrenamiento, particion_SVM == FALSE)
```

Ahora ya que tenemos nuestros conjuntos divididos en un radio de 0.85, crearemos nuestra máquina de soporte vectorial mediante la librería *e1071* [Dimitriadou et al., 2006], la cual contiene la función llamada *svm*, dicha función contiene lo que hemos explicado brevemente en esta sección.

```
SVM_clasif = svm(formula = diagnostico ~ .,
                 data = SVM_entre,
                 type = 'C-classification',
                 kernel = 'linear')
SVM_clasif
```

Lo cual nos arroja lo siguiente:

```
## Call:
## svm(formula = diagnostico ~ ., data = SVM_entre, type = "C-classification",
##     kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##       gamma:  0.04761905
##
## Number of Support Vectors:  97
```

Una vez creada la máquina de soporte vectorial, tenemos que ver su eficiencia a la hora de clasificar y predecir, por lo que apoyándonos en la librería *caret* y cuando la función *predict* de la librería *e1071* vamos a realizar la predicción y crear la matriz de confusión.

```
SVM_predic = predict(SVM_clasif, newdata = SVM_test[-14])
matriz_confSVM = table(SVM_test[, 14], SVM_predic)
matriz_confSVM
conf_caretSVM <- confusionMatrix(SVM_predic,SVM_test[,14])
conf_caretSVM
```

```
##                SVM_predic
##                 Sin Presencia Con Presencia
##   Sin Presencia            21             3
##   Con Presencia             7            14
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Sin Presencia Con Presencia
##   Sin Presencia            21             7
##   Con Presencia             3            14
##
##                Accuracy : 0.7778
##                  95% CI : (0.6291, 0.888)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 0.0006235
##
##                   Kappa : 0.5482
##  Mcnemar's Test P-Value : 0.3427817
##
##             Sensitivity : 0.8750
##             Specificity : 0.6667
##          Pos Pred Value : 0.7500
##          Neg Pred Value : 0.8235
##              Prevalence : 0.5333
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.6222
##       Balanced Accuracy : 0.7708
##
##        'Positive' Class : Sin Presencia
##
```

Esto nos deja ver que la precisión del *SVM* es de 0.7778 lo que no está mal, pero ya llegará el momento de comparar, en este algoritmo se puede hacer mucho más, por ejemplo, cambiar el tipo de kernel, pero no es el objetivo profundizar mucho más en estos algoritmos, por lo que nos conformamos con el modelo lineal y kernel estándar.

Capítulo 5

Concluciones

Realizamos la implementación de los algoritmos a nuestra base de datos que previamente mostramos y explicamos, a su vez los conjuntos de datos fueron entrenados para cada algoritmo, con esto creamos diversas muestras para cada algoritmo, generadas con R. El objetivo de esta tesis es el poder comprobar que el algoritmo *Naive Bayes* es un buen algoritmo para la predicción y mostrar su poder para el área médica, el grado de error lo medimos generando las matrices de confusión (tabla 5.1), en la tabla podemos apreciar que dicho algoritmo tiene una buena precisión. El más preciso entre los que se comparó, tiene mucho que ver el tratamiento que se le dio a la información.

Algoritmo	Precisión
Árbol de Decisión	0.7727
Naive Bayes	0.8644
Máquina de Soporte Vectorial	0.7778

Tabla 5.1: Tabla de Precisión de Algoritmos.

Resaltamos el hecho de que el tratamiento de los datos es diferente según sea el algoritmo, la tesis no estaba enfocada a los algoritmos no probabilísticos como lo es el SVM (*máquinas vectoriales*), por lo que la aplicación del SVM fue la más básica y por ello nos referimos a un *kernel* lineal, el cual se ajusta a ciertas características de la información. Una de las características de los SVM es la versatilidad de los polinomios para complejizar o simplificar los cálculos, por otra parte dividimos a los algoritmos basados en redes Bayesianas, ya que usamos un algoritmo para medir su veracidad a la hora de predecir, llamado *Validación Cruzada k-fold*, el cual nos ayudó a medir su precisión mediante la función de pérdida.

Algoritmo	Precisión
Red Bayesiana con Naive Bayes	0.27846670
Red Bayesiana TAN	0.15349003

Tabla 5.2: Tabla de precisión de redes Bayesianas.

La tabla 5.2 nos deja ver que el mejor algoritmo o al menos que su función de pérdida no es tan grande es: **La red Bayesiana TAN**, la cual falla en aproximar en clasificar a uno de los elementos tomados en los *k-folds* que por estandarización de método toma como máximo 10 muestras o según la literatura *runs* [Scutari and Ness, 2012].

Interpretación de los resultados

Como tenemos entendido en el ejemplo y en el análisis previo de la información que tenemos a la mano, tenemos una serie de pacientes los cuales están desarrollando o están por desarrollar una enfermedad cardíaca. Los anteriores algoritmos citados, nos ayudaron a clasificar y a predecir, según su cardiopatía o características mencionadas en la *data* qué pacientes son más propensos a desarrollar una enfermedad cardíaca.
De este estudio podemos sacar mucha información importante pero la principal información o el objetivo de esta tesis era encontrar la mejor manera de clasificar y poder predecir qué pacientes desarrollarán una enfermedad cardíaca según las variables que se hayan considerado de la data.
Lo que tenemos es que la mejor opción para los algoritmos de clasificación es el algoritmo de **Naive Bayes** y para las redes Bayesianas el mejor es **la red Bayesiana TAN**, a su manera diferente de clasificar y predecir pero en general bajo estas variables la *red Bayesiana TAN* es más efectiva y poderosa pero no menos importante que los anteriores algoritmos para el modelo.
Nuestro objetivo en esta tesis era justamente ver el contraste en los algoritmos de clasificación y las redes Bayesianas, es evidente que cada algoritmo tendrá sus fortalezas como sus debilidades. Como vimos de manera particular *Naive Bayes* es un algoritmo que da, según nuestras matrices de confusión con gran precisión a la hora de clasificar. Pero por otra parte las redes Bayesianas son algoritmos sumamente dinámicos y adaptativos a la información, ya que toma de la *teoría de grafos* su manera de tratar la información de forma gráfica y así mediante algunos conceptos de *teoría de la información*, optimizar y mejorar el algoritmo según sea el caso. Por lo que hacemos gran énfasis a su uso para la resolución de problemas.

Campos de Oportunidad

La aplicación de una tecnología prometedora como el aprendizaje automático a la predicción inicial de las enfermedades cardíacas tendrá un profundo impacto en la sociedad. Dado que las enfermedades cardíacas son un asesino importante en nuestro país, y esto como bien nos dice la *MBE* puede ayudar a diagnosticar de manera más veraz y temprana una afectación al corazón. Inicialmente esta información puede ayudar a generar alternativas médicas o no médicas (campañas de concientización) para una prevención o diagnóstico temprano de las enfermedades cardíacas, ya que es un problema actual serio de salud y más en una sociedad mexicana.
Los algoritmos de *machine learning* y basándonos en el concepto de aprendizaje automático, el conjunto de datos recién entrenado puede usarse para un sistema de predicción aún más confiable. Actualmente está en incremento su aplicación, y la medicina es un campo al cual puede ayudar en muchos aspectos. El futuro de la *MBE* es principalmente de apoyo para la toma de decisiones y el objetivo de los algoritmos de *machine learning* es justamente la incorporación más seria a todas las disciplinas a las que se puedan adaptar. No menos importante una gran ventana de oportunidad para los algoritmos de *machine learning* es el poder identificar dentro del espectro de variables que tenemos, las más importantes numéricamente hablando, aplicado a la medicina puede reducir costes de tiempo en estudios para la fácil detección de padecimientos, tomando como base principal el análisis estadístico previo que ayudaran a descartar variables que pueden traducirse en estudios clínicos, pruebas y punto de vista colaborativos entre varios especialistas, pues un padecimiento puede tener múltiples conexiones con otras especialidades. Es por ello que estos algoritmos mejoran día a día creando mejores modelos y formas de optimizar el tiempo de respuesta, pues los problemas pueden llevar a problemas mucho mas complejos hablando computacionalmente, el medico puede tomar una decisión basada no solo en el conocimiento experto si no también en el análisis estadístico

de las variables.
Las aplicaciones a modelos de la vida real son variadas, por cada aplicación existe una ventana de oportunidad de poder mejorar dicho servicio o trabajo en el que decidamos implementar los algoritmos, personalmente pienso que poco a poco iremos viendo la incorporación de dichos algoritmos no solo a la medicina si no también a áreas con grandes bases de datos que necesiten una perspectiva distinta.

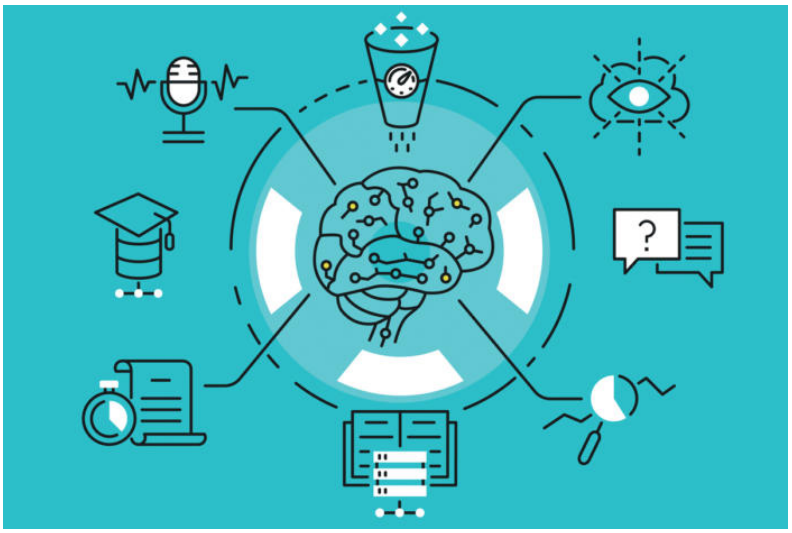


Figura 5.1: Machine learning en la actualidad.

La predicción del clima es una muy probable aplicación, día a día se generan una gran cantidad de datos la cual se almacena y clasifica según su importancia o estudio que se vaya a realizar, es cierto que el clima es uno de los temas con mas variabilidad y complejos, pero con estos algoritmos podemos establecer estimaciones serias y aproximaciones a lo que seria un modelo certero de una predicción climatológica. Los usos de estos resultados son varios, desde la industria de la construcción y bienes raíces, hasta agricultura e impactos ambientales.
Su futuro es prometedor ya que el mundo mira con buenos ojos a la modernidad y la automatización de servicios, donde estos algoritmos son de gran ayuda.

Bibliografía

[Casas, 2012] Casas, J. G. (2012). *Introducción a la Teoría de Códigos y de la Información*, volume 1. UNAM Facultad de Ciencias.

[Dimitriadou et al., 2006] Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D., Weingessel, A., and Leisch, M. F. (2006). The e1071 package. *Misc Functions of Department of Statistics (e1071), TU Wien*.

[Friedman et al., 2001] Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The Elements of Statistical Learning*, volume 1. Springer series in statistics New York.

[Kabir et al., 2017] Kabir, M. R., Onik, A. R., and Samad, T. (2017). A network intrusion detection framework based on bayesian network using wrapper approach. *International Journal of Computer Applications*, 166(4):13–17.

[Kuhn et al., 2008] Kuhn, M. et al. (2008). Building predictive models in R using the caret package. *Journal of statistical software*, 28(5):1–26.

[Moreno Rodríguez, 2005] Moreno Rodríguez, M. Á. (2005). La medicina basada en la evidencia y la práctica médica individual. *Revista Cubana de Medicina*, 44(3-4):0–0.

[Pattekari and Parveen, 2012] Pattekari, S. A. and Parveen, A. (2012). Prediction system for heart disease using naïve bayes. *International Journal of Advanced Computer and Mathematical Sciences*, 3(3):290–294.

[Scutari and Ness, 2012] Scutari, M. and Ness, R. (2012). bnlearn: Bayesian network structure learning, parameter learning and inference. *R package version*, 3.

[Soni et al., 2011] Soni, J., Ansari, U., Sharma, D., and Soni, S. (2011). Predictive data mining for medical diagnosis: An overview of heart disease prediction. *International Journal of Computer Applications*, 17(8):43–48.

[Sucar, 2015] Sucar, L. E. (2015). *Probabilistic Graphical Models*, volume 10. Springer.

[Therneau et al., 1997] Therneau, T. M., Atkinson, E. J., et al. (1997). An introduction to recursive partitioning using the rpart routines.