
Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias Físico Matemáticas


# Introducción al aprendizaje computacional a través de modelos estadísticos y probabilistas

Tesis presentada al

**Colegio de Matemáticas**

como requisito parcial para la obtención del grado de

**Licenciado en Matemáticas Aplicadas**

por


Armando Martínez Ruiz

Asesorado por

Dra. Hortensia Josefina Reyes Cervantes



Puebla Pue.
Mayo de 2023


Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias Físico Matemáticas


# Introducción al aprendizaje computacional a través de modelos estadísticos y probabilistas

Tesis presentada al

**Colegio de Matemáticas**

como requisito parcial para la obtención del grado de

**Licenciado en Matemáticas Aplicadas**

por


Armando Martínez Ruiz

Asesorado por

Dra. Hortensia Josefina Reyes Cervantes



Puebla Pue.
Mayo de 2023

**Título:** Introducción al aprendizaje computacional a través de modelos estadísticos y probabilistas
**Estudiante:** Armando Martínez Ruiz

COMITÉ

Dr. Bulmaro Juárez Hernández
Presidente

Dr. Francisco Tajonar Sanabria
Secretario

Dr. Fernando Velasco Luna
Vocal

Dra. Brenda Matías Castillo
Vocal

Dra. Hortensia Josefina Reyes Cervantes
Asesor

# Agradecimientos

Quiero agradecer principalmente, a mis padres y mi hermana, que siempre me han apoyado en el tiempo que he cursado mis estudios. A mi esposa Audrey, por su apoyo incondicional y por acompañarme en todo lo que me he propuesto.

Agradezco tambien a las personas que tuve la dicha de conocer y compartir el aula en la facultad: Marco Polo, que es el primer amigo que hice en la facultad y de quien aprendí mucho al inicio de la carrera; Hugo, Nava, Munive, Alba, Christian, Gustavo (Galufeta), Itzel, Natalia, Jazmin y Luis, con quienes compartí muchos momentos peculiares y significativos durante la carrera, a Angel Raúl, Rafa, Jorge, Arturo, Christian (químico), Bruno y Yerox, con quienes compartí el gusto de la docencia, y una mención especial al Dr. Lino por todo su apoyo en mi decisión de seguir en la maestría y al Dr. René Ponce por su amistad y sabio consejo.

Por último, quiero agradecer a mis maestros que tuve a lo largo de la carrera, pues sus conocimientos y su apoyo me permitieron conocer el mundo de las matemáticas y me motivaron a buscar más; de la misma manera, agradezco de forma particular a la Dra. Lucía Cervantes, al Dr. Raúl Linares, al M.C. Julio Poisot, al Dr. Francisco Tajonar, al Dr. César Cejudo, a la Dra. Patricia Domínguez, al M.C: Sergio Adán, a la profesora Raggi y al Dr. Escamilla. Y un agradecimiento muy especial a la Dra. Hortensia Reyes, pues este trabajo no sería posible sin su apoyo e infinita paciencia.

Siempre tendré un profundo agradecimiento hacia todos ellos.

# Índice general

# Índice de figuras

# Índice de tablas

# Resumen

El aprendizaje computacional, en general, trata de construir programas o modelos que puedan mejorar su desempeño de forma automática a través de la experiencia. Esta área de la inteligencia artificial puede dividirse en tres principales vertientes: el aprendizaje supervisado, el aprendizaje no supervisado y el aprendizaje por refuerzo. Para dar una introducción al área del aprendizaje computacional, se parte de los modelos de regresión lineal, utilizados para realizar predicciones de un valor numérico y que permiten ilustrar las bases del aprendizaje computacional, así como los procedimientos para tener mejores modelos como los métodos de optimización para las funciones de costo o la selección de atributos. Otra tarea fundamental dentro del aprendizaje computacional es la clasificación, que consiste en predecir el valor de una variable categórica dada una observación. Además, la evaluación de modelos es una parte importante que permite ilustrar técnicas estadísticas para apoyar con cierto grado de significancia la elección de un determinado modelo. Estas tareas de regresión y clasificación pueden ser vistas a través de los conceptos teóricos estudiados en los cursos formativos de la licenciatura, de modo que estos conceptos pueden servir como base para el estudio inicial del aprendizaje computacional.

**Palabras clave:** *Machine Learning (Aprendizaje Computacional), regresión, clasificación, gradiente de descenso, backwards elimination, Naive Bayes.*

# Introducción

En la actualidad, la cantidad de datos que existe va mas allá de la comprensión humana, ya que aumenta exponencialmente con cada minuto, y esto nos ha llevado a desarrollar herramientas para el análisis de dicha información. La inteligencia artificial se menciona actualmente cuando se habla del análisis de datos y pronósticos, y con una cantidad tan abundante de información, se han creado algoritmos y modelos que permiten procesar y aprender de la información, para crear sus propias conclusiones sobre ella; además, la inteligencia artificial ha tenido un gran impulso en la actualidad debido a tres factores importantes: el poder computacional, la disponibilidad de la información y el bajo costo de aplicación. El aprendizaje computacional es un subcampo de la inteligencia artificial que permite realizar estas tareas.

Podemos definir al aprendizaje computacional o Machine Learning, como un conjunto de métodos o funciones que tienen como objetivo la identificación de patrones en un conjunto de datos determinado, y luego utilizar esos patrones para hacer pronósticos sobre los datos mencionados. También es conocido como análisis predictivo o aprendizaje estadístico. En general, los algoritmos de aprendizaje más eficientes son aquellos que automatizan el proceso de toma de decisiones encontrando un patrón general a partir de ejemplos conocidos. El objetivo de esta tesis es hacer una introducción al aprendizaje computacional, utilizando conceptos aprendidos durante la licenciatura en los cursos básicos, ya que el aprendizaje computacional es un área interdisciplinaria que requiere para su comprensión, conceptos de diferentes áreas de las matemáticas.

El aprendizaje computacional, se puede clasificar de acuerdo a la naturaleza de los datos en tres tipos principales: el aprendizaje supervisado, el aprendizaje no supervisado y el aprendizaje por refuerzo. Generalmente, cuando se plantea usar un algoritmo de aprendizaje para resolver un problema de predicción o clasificación, el algoritmo seleccionado dependerá del tipo de datos que se utilizarán. Hay dos preguntas esenciales que son de importancia antes de abordar el problema: ¿Cuál es la naturaleza de los datos? y ¿La información resultante es una variable continua o una variable discreta?. En el enfoque predictivo o supervisado, el objetivo es que una función con entradas $x$ y salidas $y$ aprenda, dado un conjunto indexado de parejas de entradas y salidas $D = \{(x_i, y_i)\}_{i=1}^{n}$, con un número natural $n$ fijo. En este caso, $D$ es llamado conjunto de entrenamiento ("training set") y es finito.

En el caso más simple, cada entrada $x_i$ es un vector. A cada uno de estos $x_i$ se les conoce como atributos ("features"), ejemplos o covariables. En general, los $x_i$ pueden representar a un objeto complejamente estructurado, por ejemplo, puede ser una imagen, un audio, una gráfica, una figura molecular, etc. De forma similar, la salida o variable de respuesta puede ser lo que sea, pero en la mayoría de los casos, se asume que $y_i$ es una variable cualitativa de un conjunto finito $y_i \in \{1, ..., c\}, c \in \mathbb{N}$, o puede ser que $y_i$ sea un número real.

Cuando $y_i$ es una variable cualitativa, es decir, una variable que representa una cualidad, clase o categoría [1], entonces tenemos un problema de clasificación o reconocimiento de patrones, y

cuando $y_i$ es un número real, entonces el problema es conocido como regresión.

En clasificación, el objetivo es predecir una etiqueta de clase, la cual es una opción de una lista de posibilidades predefinida; cuando se cuenta con únicamente 2 clases, decimos que el problema es de clasificación binaria. Algunos de los algoritmos más comunes de clasificación son la regresión logística, los árboles de decisión, los clasificadores Bayesianos simples y las máquinas de soporte vectorial lineal ('linear support vector machines'). Mientras que para tareas de regresión, el objetivo es predecir un número real. En este caso, el algoritmo más simple es el de regresión lineal múltiple.

El segundo tipo principal de aprendizaje computacional es el enfoque descriptivo o no supervisado. En este caso solo se trabaja con las entradas, y el conjunto $D$ sería $D = \{(x_i)\}_{i=1}^{n}$, y el objetivo es encontrar "patrones particulares" en los datos. El problema aquí tiene menos restricciones, pues no se sabe con certeza el tipo de patrones que el modelo buscará. El tercer tipo es el aprendizaje por refuerzo, donde un agente aprende a comportarse, a través de prueba y error, en un ambiente dinámico. En este caso, el objetivo es encontrar una secuencia de pasos que pueda tomar el agente de tal forma que maximice una señal de recompensa dada como una función de sus acciones y sus estados dentro del ambiente. En esta tesis nos enfocaremos únicamente al estudio del aprendizaje supervisado, ya que es el punto de partida en el estudio del aprendizaje computacional.

El contenido de la tesis esta estructurado de la siguiente manera: el Capítulo 1 que abarca el tema de regresión lineal, comenzando con los modelos de regresión lineal simple para tareas de predicción, asi como los modelos de regresión lineal múltiple para ilustrar la optimización de funciones de pérdida o la selección de atributos. El Capítulo 2 se enfoca en algunos modelos de clasificación, como los clasificadores Bayesianos o el modelo de regresión logística, y su relación con las redes neuronales artificiales. Finalmente, en el Capítulo 3 se hace una revisión sobre la evaluación de modelos de aprendizaje, que se pueden llevar a cabo con pruebas estadísticas no paramétricas. Además, en la primera parte del Apéndice se incluye un glosario de definiciones de conceptos básicos que no se mencionan en el cuerpo de la tesis para mantener fluido el estudio del tema de interés. En la segunda parte se incluyen los códigos de R y Python que se utilizaron para los ejemplos del Capítulo 1.

# Capítulo 1

# Modelos de Regresión

La regresión es un método para modelar un valor no conocido basado en predictores independientes. En el contexto de aprendizaje, podemos utilizar la regresión para hacer predicciones de un valor continuo utilizando un conjunto de datos conocido. Este caso es el más sencillo de abordar, pues parte del concepto de regresión lineal simple, donde únicamente estaremos interesados de la relación entre la variable predictora y la variable de respuesta, así como de las predicciones que el modelo pueda otorgarnos. Más adelante, cuando el problema aumenta de complejidad, podremos ver que algunas variantes de la regresión lineal sirven como modelos de aprendizaje computacional.

## 1.1. Regresión Lineal

Comenzaremos el estudio del aprendizaje computacional utilizando el modelo simple de regresión lineal. El modelo se trata de una función lineal respecto a dos coeficientes que surgen a partir del conjunto de datos que se va a utilizar. Podemos escribir el modelo de la siguiente forma:

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i, i = 1, 2, ..., n,$$

donde $Y_i$ es una variable aleatoria y $X_i$ es una variable observable. Tenemos que $\beta_0$ y $\beta_1$ son parámetros desconocidos y serán fijos una vez se estimen. Por último, $\epsilon_i$ es una variable aleatoria a la que se le conoce como error residual. Normalmente, se supone que el valor esperado de $\epsilon_i$ sea cero, es decir, $E(\epsilon_i) = 0$, por lo que podemos representar la respuesta esperada como una línea recta:

$$E(Y_i) = \beta_0 + \beta_1 X_i.$$

También decimos que $\beta_0$ es la intersección con el eje o término de sesgo y $\beta_1$ es la pendiente de la ecuación de regresión [2]. Esta línea recta está representada en la Figura 1.1.

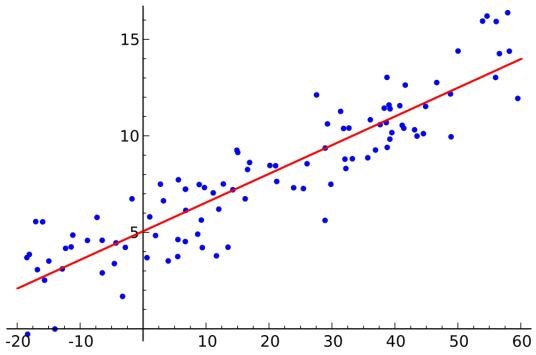


Figura 1.1: Regresión lineal simple.

Para poder utilizar el modelo de regresión lineal, es necesario tener en cuenta las suposiciones que involucran distribuciones de probabilidad sobre el modelo. Entonces, para el modelo $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$, tenemos lo siguiente:

- $Y$ es una variable aleatoria cuya distribución de probabilidad depende de $X$. Es decir, para cualquier valor de $X$, $Y$ tiene una distribución de probabilidad con media $\mu_{Y|X}$ y varianza $\sigma^2_{Y|X}$. Con esta suposición, podemos entonces escribir la respuesta esperada del modelo como:

$$\mu_{Y|X_i} = \beta_0 + \beta_1 X_i.$$

- Modelo de línea recta. Con base en la suposición anterior, podemos entonces escribir al modelo como $Y = \beta_0 + \beta_1 X_i + \epsilon_i$, el cual corresponde a una función lineal. Observemos que esta suposición da lugar a los siguientes resultados:

$$E(\epsilon_i) = 0,$$
$$V(Y_i) = V(\beta_0 + \beta_1 X_i + \epsilon_i) = V(\epsilon_i).$$

- Homogeneidad de varianzas. Las varianzas de las distribuciones de $Y$ son iguales para cualquier $X$, es decir,

$$\sigma^2_{Y|X_1} = \sigma^2_{Y|X_2} = ... = \sigma^2_{Y|X_n} = \sigma^2.$$

- Independencia. Los valores de $Y$ deberán ser estadísticamente independientes de $X$.

- Normalidad. La distribución de $Y$ para cualquier valor de $X$ es normal. Debido a esto, tenemos que la variable aleatoria $\epsilon_i$ es normal, pues $X$ se toma como una variable arbitraria. Tenemos entonces que $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$.

Cuando no se cumple alguna de las suposiciones del modelo de regresión lineal, la estimación por mínimos cuadrados de los parámetros pierde eficiencia, pues los sesgos en los errores estimados pueden llevarnos a hacer inferencias y predicciones inválidas; además, utilizar el método de máxima verosimilitud implicaría minimizar la función logarítmica de verosimilitud, que es computacionalmente más compleja cuando el conjunto de datos es muy extenso. Una vez que tengamos la propuesta del modelo, haremos pruebas sobre los parámetros y los errores $\epsilon_i$ para valorar qué tanto se ajusta éste al conjunto de datos.

Para ejemplificar el procedimiento del uso de regresión lineal simple en un contexto de aprendizaje, tomaremos un conjunto de datos extraido de la base de datos Kaggle [3], que consiste en 200

observaciones que pretenden representar la relación entre las ventas de un producto o servicio y la inversión en publicidad por televisión. Además, se emplea el código del Apéndice 2.1 para realizar los cálculos necesarios para la generación y validación del modelo.

En general, los datos de entrada para la generación de un modelo de aprendizaje van a tener la siguiente forma:

| **Variable independiente** $X$ | **Variable dependiente** $Y$ |
|---|---|
| $x_1$ | $y_1$ |
| $x_2$ | $y_2$ |
| . | . |
| . | . |
| . | . |
| $x_n$ | $y_n$ |

En caso de tener una matriz de ejemplos en lugar de un vector, es necesario que la variable dependiente ($Y_i$) se encuentre en la última columna de izquierda a derecha. Puesto que queremos encontrar información relevante y hacer predicciones sobre un número real, podemos observar el conjunto de datos en el plano para poder conjeturar que existe una relación lineal entre las variables propuestas.

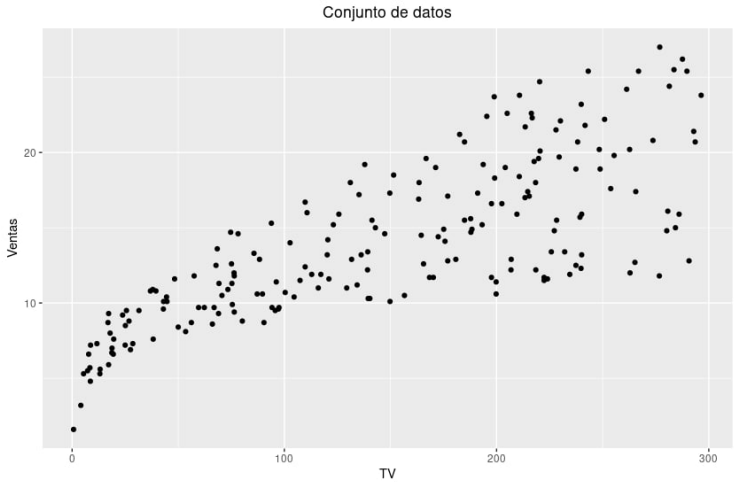


Figura 1.2: Gráfico de los datos extraídos de Kaggle entre $X$ y $Y$, que presentan una tendencia lineal.

Tomando en cuenta la forma en la que las observaciones se encuentran en el plano, podemos suponer que existe una relación lineal positiva entre las variables, debido a la orientación ascendente que tiene el conjunto de datos. El objetivo entonces es encontrar un modelo de la siguiente forma:

$$Y(X) = \beta_0 + \beta_1 X + \epsilon,$$

que represente dicha relación de las variables $X$ y $Y$. En la expresión anterior, $\beta_0$ y $\beta_1$ son dos constantes desconocidas que serán estimadas a partir del conjunto de datos dado. De esta forma podremos hacer predicciones sobre la variable dependiente $Y$. Para ello utilizaremos el conjunto de entrenamiento $D$ para estimarlos por el método de mínimos cuadrados, donde el objetivo es minimizar la expresión:

$$S(\beta_0, \beta_1) = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (Y_i - (\beta_0 + \beta_1 X_i))^2.$$

A S se le conoce como función de suma de cuadrados. Dado que conocemos los valores de $X_i$ y $Y_i$, podemos estimar los valores de $\beta_0$ y $\beta_1$ a partir de la función $S$, derivando respecto a los parámetros e igualando a cero, de modo que obtenemos las siguientes ecuaciones:

$$\frac{\partial S}{\partial \beta_0} = -2 \sum_{i=0}^{n} (Y_i - \beta_0 - \beta_1 X_i),$$
$$\frac{\partial S}{\partial \beta_1} = -2 \sum_{i=0}^{n} X_i (Y_i - \beta_0 - \beta_1 X_i).$$

Ahora, igualando a cero estas expresiones, tenemos lo siguiente:

$$\sum_{i=0}^{n} (Y_i - \beta_0 - \beta_1 X_i) = 0,$$
$$\sum_{i=0}^{n} X_i (Y_i - \beta_0 - \beta_1 X_i) = 0.$$

Que son equivalentes al siguiente sistema de ecuaciones:

$$\hat{\beta}_0 n + \hat{\beta}_1 \sum_{i=0}^{n} X_i = \sum_{i=0}^{n} Y_i,$$
$$\hat{\beta}_0 \sum_{i=0}^{n} X_i + \hat{\beta}_1 \sum_{i=0}^{n} X_i^2 = \sum_{i=0}^{n} X_i Y_i.$$

Estas ecuaciones son conocidas como ecuaciones normales. Las soluciones $\hat{\beta}_0$ y $\hat{\beta}_1$ son los parámetros estimados para $\beta_0$ y $\beta_1$.

Hay que destacar que, aunque los estimadores por mínimos cuadrados nos proporcionan ciertas propiedades de los estimadores, que nos sirven para hacer inferencias probabilísticas sobre ellos (entre otras), no es el único método para encontrar a los coeficientes $\beta_0$ y $\beta_1$. Otro método es el gradiente de descenso, donde se plantea una función de costo, que es una variante de la ecuación anterior. Es más común utilizar estos métodos cuando tenemos más de dos atributos (o variables independientes), ya que la alta dimensionalidad ocasiona que $X$ sea una matriz que podría no tener inversa o su cálculo podría resultar en un costo computacionalmente alto. Para el caso de un solo atributo, el método de mínimos cuadrados funciona adecuadamente en cualquier caso.

Tenemos entonces que las soluciones de este sistema de ecuaciones, que en este caso son los coeficientes estimados por mínimos cuadrados para $\beta_0$ y $\beta_1$ son:

$$\tilde{\beta}_0 = \overline{Y} - \tilde{\beta}_1 \overline{X}$$
$$\tilde{\beta}_1 = \frac{\sum_{i=1}^{n} (X_i - \overline{X})(Y_i - \overline{Y})}{\sum_{i=1}^{n} (X_i - \overline{X})^2}.$$

Donde $\overline{X}$ y $\overline{Y}$ representan a la media aritmética de $X$ y $Y$ respectivamente. En general, $\sigma^2$ no es conocido, pero podemos estimarlo a través de los errores residuales. Este estimador es conocido como Error Residual Estándar y está definido como:

$$ERE = \sqrt{\frac{\sum_{i=1}^{n} \epsilon_i^2}{n-2}}.$$

Para dividir el conjunto de datos en el conjunto de entrenamiento y el conjunto de prueba, comúnmente se utiliza el criterio de utilizar el 80 % del total de los datos para el conjunto de entrenamiento y el 20 % restante para el conjunto de prueba, elegidos de forma aleatoria. En casos particulares, cuando el número de observaciones en el conjunto de datos no es tan numeroso, se

opta por dividir el conjunto de datos en una razón de dos tercios, es decir, la razón del número de datos del conjunto de entrenamiento al conjunto de prueba será de $2:3$ [4]. Para nuestro ejemplo, utilizaremos el criterio 80/20, dado que el conjunto de datos contiene una cantidad considerable de observaciones.

Una vez que tenemos el conjunto de entrenamiento, vamos a verificar que existe una asociación lineal entre $X$ y $Y$. Para esto utilizaremos el coeficiente de correlación muestral $r_{XY}$ que se define de la siguiente forma:

$$r_{XY} = \frac{\sum_{i=1}^{n}(X_i-\overline{X})(Y_i-\overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i-\overline{Y})^2}} = \frac{\sum_{i=1}^{n}X_iY_i-n\overline{XY}}{\sqrt{[\sum_{i=1}^{n}X_i^2-n\overline{X}^2]}\sqrt{[\sum_{i=1}^{n}Y_i^2-n\overline{Y}^2]}}.$$

Además, tenemos que $r_{XY}$ es un estimador de el coeficiente de correlación entre $X$ y $Y$ ($\rho_{XY}$). Por lo tanto, podemos hacer inferencias sobre él, con el objetivo de poder afirmar con cierto nivel de confianza que $\rho_{XY} \neq 0$. Para esto, utilizaremos la aproximación de Fisher [5] como estadístico de prueba, que esta definido por la siguiente ecuación:

$$z = \tfrac{1}{2}\ln(\tfrac{1+r}{1-r}) \sim \mathcal{N}(\tanh^{-1}\rho, \tfrac{1}{n-3}).$$

El intervalo de confianza con nivel de significancia $\alpha$ para $\rho$ esta dado por las soluciones de la siguiente ecuación:

$$\tfrac{1}{2}\ln(\tfrac{1+r}{1-r}) \pm z_{(1-\frac{\alpha}{2})}\sqrt{\tfrac{1}{n-3}} = \tfrac{1}{2}\ln(\tfrac{1+\rho}{1-\rho})$$

donde $z_{(1-\frac{\alpha}{2})}$ es el porcentaje superior de $\frac{\alpha}{2}$ en la distribución normal estándar $\mathcal{N}(0,1)$. El intervalo de confianza lo obtenemos resolviendo la ecuación para $\rho$ en cada uno de los límites. Entonces, en el caso de el conjunto de entrenamiento, tenemos que el coeficiente de correlación muestral es $r_{XY} = 0.7998$, lo cual nos indica que puede existir una asociación positiva entre $X$ y $Y$. Para comprobarlo, podemos calcular el intervalo de confianza para $\rho_{XY}$ con un nivel de significancia de $\alpha = 0.05$ o nivel de confianza del $95\,\%$.

Para este ejemplo, tenemos que el intervalo de confianza es el siguiente:

$$IC_{\rho_{XY}} = (0.7359, 0.8495).$$

Puesto que $r_{XY} = 0.7998 \in IC_{\rho_{XY}}$, tenemos evidencia para argumentar con el $95\,\%$ de confianza, que la asociación entre $X$ y $Y$ es lineal. Para poder continuar con mayor certeza sobre esta suposición, plantearemos el juego de hipótesis $H_0 : \rho_{XY} = 0$ vs. $H_1 : \rho_{XY} \neq 0$ y utilizaremos el siguiente estadístico:

$$z' = \sqrt{n-3}(\tfrac{1}{2}\ln(\tfrac{1+r}{1-r})).$$

Rechazaremos la hipótesis nula $H_0 : \rho_{XY} = 0$ si se cumple que el valor absoluto de $z'$ es mayor o igual que el valor crítico $z_{(1-\frac{\alpha}{2})}$, es decir, si $z' \leq -z_{(1-\frac{\alpha}{2})}$ o $z_{(1-\frac{\alpha}{2})} \leq z'$. Para nuestro ejemplo, haciendo las operaciones algebraicas correspondientes con $n = 160$ y $r_{XY} = 0.7998$, obtenemos el estadístico $z' = 13.75$.

Tenemos que $z_{(1-\frac{0.05}{2})} = z_{(0.975)} = 1.96$. Así, $z_{(1-\frac{\alpha}{2})} = 1.96 \leq 13.75 = z'$, por lo tanto, rechazamos $H_0 : \rho_{XY} = 0$ con un nivel de significancia de $\alpha = 0.05$. Con estos resultados podemos continuar con el estudio del modelo, suponiendo correctamente (con un $95\,\%$ de confianza), que hay una asociación lineal entre $X$ y $Y$.

Ahora podemos plantear un modelo de aprendizaje utilizando regresión lineal simple. Para nuestro ejemplo, tenemos que el modelo para el conjunto de entrenamiento es el siguiente:

$$Y(X) = 6.7082 + 0.0499X.$$

El cual podemos visualizar en la Figura 1.3:

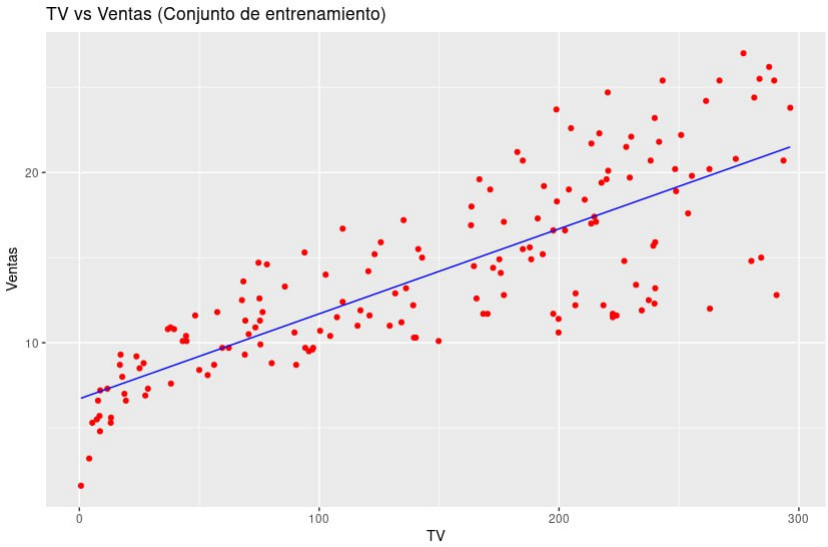


Figura 1.3: Representación gráfica del modelo de regresión generado a partir del conjunto de entrenamiento.

Ahora que contamos con una expresión para representar la relación entre $X$ y $Y$, o bien, entre la cantidad de inversión en publicidad por televisión y las ventas de un producto o servicio, podemos utilizar el conjunto de prueba para empezar a corroborar la precisión del modelo de regresión obtenido. Para esto, comenzamos considerando 10 observaciones del conjunto de prueba, tomadas de forma aleatoria, para poder cotejar con 10 predicciones de los mismos valores de $X$ que resultan de evaluar $Y(X)$. Este cotejo podemos observarlo en la Tabla 1.1 :

| **n** | $\hat{Y}$ | $Y$ **Test set** |
|---|---|---|
| 4 | 14.26 | 18.5 |
| 24 | 18.1 | 15.5 |
| 50 | 10.04 | 9.7 |
| 67 | 8.28 | 9.5 |
| 88 | 12.23 | 16 |
| 107 | 7.95 | 7.2 |
| 126 | 11.06 | 10.6 |
| 145 | 11.5 | 11.4 |
| 181 | 14.52 | 10.5 |
| 195 | 14.17 | 17.3 |

Tabla 1.1: Comparación de las predicciones con el conjunto de prueba.

Como podemos observar, el modelo de regresión lineal hace predicciones cercanas a los valores reales del conjunto de prueba, pero antes de afirmar que el modelo hace predicciones confiables, es necesario verificar las suposiciones iniciales. En el siguiente paso se va a verificar que las suposiciones sobre la variable aleatoria $\epsilon$ son correctas, pues la normalidad de los residuales nos dan pauta a resolver cuestiones sobre los parámetros del modelo.

Es de especial interés la suposición de la normalidad de la variable $\epsilon$, ya que las inferencias que

podamos hacer sobre los parámetros del modelo se basan en este hecho. Una forma para comprobar la normalidad es trazar una gráfica de probabilidad normal de los residuales [6]. Podemos observar el histograma de residuales y la gráfica de probabilidad normal en las figuras 1.4 y 1.5 respectivamente.

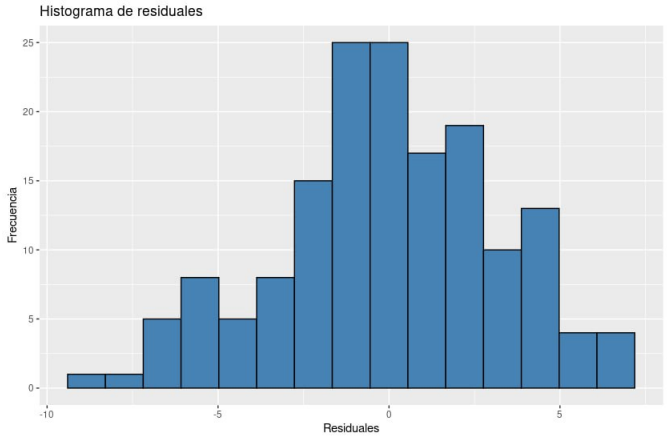


Figura 1.4: Histograma de residuales.

En el histograma podemos observar que la mayoría de los residuales se encuentran alrededor del 0 (naturalmente, pues sabemos que $\mu = 0$ para cualquier $\epsilon_i, i = 1, 2, ..., n$), además, es posible visualizar la normalidad de los residuales.

El objetivo de la gráfica de probabilidad normal es comparar los valores estandarizados de los residuales con valores teóricos de la distribución normal de tal forma que los residuales forman una linea recta aproximada. En este caso, los residuales estandarizados se definen mediante la siguiente ecuación:

$$d_i = \frac{\epsilon_i}{\sqrt{\frac{\sum_{i=1}^{n} \epsilon_i^2}{n-2}}} = \frac{\epsilon_i}{ERE}, i = 1, 2, ..., n.$$

Si tenemos que existen algunos valores que se alejan de esta linea recta, entonces podemos suponer que no se cumple la condición de normalidad en los residuales.

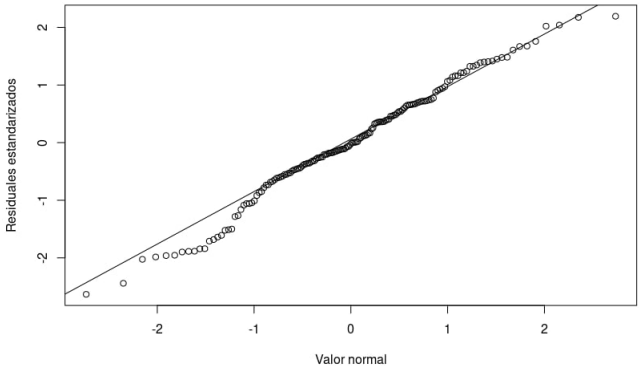


Figura 1.5: Gráfica de probabilidad normal.

Estas gráficas son buenas herramientas para visualizar los residuales y su distribución aproximada y, aunque en la práctica estas visualizaciones son suficientes para suponer normalidad, una gráfica no tiene el mismo nivel de confiabilidad cuando queremos hacer afirmaciones sobre el modelo. Podemos entonces usar la prueba de normalidad Kolmogorov-Smirnov [7] sobre los residuales $\epsilon_i$ para evaluar el siguiente juego de hipótesis:

$H_0$: Los $\epsilon_i$ tienen una distribución normal. vs. $H_1$: Los $\epsilon_i$ no tienen una distribución normal.

La prueba de Kolmogorov-Smirnov nos permite averiguar si un conjunto de observaciones tiene una distribución continua en específico. Para este caso nos interesa que el conjunto de residuales se distribuya de forma normal. El procedimiento de esta prueba consiste en calcular el valor del estadístico $D$, definido como $D = max|F_i(\epsilon) - F_0(\epsilon_{(i)})|$, donde $F_i(\epsilon)$ es la función de distribución empírica y $F_0(\epsilon)$ es la función de distribución normal respecto a $\epsilon_i$. Para determinar si rechazaremos la hipótesis nula, tenemos el valor crítico fijo para esta prueba definido como $D(n, \alpha)$, pues depende del tamaño de la muestra y del nivel de significancia de la prueba. Si $D$ es mayor que el valor crítico $D(n, \alpha)$, rechazaremos la hipótesis de que $\epsilon_i$ tiene una distribución normal.

Cuando se desea hacer inferencias sobre el modelo obtenido, generalmente se utiliza un nivel de significancia de $\alpha = 0.05$, es decir, se afirma con un 95 % de confianza los resultados obtenidos. Dado que el conjunto de entrenamiento cuenta con $n = 160$ observaciones, utilizamos el valor crítico $D(160, 0.05) = \frac{0.886}{\sqrt{160}} = 0.07$. Además, dado el tamaño de la muestra, podemos utilizar una librería de software para calcular el estadístico $D$, que en este caso es $D = 0.0505$. Por lo tanto, no se rechaza la hipótesis nula sobre la normalidad de $\epsilon_i$ con un nivel de significancia de $\alpha = 0.05$.

En otras palabras, podemos afirmar con un 95 % de confianza que los errores $\epsilon_i$ tienen una distribución normal. Hay que destacar que existen otro tipo de pruebas que podrían proporcionar resultados más exactos como la prueba de Shapiro-Wilk o el test de Anderson-Darling, ya que la prueba de Kolmogorov-Smirnov es muy sensible a datos atípicos de los residuales, por lo que esta prueba puede no ser muy útil para muestras muy grandes; sin embargo, la prueba de Kolmogorov-Smirnov junto con la gráfica de probabilidad normal en general son suficientes para aceptar la suposición de normalidad del modelo.

En este momento podemos utilizar el concepto de error estándar para poder verificar que tan cerca están $\tilde{\beta_0}$ y $\tilde{\beta_1}$ de los verdaderos valores de $\beta_0$ y $\beta_1$, asi como si las variables aleatorias $\epsilon_i$ cumplen con las suposiciones iniciales. Tenemos que los errores estándar asociados con $\tilde{\beta_0}$ y $\tilde{\beta_1}$ son:

$$ES(\tilde{\beta_0}) = \sigma^2[\frac{1}{n} + \frac{\overline{X}^2}{\sum_{i=1}^{n}(X_i - \overline{X})^2}].$$
$$ES(\tilde{\beta_1}) = \frac{\sigma^2}{\sum_{i=1}^{n}(X_i - \overline{X})^2}.$$

La utilidad de estas expresiones la encontramos al plantear una prueba de hipótesis sobre los coeficientes $\beta_0$ y $\beta_1$. Como se mencionó antes, tomando el valor de significancia $\alpha = 0.05$, tendremos interés especial por la prueba:

$$H_0 : \text{No hay relación entre } X \text{ y } Y \text{ vs. } H_1 : \text{Hay alguna relación entre } X \text{ y } Y.$$

es decir,

$$H_0 : \beta_1 = 0 \text{ vs. } H_1 : \beta_1 \neq 0.$$

Lo que queremos comprobar es que $\tilde{\beta_1}$, nuestro estimador de $\beta_1$, es suficientemente diferente de cero para poder asegurar algún tipo de relación entre las variables $X$ y $Y$. Para esto utilizamos el estadístico $t$ definido por la siguiente expresión:

$$t = \frac{\tilde{\beta_i}}{ES(\tilde{\beta_1})}.$$

Este estadístico mide el número de desviaciones estándar que hay entre $\beta_0$ y 0. Si tenemos que no hay relación entre $Y$ y $X$, entonces $t$ tendría una distribución $t$ con $n-2$ grados de libertad. Observemos que la prueba de hipótesis para $\beta_1$ es de alguna manera similar a la prueba de hipótesis sobre el coeficiente de correlación $\rho_{XY}$, la interpretación para cada una en el modelo es diferente, pues mientras $\rho_{XY}$ nos habla sobre la manera en la que están asociados $X$ y $Y$, $\beta_1$ nos dice de que forma va a afectar a la variable dependiente $Y$ un cambio en la variable $X_i$.

Para el conjunto de entrenamiento, tenemos que el error estándar de $\beta_1$ depende también de la varianza $\sigma^2$, por lo que usaremos el Error Residual Estándar para estimarla. Por lo que el estadístico $t$ es el siguiente:

$$t = \frac{\tilde{\beta_1}}{ES(\tilde{\beta_1})} = \frac{0.0499}{0.002979} = 16.75.$$

Observemos que el valor crítico para la prueba es $t(n-2, \frac{\alpha}{2}) = t(158, 0.025) = 1.975$. Rechazaremos la hipótesis nula si el valor absoluto del estadístico $t$ es mayor que $t(158, 0.025)$. Como tenemos que $t = 16.75 > 1.975 = t(158, 0.025)$, rechazaremos $H_0$ con un nivel de significancia de $\alpha = 0.05$. Con esta prueba, hemos visto que efectivamente la variable predictora $X$ nos ayuda a explicar el comportamiento de $Y$ con un 95 % de confianza. Más aún, el intervalo de confianza para $\beta_1$ es:

$$IC_{\beta_1} = (\tilde{\beta_1} - ES(\tilde{\beta_1}) \cdot t(n-2, \tfrac{\alpha}{2}), \tilde{\beta_1} + ES(\tilde{\beta_1}) \cdot t(n-2, \tfrac{\alpha}{2})).$$

De modo que en nuestro modelo de aprendizaje utilizando el conjunto de entrenamiento, el intervalo de confianza con $\alpha = 0.05$ para $\beta_1$ es el siguiente:

$$IC_{\beta_1} = (0.044, 0.055).$$

De forma clara vemos que $\tilde{\beta_1} = 0.0499 \in IC_{\beta_1}$. Así, hemos comprobado que además de que existe una asociación lineal entre las variables $X$ y $Y$, también pudimos ver que $X$ contribuye a predecir a $Y$. Pero las predicciones hechas sin soporte teórico son más probables de ser imprecisas, o incluso ser incorrectas. Puesto que el modelo de aprendizaje requiere hacer la división del conjunto de datos, vamos a realizar el intervalo de confianza de algunas predicciones arbitrarias, esto con el fin de comparar el valor estimado por el modelo de regresión y el valor correspondiente del conjunto de prueba, con los límites superior e inferior del intervalo de confianza, nuevamente con el nivel de significancia $\alpha = 0.05$.

Sea $X = X_0$ un valor arbitrario del cual pretendemos hacer una predicción $\hat{Y}(X_0)$. Sabemos que la distribución de $\hat{Y}(X_0)$ [8] es:

$$\hat{Y}(X_0) \sim \mathcal{N}(\beta_0 + \beta_1 X_0, \, \sigma^2_{\hat{Y}(X_0)}).$$

Además, el estimador de $\sigma^2_{\hat{Y}(X_0)}$ con $n-2$ grados de libertad es:

$$S^2_{\hat{Y}} = ERE^2[\tfrac{1}{n} + \frac{(X_0 - \overline{X})^2}{\sum_{i=1}^{n}(X_i - \overline{X})^2}].$$

De esta forma, el estadístico de prueba es:

$$t_0 = \frac{\hat{Y}(X_0) - Y'}{S^2_{\hat{Y}}},$$

donde $Y'$ es el valor específico que se propondrá en la hipótesis nula. Los valores críticos se toman de una distribución $t$ con $n-2$ grados de libertad (que denotaremos como $t = t(\alpha, n-2)$, donde $n$ es el número de elementos en el conjunto de datos), por lo que la expresión que representa el intervalo de confianza con nivel de significancia $\alpha = 0.05$ es:

$$(\hat{Y}(X_0) - S^2_{\hat{Y}} t(\tfrac{\alpha}{2}, n-2), \hat{Y}(X_0) + S^2_{\hat{Y}} t(\tfrac{\alpha}{2}, n-2).$$

Para el ejemplo, los intervalos de confianza de las predicciones que corresponden a las 10 observaciones tomadas del conjunto de prueba se pueden observar en la Tabla 1.2:

| $X_i$ | $Y(X_i)$ | **Límite inferior** | **Límite superior** |
|---|---|---|---|
| 151.5 | 14.26 | 7.85 | 20.68 |
| 228.3 | 18.1 | 11.67 | 24.53 |
| 66.9 | 10.04 | 3.61 | 16.47 |
| 31.5 | 8.28 | 1.83 | 14.72 |
| 110.7 | 12.23 | 5.81 | 18.64 |
| 25 | 7.96 | 1.5 | 14.4 |
| 87.2 | 11.06 | 4.63 | 17.4 |
| 96.2 | 11.5 | 5.09 | 17.92 |
| 156.6 | 14.5 | 8.11 | 20.93 |
| 149.7 | 14.17 | 7.76 | 20.59 |

Tabla 1.2: Intervalos de confianza para 10 predicciones al 95 %.

Como podemos observar, cada uno de los valores del conjunto de prueba se encuentra dentro del intervalo de confianza generado por el modelo de regresión construido utilizando el conjunto de entrenamiento. Aunque esto nos da un indicio de la eficacia del modelo, los intervalos de confianza no son una métrica para evaluar qué tan bueno es el modelo que tenemos. Para la evaluación del desempeño del modelo de regresión, recurrimos al estadístico $R^2$, el cual nos permite evaluar el desempeño de un modelo de aprendizaje independientemente de la escala de $Y$. $R^2$ está definido como:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (Y_i - \tilde{Y}_i)^2}{\sum_{i=1}^{n} (Y_i - \bar{Y})^2}.$$

La expresión $\sum_{i=1}^{n}(Y_i - \tilde{Y}_i)^2$ es la suma total de cuadrados, mientras que $\sum_{i=1}^{n}(Y_i - \bar{Y})^2$ es el error residual estándar. En particular, el estadístico $R^2$ mide la proporción de variabilidad en $Y$ que puede ser explicado por los atributos de $X$. Podemos ver que $0 \leq R^2 \leq 1$, además, un valor cercano a 1 nos diría que la razón entre la suma total de cuadrados y el error residual estándar es muy pequeño, por lo tanto, una proporción grande de la variabilidad en la respuesta puede explicarse con el modelo de regresión. Por el contrario, si tenemos un valor cercano a 0, el estadístico nos indicaría que el modelo no explica de la mejor forma la variabilidad de la respuesta $Y$. En los modelos de regresión lineal simple, el valor de $R^2$ coincide con el cuadrado del coeficiente de correlación $r_{XY}$. Generalmente existen métodos de evaluación de las predicciones en el caso de que éstas sean números reales. El error cuadrático medio ("mean squared error"), es la medida principal que es usada de forma más frecuente en los casos prácticos. También es común tomar la raíz cuadrada de este valor ("root square mean error"), con la finalidad de tener un valor parecido en magnitud respecto a las predicciones. Consideramos a la raíz del error cuadrático medio como una medida de rendimiento del modelo, y está definido como:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (\tilde{Y}_i - Y_i)^2}{n}}.$$

Notemos que la raíz del error cuadrático medio es similar al error residual estándar, con la diferencia de los grados de libertad en el denominador de la expresión.

Otras medidas de rendimiento son el error absoluto medio, el error relativo medio, el error relativo cuadrado, entre otros. Cada una de estas medidas cuentan con propiedades particulares

que son útiles para los modelos propuestos. Para nuestro ejemplo, la raíz de el error cuadrático medio y el valor del estadístico $R^2$ son:

$$R^2 = 0.64,$$
$$RMSE = 3.37.$$

Tomando el estadístico $R^2$, podemos afirmar que el modelo propuesto expica el 65 % de la variabilidad que observamos en la variable dependiente $Y$ explicada por la variable independiente $X$. El 35 % restante corresponde a la variación en $Y$ debido a otros factores que el modelo no incluye. Para el valor del error cuadrático medio $RMSE$, tenemos que en promedio, la diferencia entre los valores previstos y los valores reales de $Y$ es de 3.37 unidades.

Finalmente, dejemos clara la diferencia entre estas dos métricas de evaluación del modelo: el error cuadrático medio nos permite comparar los valores originales del conjunto de prueba con los valores estimados por el modelo propuesto. Este error es la cantidad en la que los valores del conjunto de prueba difieren de los valores estimados por el modelo. El error cuadrático medio será pequeño si las predicciones se acercan a los valores reales, y será grande si para algunas observaciones, la predicción y el valor real difieren de forma considerable. El coeficiente de correlación nos indicará qué tan linealmente está relacionada la variable independiente y la variable cualitativa.

El modelo de regresión lineal simple es un buen escalón para comenzar el estudio del aprendizaje computacional. Aunque en la práctica los problemas suelen ser más complejos en cuanto al número y tipo de variables, el modelo de regresión lineal nos proveé de una idea intuitiva de lo que se trata una tarea de aprendizaje: utilizar un conjunto de datos para entrenar un modelo, y después comprobar su eficacia con un conjunto de prueba, utilizando las herramientas estadísticas con las que contamos para dar validación a las afirmaciones a las que podemos llegar.

## 1.2. Regresión Lineal Múltiple

Cuando observamos el comportamiento de una variable real que depende de dos o más atributos o propiedades de un fenómeno que esta sujeto a observaciones, podemos utilizar la regresión lineal múltiple. En este caso vamos a considerar un modelo lineal respecto a los coeficientes $\beta_i$ de la siguiente forma:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon_i.$$

Como podemos ver, el modelo de regresión lineal múltiple contiene variables $X_i$ correspondientes a cada atributo del fenómeno en cuestión. Usualmente, estas variables pueden ser continuas o categóricas, ya que este modelo suele abstraer problemas más complejos, donde un fenómeno se puede explicar a través de dos o mas variables. Si se realizan $n$ observaciones independientes $Y_1, Y_2, ..., Y_n$, podemos expresar la observación $y_i$ como:

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ... + \beta_k X_{ik} + \epsilon_i,$$

donde $X_{ij}$ es la $j$-ésima variable independiente para la $i$-ésima observación.

Ahora, para saber qué valores utilizar para los coeficientes $\beta_i$, podemos utilizar el método de mínimos cuadrados, aunque en este caso, la función que deseamos minimizar respecto a los coeficientes $\beta_i$ es la siguiente:

$$f(X_1, X_2, ..., X_n; Y) = \sum_{i=1}^{n} [Y_i - (\beta_0 + \beta_1 X_1 + ... + \beta_n X_n)]^2.$$

La forma que tendrá cada coeficiente $\beta_i$ va a cambiar dependiendo del número de variables que se consideren para la construcción del modelo.

Si bien el método de mínimos cuadrados es directo para encontrar los valores mínimos de $\beta_i$, suele no ser utilizado en casos prácticos, ya que ahora, las variables independientes o atributos forman una matriz $X$, de forma que esta matriz debe cumplir ciertas condiciones para que sea computacionalmente viable la estimación de los coeficientes $\beta_i$.

En este sentido, representamos un conjunto de datos dado definiendo a $Y$ como un vector de observaciones, $X$ es una matriz de variables predictoras; por otro lado, sea $\beta$ un vector de parámetros que se estimarán a partir del conjunto de datos, $\epsilon$ un vector de errores y $\hat{1}$ como un vector con componentes 1. Entonces, para el modelo propuesto anteriormente, podemos escribirlo de forma matricial de la siguiente manera:

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ ... \\ y_n \end{bmatrix}, X = \begin{bmatrix} 1 & x_{11} & x_{12} & . & x_{1k} \\ 1 & x_{21} & x_{22} & . & x_{2k} \\ 1 & . & . & . & . \\ 1 & x_{n1} & x_{n2} & . & x_{nk} \end{bmatrix}, \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ ... \\ \beta_k \end{bmatrix}, \epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ ... \\ \epsilon_k \end{bmatrix}.$$

Tenemos entonces que el modelo que representan las $n$ ecuaciones de $Y_i$ como función de $X, \beta$ y $\epsilon$, se puede escribir como:

$$Y = X\beta + \epsilon = \begin{bmatrix} 1 & x_{11} & x_{12} & . & x_{1k} \\ 1 & x_{21} & x_{22} & . & x_{2k} \\ 1 & . & . & . & . \\ 1 & x_{n1} & x_{n2} & . & x_{nk} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ ... \\ \beta_k \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ ... \\ \epsilon_k \end{bmatrix}.$$

Utilizando esta notación para el caso de regresión lineal simple, tenemos las siguientes expresiones:

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ ... \\ y_n \end{bmatrix}, X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ ... & ... \\ x_n & d_2 \end{bmatrix}, \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}, \epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ ... \\ \epsilon_n \end{bmatrix}.$$

Además, recordemos que las ecuaciones normales están dadas por el siguiente sistema:

$$\hat{\beta}_0 n + \hat{\beta}_1 \sum_{i=0}^{n} X_i = \sum_{i=0}^{n} Y_i,$$
$$\hat{\beta}_0 \sum_{i=0}^{n} X_i + \hat{\beta}_1 \sum_{i=0}^{n} X_i^2 = \sum_{i=0}^{n} X_i Y_i.$$

Luego, considerando el producto de matrices entre $X$ y su transpuesta $X^T$ y el producto de $X^T$ con $Y$, tenemos las siguientes matrices:

$$X^T X = \begin{bmatrix} 1 & 1 & ... & 1 \\ x_1 & x_2 & ... & x_n \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ ... & ... \\ 1 & x_n \end{bmatrix} = \begin{bmatrix} n & \sum_{i=0}^{n} X_i \\ \sum_{i=0}^{n} X_i & \sum_{i=0}^{n} X_i^2 \end{bmatrix},$$

$$X^T Y = \begin{bmatrix} 1 & 1 & ... & 1 \\ x_1 & x_2 & ... & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ ... \\ y_n \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{n} Y_i \\ \sum_{i=0}^{n} X_i Y_i \end{bmatrix}.$$

Por lo tanto, las ecuaciones de mínimos cuadrados en forma matricial son:

$$(X^T X)\beta = X^T Y.$$

Y su solución está expresada por:

$$\hat{\beta} = (X^T X)^{-1} X^T Y.$$

Sabemos que la solución de las ecuaciones normales nos proporcionan los estimadores de mínimos cuadrados de $\beta_0$ y $\beta_1$. Como podemos ver, esta solución depende de la matriz inversa $(X^T X)^{-1}$, de modo que surgen restricciones sobre este producto de matrices. Lo ideal sobre el producto $X^T X$ es que el resultado sea una matriz no singular, ya que de esta manera, $\hat{\beta}_0$ y $\hat{\beta}_1$ serán los mejores estimadores de los parámetros $\beta_0$ y $\beta_1$.

Sin embargo, en las tareas de aprendizaje computacional es común contar con un conjunto de datos donde las variables predictoras no forma una matriz singular. Por este impedimento, se opta por un método distinto para estimar los parámetros $\beta_0$ y $\beta_1$. El método más común y que es utilizado en distintos ámbitos del aprendizaje computacional, es el algoritmo de Gradiente de Descenso.

Este algoritmo es uno de los procedimientos fundamentales para minimizar una función diferenciable de varias variables [9]. De forma general, podemos resumirlo de la siguiente manera: dado un punto $x$ y un campo escalar diferenciable y convexo $f(x)$, el algoritmo de gradiente de descenso procede realizando una búsqueda en línea recta con dirección del gradiente negativo normalizado $-\frac{\nabla f(x)}{||f(x)||}$, o de forma equivalente, en la dirección del gradiente $-\nabla f(x)$. Los pasos del algoritmo se encuentran enunciados en el Algoritmo 1. Al parámetro $\lambda$ se le conoce como paso o razón de aprendizaje.

**Algoritmo 1** : Gradiente de descenso

**Paso de inicialización**
Sea $\epsilon > 0$ un escalar de terminación. Se elige un punto de inicio $x_1$.
Sea $k = 1$.
**Paso principal**
Si $||\nabla f(x_k)|| < \epsilon$, detenerse, en otro caso:
    Sea $d_k = -\nabla f(x_k)$ y sea $\lambda_k$ una solución óptima del problema $f(x_k + \lambda d_k), \lambda \geq 0$.
    Sea $x_{k+1} = x_k + \lambda_k d_k$, se reemplaza $k$ por $k + 1$ y se repite el paso principal.

Una representación gráfica de este algoritmo se encuentra en la Figura 1.5. Observemos que la razón de aprendizaje es fundamental en la convergencia del algoritmo, pues un valor grande de $\lambda$ podría impedir que el algoritmo encuentre una solución al pasar de un punto a otro de la función de costo sin encontrar un mínimo, y por el contrario, un valor pequeño de $\lambda$ haría que el algoritmo tarde demasiado en encontrar el punto mínimo de la función.

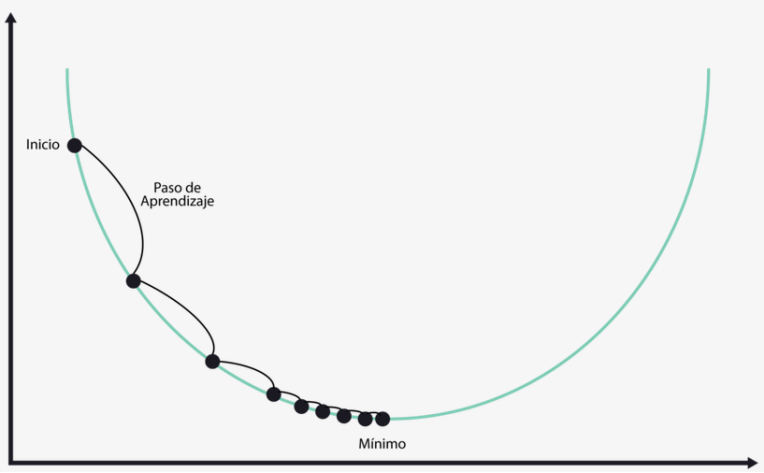


Figura 1.6: Ejemplo del funcionamiento del algoritmo de Gradiente de Descenso.

El algoritmo de gradiente de descenso siempre converge a un punto con gradiente cero. En el caso de regresión lineal múltiple, este algoritmo es utilizado para minimizar una función llamada función de costo, la cual se puede considerar como una variante de la ecuación del caso de mínimos cuadrados y que, además, se considera como una medida de la pérdida sufrida al tomar cualquiera de las decisiones o aciones disponibles, por lo que el objetivo es minimizar estas pérdidas. La función de costo está definida de la siguiente forma:

$$J(\beta) = \frac{1}{2n} \sum_{i=1}^{n} [Y_\beta(X_i) - Y_i]^2.$$

Donde $\beta$ representa al vector de parámetros del modelo de regresión. Tomemos como ejemplo el siguiente conjunto de datos, el cual utilizaremos para comparar los modelos generados por los dos métodos mencionados con anterioridad:

| $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ | $Y$ |
|---|---|---|---|---|---|
| 6.2 | 241 | 0.872 | 0.74 | 245 | 0.201 |
| 5.6 | 250 | 0.86 | 0.77 | 229 | 0.053 |
| 6.5 | 258 | 0.853 | 0.64 | 266 | 0.239 |
| 6.4 | 239 | 0.891 | 0.68 | 251 | 0.242 |
| 5.7 | 260 | 0.888 | 0.81 | 262 | 0.075 |
| 5.8 | 254 | 0.876 | 0.75 | 230 | 0.132 |
| 5.5 | 250 | 0.869 | 0.71 | 228 | 0.053 |
| 6.1 | 241 | 0.86 | 0.76 | 234 | 0.119 |
| 6.1 | 256 | 0.854 | 0.62 | 269 | 0.172 |
| 6.3 | 260 | 0.872 | 0.64 | 240 | 0.171 |
| 6.6 | 249 | 0.877 | 0.69 | 250 | 0.369 |
| 5.7 | 255 | 0.868 | 0.73 | 246 | 0.1 |
| 5.8 | 258 | 0.854 | 0.8 | 261 | 0.105 |
| 6.1 | 260 | 0.879 | 0.77 | 270 | 0.196 |
| 5.8 | 262 | 0.888 | 0.7 | 267 | 0.126 |
| 6.3 | 256 | 0.87 | 0.81 | 246 | 0.216 |
| 6.4 | 254 | 0.862 | 0.76 | 233 | 0.286 |
| 6.8 | 247 | 0.855 | 0.65 | 250 | 0.306 |
| 6.7 | 238 | 0.876 | 0.69 | 249 | 0.403 |
| 6.3 | 264 | 0.884 | 0.71 | 265 | 0.162 |
| 6.4 | 260 | 0.891 | 0.79 | 252 | 0.214 |
| 5.7 | 259 | 0.881 | 0.8 | 245 | 0.287 |
| 5.8 | 244 | 0.863 | 0.76 | 238 | 0.092 |
| 5.4 | 259 | 0.875 | 0.68 | 217 | 0.008 |
| 5.7 | 264 | 0.87 | 0.64 | 276 | 0.102 |

Tabla 1.3: Conjunto de datos utilizado para generar los modelos de regresión utilizando el método de mínimos cuadrados y el algoritmo de gradiente de descenso.

Si utilizamos el método de mínimos cuadrados para estimar los parámetros $\beta_i$, obtenemos el siguiente modelo de regresión:

$$Y_1 = -1.788 + 0.2136X_1 - 0.001X_2 + 0.8978X_3 + 0.1216X_4 + 0.0002X_5.$$

Mientras que, utilizando el algoritmo de gradiente de descenso con una razón de aprendizaje de $\lambda = 0.15$ y con 400 iteraciones, obtenemos el siguiente modelo:

$$Y_2 = 0.177 + 0.0842X_1 - 0.00762X_2 + 0.0106X_3 + 0.00721X_4 + 0.00261X_5.$$

Los modelos $Y_1$ y $Y_2$ se determinaron a partir de los Códigos 1 y 2 respectivamente incluidos en el Apéndice 2 de esta tesis. Observemos que ambos modelos tienen diferentes valores en sus parámetros. Esto sucede debido a que el método de mínimos cuadrados es un cálculo exacto del vector $\beta_i$, mientras que el algoritmo de gradiente de descenso nos proporciona únicamente una aproximación de estos valores. En la práctica, el uso de cada método va a depender de diferentes factores, aunque lo más común es que se prefiera el algoritmo de gradiente de descenso, ya que éste puede encontrar soluciones casi óptimas en menos tiempo que el método de mínimos cuadrados cuando el problema de aprendizaje contiene una gran cantidad de datos. Esta diferencia se puede observar al realizar predicciones, si tomamos la observación $X_1 = 5, X_2 = 240, X_3 = 0.87, X_4 = 0.62, X_5 = 240$, observemos que para cada modelo, obtenemos los siguientes resultados:

$$Y_1 = -0.0535,$$
$$Y_2 = -0.0534.$$

Luego, si la observación es $X_1 = 5, X_2 = 240, X_3 = 0.87, X_4 = 0.62, X_5 = 290$, tenemos lo siguiente:

$$Y_1 = -0.0249,$$
$$Y_2 = -0.0449.$$

La importancia en la diferencia de las predicciones de ambos modelos va a depender de la naturaleza del problema. Aunque la intuición nos incline a pensar que la mejor opción es utilizar el método de mínimos cuadrados, en la práctica, el algoritmo de gradiente de descenso se utiliza con más frecuencia, ya que también es útil en otros ámbitos del aprendizaje computacional, pues la optimización de funciones es un concepto fundamental dentro de éste.

### 1.2.1. Selección de Atributos

Una cuestión importante al construir un modelo de regresión múltiple es saber si las variables elegidas para establecer la relación con el valor que queremos predecir son estadísticamente significativas para el modelo. Es decir, queremos averiguar qué tanto afectan las variables al fenómeno sujeto a análisis.

Para esto tenemos 5 técnicas que comúnmente se utilizan en el aprendizaje computacional para la construcción de modelos de regresión:

- **Modelo saturado**. Este modelo utiliza todas las variables de las que podemos disponer. Se utiliza si ya tenemos conocimiento de que las variables independientes afecta a la variable dependiente. También constituye el paso previo a la técnica de eliminación hacia atrás o "backward elimination".

- **Eliminación hacia atrás o Backwards Elimination**. Consiste en eliminar variables independientes una a una que no son estadísticamente significativas. Para esto utilizaremos el $p$ valor o $p$-value, el cual se puede intepretar como la probabilidad de que la variable $X_i$ tenga un coeficiente $\beta_i$ igual a cero. Es decir, el $p$-value nos proporciona una medida para saber si una variable independiente es relevante estadisticamente en nuestro modelo. Dicho lo anterior, utilizamos el $p$-value como criterio siguiendo los siguientes pasos:

  1. Escoger un nivel de significancia fijo para el modelo (por ejemplo, $S_l = 0.05$).
  2. Ajustar el modelo con todos los posibles predictores.
  3. Considerar el predictor con el $p$ valor más alto. Si $p > S_l$, pasar al siguiente paso, de otra forma, hemos terminado.
  4. Eliminar al predictor seleccionado.
  5. Ajustar el modelo sin la variable antes seleccionada y repetir el paso 3.

- **Selección directa o Forward Selection**. En este método comenzamos con un modelo nulo que contiene solamente a la constante $\beta_0$ pero sin variables predictoras. De la misma forma utilizamos el $p$ valor como criterio de entrada para las variables independientes siguiendo los pasos que se describen a continuación:

  1. Elegir un nivel de significancia fijo para el modelo.
  2. Ajustamos todos los modelos de regresión $Y = \beta_0 + \beta_1 X_i$ para cada $X_i$. Elegimos el modelo que tenga el menor $p$ valor.
  3. Mantenemos esta variable y ajustamos todos los modelos posibles con una variable predictora extra añadida a las que ya tenemos. Es decir, construimos los modelos $Y = \beta_0 + \beta_1 X_k + \beta_2 X_i$, con $k \neq i$.

4. Consideramos al predictor con el menor $p$ valor. Si $p < S_l$, regresamos al paso 3, de otra forma hemos terminado.

- **Eliminación bidireccional**. Este método consiste en la combinación de los métodos de Selección directa y de Eliminación hacia atrás. Los pasos son los siguientes:

  1. Elegimos niveles de significancia para eliminar o agregar variables en el modelo. Definimos entonces $S_e$ y $S_s$
  2. Ajustamos todos los modelos de regresión $Y = \beta_0 + \beta_1 X_i$ para cada $X_i$. Elegimos el que tenga el $p$ valor que cumpla $p < S_e$.
  3. Aplicamos Eliminación hacia atrás utilizando el criterio $p < S_s$ para mantener a las variables en el modelo.
  4. Si ya no se pueden añadir o eliminar variables, el modelo esta completado.

- **Comparación de parámetros**. En este método elegimos un modelo de regresión de acuerdo a un criterio definido con el siguiente procedimiento:

  1. Se elige un criterio de bondad de ajuste.
  2. Construimos todos los posibles modelos de regresión (hay $2^n - 1$ modelos para $n$ variables).
  3. Elegir el modelo que mejor cumple el criterio elegido.

En este trabajo ilustraremos el método de Backwards Elimination o Eliminación hacia atrás, pues además de ser el más intuitivo en tareas de regresión lineal múltiple, es utilizado en otros aspectos del aprendizaje computacional, pues en términos generales se puede aplicar como algoritmo de selección de atributos para modelos de regresión o clasificación, tomando como base alguna métrica representativa del modelo en lugar del $p$-value.

Utilizando el ejemplo anterior, tenemos que el modelo de regresión múltiple basado a partir del método de mínimos cuadrados es el siguiente:

$$Y = -1.7885 + 0.2136X_1 - 0.001X_2 + 0.8978X_3 + 0.1216X_4 + 0.0002X_5.$$

El Código 3 del Apéndice 2 realiza el proceo de Backwards Elimination, donde podemos observar que los $p$-values $p_i$ para cada $\beta_i$ en la primera iteración son los siguientes:

| **Parámetros** $\beta_i$ | **p-values** |
|---|---|
| $\beta_0$ | 0.08 |
| $\beta_1$ | 0 |
| $\beta_2$ | 0.615 |
| $\beta_3$ | 0.402 |
| $\beta_4$ | 0.581 |
| $\beta_5$ | 0.86 |

Tabla 1.4: Tabla de p-values para los parámetros $\beta_i$ en la primera iteración del proceso de Backwards Elimination.

Como el $p$-value para $\beta_5$ es el mayor, tomando como nivel de significancia $S_l = 0.05$, tenemos que $p_5 > S_l$, por lo que se entrena un nuevo modelo omitiendo la variable $X_5$, obteniendo la siguiente ecuación de regresión. En este caso, los $p$-values se presentan en la Tabla 1.5:

$$Y = -1.8147 + 0.2162X_1 - 0.0008X_2 + 0.9131X_3 + 0.1109X_4.$$

| **Parámetros** $\beta_i$ | **p-values** |
|---|---|
| $\beta_0$ | 0.065 |
| $\beta_1$ | 0 |
| $\beta_2$ | 0.622 |
| $\beta_3$ | 0.38 |
| $\beta_4$ | 0.585 |

Tabla 1.5: Tabla de p-values para los parámetros $\beta_i$ en la segunda iteración del proceso de Backwards Elimination.

Ahora, tenemos que $p_2$ es el $p$-value más grande y además, $p_2 > S_l$, de modo que eliminamos esta variable y se genera un nuevo modelo a partir únicamente de las variables $X_1, X_2$ y $X_3$:

$$Y = -1.9718 + 0.2217X_1 + 0.8162X_3 + 0.1274X_4.$$

Con los siguientes $p$-values:

| **Parámetros** $\beta_i$ | **p-values** |
|---|---|
| $\beta_0$ | 0.032 |
| $\beta_1$ | 0 |
| $\beta_3$ | 0.415 |
| $\beta_4$ | 0.539 |

Tabla 1.6: Tabla de p-values para los parámetros $\beta_i$ en la tercera iteración del proceso de Backwards Elimination.

En la siguiente iteración, se procede a remover la variable $X_4$ y se entrena un nuevo modelo, definido como:

$$Y = -1.9612 + 0.2165X_1 + 0.9462X_3.$$

Como se puede intuir, la siguiente variable en ser eliminada es $X_3$, pues su $p$-value no disminuye lo suficiente para superar el umbral definido por el nivel de significancia ($p_3 = 0.328$ en la última iteración). Por lo tanto, el modelo original termina siendo simplificado en un modelo de regresión lineal simple:

$$Y = -1.1298 + 0.2154X_1.$$

La reducción de complejidad en tareas de aprendizaje es posible si dentro del problema se puede prescindir de algunas variables que no se consideren fundamentales cuando se defina el significado de la variable $Y$. La eliminación de cada variable va a quedar determinada por la importancia que tiene cada una en el contexto del problema.

En general, los modelos de regresión dentro del aprendizaje computacional se consideran como un principio básico para realizar predicciones de valores numéricos. El análisis estadístico de estos modelos permite ilustrar de forma minuciosa una tarea de aprendizaje, ya que las comprobaciones de las suposiciones sobre los modelos y las inferencias que podemos hacer con ellos sirven como fundamento para comenzar a construir modelos de aprendizaje utilizando herramientas computacionales, que a su vez introducen al estudio y el desarrollo de modelos más complejos.

# Capítulo 2

# Modelos de Clasificación

El objetivo de la clasificación es aprender a encontrar una función de entradas $x$ y salidas $y$, donde $y \in \{1, ..., c\}$ y $c$ es el número de clases. Si $c = 2$, entonces tenemos el caso de clasificación binaria. Si $c > 2$, entonces tenemos clasificación multiclase.

Una manera de formalizar el problema es con aproximación de funciones [10]. Asumimos que $y = f(x)$ para alguna función $f$ desconocida; el objetivo del aprendizaje es estimar la función $f$ dado el conjunto de entrenamiento $D$, para después hacer predicciones usando $\hat{y} = f(x)$, las cuales consisten en las clases que puede tomar $y$.

## 2.1. Clasificadores Bayesianos

Desde el punto de vista probabilista en aprendizaje supervisado, el problema de clasificación consiste en asignar a un objeto en particular, el cual está descrito por variables independientes $A_1, A_2, ..., A_n$, una de las $m$ clases incluidas en $C = \{c_1, c_2, ..., c_m\}$, tal que se maximice la probabilidad de que dicho objeto pertenezca a una de estas clases dados sus atributos $A_i$. En otras palabras:

$$\arg\max_C P(C|A_1, A_2, ..., A_n).$$

Los clasificadores Bayesianos toman su nombre por la aplicación del Teorema de Bayes en la formulación del problema de clasificación, y se pueden escribir de la siguiente forma:

$$\arg\max_C [\frac{P(C)P(C|A)}{P(A)}],$$

donde $A = \{A_1, A_2, ..., A_n\}$. En este caso llamamos a la probabilidad de la clase, $P(C)$ como la probabilidad a priori, la cual nos interesa encontrar para dar una solución al problema de clasificación. En este sentido, las probabilidades $P(A|C)$ y $P(A)$ se les conoce como la verosimilitud y la probabilidad posterior respectivamente.

Es importante mencionar que la aplicación del teorema de Bayes puede resultar computacionalmente poco factible, pues el número de parámetros en la probabilidad $P(A|C)$ crece de forma exponencial con el número de atributos o variables independientes, de modo que el clasificador Bayesiano funciona mejor para problemas relativamente pequeños respecto al número de atributos. La alternativa más común es hacer la suposición de independencia sobre los atributos dada la clase. En este caso, el clasificador se denomina como clasificador Bayesiano simple o "Naive Bayes".

La suposición de independencia entre los atributos dada la clase de los clasificadores Bayesianos simples (o NBC por Naive Bayes Classifier) se puede interpretar como una independencia condicional entre los atributos, esto es, $P(A_i|A_j, C) = P(A_i|C), \forall i \neq j$. Por lo tanto, el problema de clasificación se simplifica a la siguiente expresión:

$$P(C|A_1, A_2, ..., A_n) = \frac{P(C)P(A_1|C)P(A_2|C)...P(A_n|C)}{P(A)}.$$

Notemos que para este caso, solamente necesitamos conocer la probabilidad a priori $P(C)$ y las $n$ probabilidades condicionales de cada atributo $P(A_i|C)$, los cuales se definen como los parámetros del modelo. Una representación gráfica del clasificador Bayesiano simple se muestra en la Figura 2.1. Esta figura por definición es un grafo acíclico dirigido, y este tipo de grafos forman la base de los modelos probabilistas dentro del aprendizaje computacional.

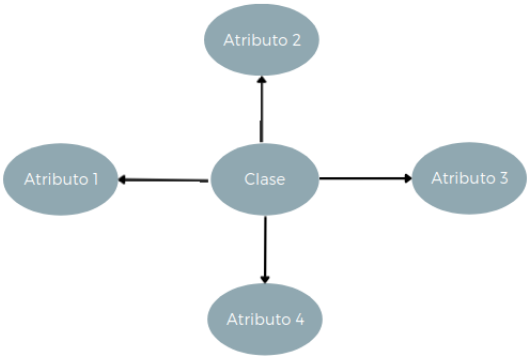


Figura 2.1: Ejemplo de un clasificador bayesiano con variable categórica y cuatro atributos.

Antes de conocer cómo se realiza la tarea de aprendizaje en un NBC, es necesario detallar las formas de evaluación de modelos categóricos, pues en estos casos, no es común contar con una variable como $R^2$ o el coeficiente de correlación $\rho$, pues generalmente, los atributos pueden ser variables categóricas, o se expresan de forma discretizada si son variables continuas, por lo que el cálculo de estas medidas no es viable en la práctica. Entre los diferentes aspectos que se consideran para la evaluación de estos modelos, podemos encontrar los siguientes [11] :

- Precisión: Se refiere al número de ejemplos clasificados correctamente para observaciones nuevas.
- Tiempo de clasificación: Cuánto tiempo tarda el clasificador en hacer una predicción de la clase, una vez que el clasificador ha sido entrenado.
- Tiempo de entrenamiento: Cuánto tiempo tarda el clasificador en aprender de los datos dados.
- Requerimientos de memoria: Cuánto espacio en términos de memoria se requiere para almacenar los parámetros del modelo.
- Claridad: Qué tan entendible es el clasificador para una persona.

Puesto que el análisis en el aprendizaje computacional se basa en las predicciones acertadas del modelo, la métrica de evaluación generalmente usada en el caso de clasificación es la precisión, la cual se define de la siguiente forma:

$$Acc = \frac{k}{n}.$$

Donde $k$ representa el número de predicciones correctas y $n$ es el número de observaciones en el conjunto de datos.

Otra forma de presentar la precisión del modelo es a través de una matriz de confusión. Esta matriz nos permite observar específicamente el número de casos donde el clasificador realizó una predicción correcta o no. Para el caso de clasificación binaria, la matriz de confusión tiene dimensión $2 \times 2$, y sus componentes se definen como:

1. Verdaderos Positivos o True Positives (TP): es el número de casos donde la clase positiva se predice como positiva.
2. Verdaderos Negativos o True Negatives (TN): es el número de casos donde la clase positiva se predice como negativa.
3. Falsos Positivos (FP): es el número de casos donde la clase negativa se predice como positiva.
4. Falsos Negativos (FN): es el número de casos donde la clase positiva se predice como negativa.

Con base en esta notación, la precisión se puede calcular de la siguiente manera:

$$Acc = \frac{TP+TN}{P+N} = \frac{TP+TN}{TP+TN+FP+FN},$$

donde $P$ y $N$ representan los casos positivos y negativos respectivamente.

| | | Clase Real | |
|---|---|---|---|
| | | Positivo | Negativo |
| Clase Prevista | Positivo | TP | FP |
| | Negativo | FN | TN |

Tabla 2.1: Matriz de confusión.

La forma en la que un clasificador Bayesiano puede aprender a partir de un conjunto de datos, donde todos los atributos son variables categóricas, es a través de la estimación de probabilidades a priori $P(C)$, pues nos interesa conocer qué tan probable es que el objeto de estudio en cuestión pertenezca a una clase, dadas las variables independientes o atributos, además de las tablas de probabilidad condicional para cada atributo dada la clase, esto es, $P(A_i|C)$. Estos valores se pueden encontrar de forma frecuentista, comúnmente denominada como máxima verosimilitud. Por ejemplo, los valores para la probabilidad a priori de la clase $C$ estan dados por:

$$P(C = c_i) = \frac{n_i}{n},$$

donde $n_i$ es el número de ocurrencias de la clase $c_i$ y $n$ es el número total de observaciones. Por otro lado, se pueden estimar las probabilidades condicionales de cada atributo de la siguiente manera:

$$P(A_{jk}|C) = \frac{n_{jki}}{n},$$

donde $n_{jki}$ representa el número de veces que el atributo $A_j$ toma el valor $k$ y que pertenece a la clase $i$. El cálculo de estas probabilidades condicionales dan lugar a tablas de probabilidad condicional (CPT por su siglas en inglés), las cuales son matrices que representan a los parámetros del clasificador Bayesiano.

Con estos valores es posible realizar predicciones con ejemplos nuevos, pues si tenemos valores arbitrarios para los atributos, podemos encontrar qué clase tiene mayor probabilidad, la cual se escogerá como la predicción que realiza el modelo.

Tomemos como ejemplo [11] la siguiente base de datos que consiste en variables categóricas que determinan condiciones para jugar golf. En este caso consideraremos atributos nominales y una clase binaria. El conjunto se puede observar en la Tabla 2.2.

En este ejemplo, el objetivo es determinar si se puede o no jugar golf dependiendo de las condiciones del clima. Estas condiciones climáticas están dadas por el panorama meteorológico (Outlook), la temperatura (Temperature), la humedad (Humidity) y el viento (Windy). Notemos que todos los atributos en este ejemplo son categóricos. En caso de que hubiera alguna variable continua en el conjunto de datos, se procedería a discretizar dicha variable para así tener todas

| **Outlook** | **Temperature** | **Humidity** | **Windy** | **Play** |
|---|---|---|---|---|
| Sunny | High | High | False | No |
| Sunny | High | High | True | No |
| Overcast | High | High | False | Yes |
| Rain | Medium | High | False | Yes |
| Rain | Low | Normal | False | Yes |
| Rain | Low | Normal | True | No |
| Overcast | Low | Normal | True | Yes |
| Sunny | Medium | High | False | No |
| Sunny | Low | Normal | False | Yes |
| Rain | Medium | Normal | False | Yes |
| Sunny | Medium | Normal | True | Yes |
| Overcast | Medium | High | True | Yes |
| Overcast | High | Normal | False | Yes |
| Rain | Medium | High | True | No |

Tabla 2.2: Datos del ejemplo de golf.

las variables independientes en el mismo formato. El grafo que representa el clasificador Bayesiano para este problema se puede observar en la Figura 2.2.

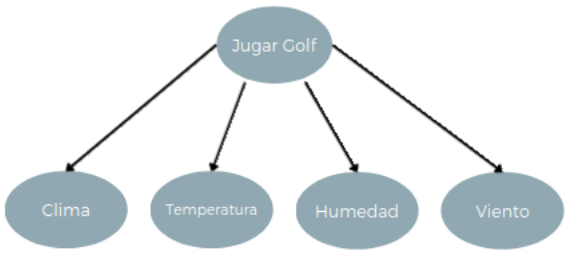


Figura 2.2: Clasificador Bayesiano para el ejemplo de jugar golf.

En primer lugar, observemos las probabilidades de cada clase. En este caso, tenemos dos probabilidades de clase, cuando si se juega al golf ($P(Play = Yes)$) y cuando no se juega ($P(Play = No)$). Entonces, como en 9 de las 14 observaciones tenemos la clase "Yes", la probabilidad a priori es $P(Play = Yes) = \frac{9}{14}$, y de forma directa, $P(Play = No) = \frac{5}{14}$. Procediendo de manera similar para los atributos dada cada clase, se obtienen las tablas de probabilidad condicional o CPTs [18] que se muestran en la Figura 2.3.

Para las probabilidades de los atributos dada la clase, se generan las respectivas tablas de probabilidad condicional tomando en cuenta cada clase y el número de instancias correspondientes en el conjunto de datos. Incluyendo la tabla de la probabilidad a priori, podemos observar las CPTs resultantes en la Figura 2.3. Notemos que cada columna la suma de los valores es igual a 1.

Ya teniendo en cuenta las probabilidades de cada atributo dados los valores de la clase, podemos hacer predicciones para valores arbitrarios de los atributos. Supongamos ahora que queremos hacer una predicción tomando en cuenta los siguientes valores para cada atributo:

$$Outlook = Rain,\ Temperature = High,\ Humidity = Normal,\ Windy = False$$

**Outlook**

| | Yes | No | P(yes) | P(no) |
|---|---|---|---|---|
| Sunny | 2 | 3 | 2/9 | 3/5 |
| Overcast | 4 | 0 | 4/9 | 0/5 |
| Rainy | 3 | 2 | 3/9 | 2/5 |
| **Total** | 9 | 5 | 100% | 100% |

**Temperature**

| | Yes | No | P(yes) | P(no) |
|---|---|---|---|---|
| Hot | 2 | 2 | 2/9 | 2/5 |
| Mild | 4 | 2 | 4/9 | 2/5 |
| Cool | 3 | 1 | 3/9 | 1/5 |
| **Total** | 9 | 5 | 100% | 100% |

**Humidity**

| | Yes | No | P(yes) | P(no) |
|---|---|---|---|---|
| High | 3 | 4 | 3/9 | 4/5 |
| Normal | 6 | 1 | 6/9 | 1/5 |
| **Total** | 9 | 5 | 100% | 100% |

**Wind**

| | Yes | No | P(yes) | P(no) |
|---|---|---|---|---|
| False | 6 | 2 | 6/9 | 2/5 |
| True | 3 | 3 | 3/9 | 3/5 |
| **Total** | 9 | 5 | 100% | 100% |

| Play | | P(Yes)/P(No) |
|---|---|---|
| Yes | 9 | 9/14 |
| No | 5 | 5/14 |
| **Total** | 14 | 100% |

Figura 2.3: Tablas de probabilidad condicional para el ejemplo de golf.

Para realizar la predicción, debemos entonces utilizar la siguiente expresión basándonos en el planteamiento del problema de clasificación Bayesiana:

$$P(C|Rain, High, Normal, False) = \frac{P(C)P(Outlook{=}Rain|C)P(Temperature{=}High|C)P(Humidity{=}Normal|C)P(Windy{=}No|C)}{P(A)}.$$

En particular, el valor de $P(A)$ se toma como una constante de normalización $k$. De modo que, calculando las probabilidades para cada clase ($Play = Yes$ y $Play = No$), tenemos las siguientes probabilidades:

$$P(Play = Yes|Rain, High, Normal, False) = k(\tfrac{9}{14})(\tfrac{3}{9})(\tfrac{2}{9})(\tfrac{6}{9})(\tfrac{6}{9}) = 0.021k$$
$$P(Play = No|Rain, High, Normal, False) = k(\tfrac{5}{14})(\tfrac{2}{5})(\tfrac{2}{5})(\tfrac{1}{5})(\tfrac{2}{5}) = 0.0046k.$$

Puesto que tenemos que $P(C = Yes|A) + P(C = No|A) = 1$, podemos encontrar a $k$ de la siguiente forma:

$$k = \frac{1}{0.021+0.0046} = 39.06.$$

Por lo tanto, las probabilidades de cada clase, dados los atributos del ejemplo son:

$$P(Play = Yes|Rain, High, Normal, False) = 0.021(39.06) = 0.82,$$
$$P(Play = No|Rain, High, Normal, False) = 0.0046(39.06) = 0.18.$$

Finalmente, la predicción se realiza tomando la mayor probabilidad de cada clase, de modo que en este ejemplo, dados los atributos $Rain, High, Normal$ y $False$, la clase que obtiene la mayor probabilidad es $Play = Yes$, por lo que la predicción en este caso sería la clase $Play = Yes$.

Para evaluar el rendimiento de un clasificador Bayesiano, dividimos el conjunto de datos en dos particiones de manera aleatoria como lo hicimos anteriormente: el 80 % de los datos se asignan al conjunto de entrenamiento y el resto al conjunto de prueba. Para este ejemplo obtenemos los conjuntos de datos de las Tablas 2.3 y 2.4. Aunque esta división del conjunto de datos no está hecha a la razón $2:3$ como se mencionó con anterioridad, utilizar el 80 % de los datos también es una medida habitual.

| **Outlook** | **Temperature** | **Humidity** | **Windy** | **Play** |
|---|---|---|---|---|
| Sunny | High | High | False | No |
| Sunny | High | High | True | No |
| Overcast | High | High | False | Yes |
| Rain | Medium | High | False | Yes |
| Rain | Low | Normal | False | Yes |
| Overcast | Low | Normal | True | Yes |
| Sunny | Low | Normal | False | Yes |
| Rain | Medium | Normal | False | Yes |
| Sunny | Medium | Normal | True | Yes |
| Overcast | Medium | High | True | Yes |

Tabla 2.3: Conjunto de entrenamiento para ejemplo de golf.

| **Outlook** | **Temperature** | **Humidity** | **Windy** | **Play** |
|---|---|---|---|---|
| Rain | Low | Normal | True | No |
| Sunny | Low | Normal | False | Yes |
| Overcast | High | Normal | False | Yes |
| Rain | Medium | High | True | No |

Tabla 2.4: Conjunto de prueba para ejemplo de golf.

El objetivo ahora es encontrar las predicciones del conjunto de prueba usando el conjunto de entrenamiento para entrenar el clasificador Bayesiano. Para esto el cálculo de la probabilidad a priori $P(C)$ y las probabilidades $P(A_i|C)$ se realizan en el conjunto de entrenamiento. Una vez encontrados los parámetros (las tablas de probabilidad condicional), se realizan las predicciones utilizando los atributos del conjunto de prueba, para finalmente verificar si la predicción ha sido la correcta.

Para el ejemplo propuesto, si las clases de la variable Play son $c_1 = Yes$ y $c_2 = No$, entonces las predicciones sobre el conjunto de prueba son las siguientes:

$$\begin{aligned}
\arg\max_C [\tfrac{P(C)P(C|Rain,Low,Normal,True)}{P(A)}] &= c_1 \\
\arg\max_C [\tfrac{P(C)P(C|Sunny,Low,Normal,False)}{P(A)}] &= c_1 \\
\arg\max_C [\tfrac{P(C)P(C|Overcast,High,Normal,False)}{P(A)}] &= c_1 \\
\arg\max_C [\tfrac{P(C)P(C|Rain,Medium,High,True)}{P(A)}] &= c_2.
\end{aligned}$$

Como podemos ver, una predicción resultó en un Falso Positivo, un Verdadero Negativo y dos Verdaderos Positivos. Utilizando estos resultados, tenemos la siguiente matriz de confusión:

| | | Clase Real | |
|---|---|---|---|
| | | Yes | No |
| Clase Prevista | Yes | 2 | 1 |
| | No | 0 | 1 |

Tabla 2.5: Matriz de confusión para el ejemplo de golf.

Finalmente, podemos calcular la precisión del modelo con los datos recolectados del entrenamiento del clasificador Bayesiano:

$$Acc = \frac{TP+TN}{TP+TN+FP+FN} = \frac{2+1}{2+1+1+0} = \frac{3}{4} = 0.75.$$

Los clasificadores Bayesianos son útiles en problemas relativamente simples. Sus mayores ventajas es que no se requiere una gran cantidad de parámetros, lo cual reduce los requerimientos de memoria, además del bajo costo computacional para el cálculo de probabilidades. Su mayor desventaja radica en que no es muy útil para problemas con más atributos que datos, o también puede reducirse la eficacia del modelo si la suposición de independencia condicional de los atributos no se cumple.

## 2.2. Regresión Logística y Redes Neuronales

El modelo de regresión logística es un modelo de probabilidad lineal donde la variable dependiente $Y$ es binaria, es decir, es una variable categórica con dos posibles valores, los cuales generalmente se designan como $Y = 1$ y $Y = 0$, donde cada valor representa una clase de la variable. Además, la probabilidad de que la variable $Y$ pertenezca a la clase 1 se puede escribir como $P(Y = 1) = \pi$ y de la misma forma, $P(Y = 0) = 1 - \pi$. El modelo de regresión logística resulta útil para realizar tareas de clasificación donde los atributos $X$ pueden ser continuos o discretos.

Teniendo en cuenta un conjunto de datos con $n$ observaciones, para cada $i$-ésimo elemento, escribimos a $Y_i$ como una variable de Bernoulli con la siguiente distribución de probabilidad:

$$P(Y_i = y_i) = \pi_i^{y_i}(1 - \pi_i)^{1-y_i},\ y = 0, 1.$$

El análisis de regresión logística asume [12] la relación entre las probabilidades $\pi_i$ y las variables $x_i$ del mismo elemento descrito por la función logística:

$$\pi_i = \frac{1}{1+e^{-(\beta_0+\beta_1 x_i)}}.$$

Notemos que el rango de la función logística es $(0, 1)$, de modo que es útil como modelo probabilístico. Si consideramos un modelo lineal para $\pi$ y un solo atributo $X$, podemos considerar la siguiente expresión:

$$\pi(X) = \beta_0 + \beta_1 X.$$

Puesto que el rango del lado derecho son todos los números reales, en la práctica podemos utilizar la transformación Logit, la cual es una función $G$ definida de la siguiente forma:

$$G(z) = \frac{e^z}{1+e^z}.$$

La transformación Logit nos permite además abordar algunos problemas que se presentan en el modelo de probabilidad lineal para la estimación de $E(Y|X)$, es decir, el valor esperado de una clase $Y$ dados los atributos $X$. Algunas de estas complicaciones se enuncian a continuación:

- Heterocedasticidad. La estructura del modelo de probabilidad lineal es la siguiente:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i,$$

donde $x_i$ es un vector de atributos y $\epsilon_i$ es el término de error. Puesto que $y_i$ puede tomar solo valores entre 0 y 1, tenemos que la esperanza condicional de $y_i$ dado $x$ es la probabilidad de $y = 1$ dado $x$:

$$E(y_i|x_i) = 1 \times P(y_i = 1|x_i) + 0 \times (Py_i = 0|x_i) = P(y_i = 1|x_i).$$

Por lo que el modelo es:

$$P(y_i = 1|x_i) = x_i\beta_1.$$

Luego, puesto que $Y$ es una variable Bernoulli, la varianza condicional de $y$ dado $x_i$ es:

$$Var(y|x_i) = P(y = 1|x_i)[1 - P(y = 1|x_i)] = x_i\beta(1 - x\beta).$$

Observemos que esta expresión depende de los valores de $x$ y no es constante, por lo que los errores del modelo estarían sesgados y el estimador de mínimos cuadrados resultaría inapropiado.

- Normalidad. El análisis del modelo de regresión logística se basa principalmente en una distribución binomial.

Tomando en cuenta estas limitaciones, vamos a considerar los modelos de la forma:

$$P(y = 1|x) = G(\beta_0 + \beta_1 x_i + ... + \beta_k x_k).$$

Donde $G$ es una función con rango $(0, 1)$. La transformación Logit, ilustrada en la Figura 2.4, nos sirve en el modelo de regresión logística ya que el rango de la función puede representar las probabilidades de que una observación o ejemplo nuevo dentro del modelo pertenezca a una de las dos clases de $Y$.

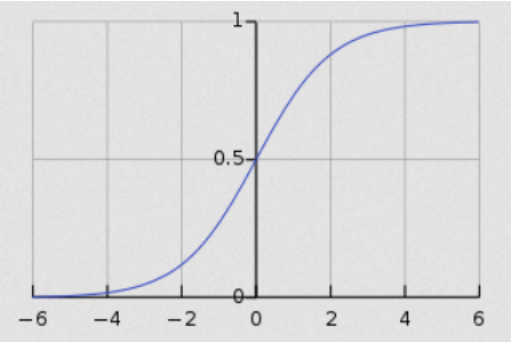


Figura 2.4: Gráfica de la función logística.

El modelo de regresión logística con un solo atributo $x$ específicamente se puede escribir como:

$$\pi(x) = \frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}} = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}}.$$

Despejando $e^{\beta_0+\beta_1 x}$, tenemos que:

$$e^{\beta_0+\beta_1 x} = \frac{\pi(x)}{1-\pi(x)}.$$

Luego, aplicando logaritmo natural en ambos lados de la igualdad, obtenemos:

$$\ln(\tfrac{\pi(x)}{1-\pi(x)}) = \beta_0 + \beta_1 x.$$

Podemos observar que la transformación Logit es lineal en sus parámetros, además, $\pi(x)$ es un número real para cualquier $x$.

La función logística $\pi(x) = \frac{1}{1+\exp[-(\beta_0+\beta_1 x)]}$, es utilizada principalmente en el aprendizaje computacional como función de activación de redes neuronales. Las redes neuronales artificiales (ANNs por sus siglas en inglés), es una de las primeras técnicas que surgieron con el objetivo de que una computadora aprenda con datos recibidos. Las redes neuronales están compuestas por unidades básicas de procesamiento, denominadas neuronas, las cuales se pueden representar como una función de entradas $x_i$ que calculan la combinación lineal de estas entradas con magnitudes o pesos $w_i$, que ajustan la contribución de cada entrada $x_i$ a la salida de la red neuronal. Se puede definir mediante la siguiente función $z$:

$$z_j(x_i) = \sum_i w_{ij} x_i.$$

El caso más simple de una red neuronal artificial es el perceptrón, el cual es un clasificador binario definido por las siguiente función:

$$y = f(z) = \begin{cases} 1, & z + b > 0 \\ 0, & z + b \leq 0 \end{cases}.$$

Donde $b$ representa el sesgo que controla el umbral de decisión y se determina de manera arbitraria dependiendo la naturaleza del problema. Si tenemos que $Y$ es una variable binaria, entonces la función logística es la mejor opción, pues podemos representar la probabilidad de que $Y$ pertenezca a una de sus clases de la siguiente forma:

$$P(Y = 1) = \pi(\sum_i^n w_i x_i).$$

En la Figura 2.5 tenemos el caso en que se tienen dos o más neuronas apiladas en forma de capas, donde la salida de una capa es la entrada de la siguiente capa. Se puede observar que en el nodo principal, la función $\Phi$ es la función de activación que introduce no linealidad a la salida del perceptrón. Este caso es conocido como red neuronal multicapa o perceptrón multicapa. Podemos observar una representación gráfica en la Figura 2.6. Hay que destacar que esta red va a describir mejor los atributos que representan las entradas $x_i$ entre más capas ocultas existan entre la capa de entrada y la capa de salida; usualmente estas redes neuronales multicapas son utilizadas en problemas de clasificación más complejos, pues cada capa analiza y detecta los atributos que conforman las observaciones del conjunto de datos. Estos modelos con frecuencia se utilizan para evaluar entradas relacionadas a tareas que involucran el procesamiento de una señal, como por ejemplo, audio, imágenes, video, etc.

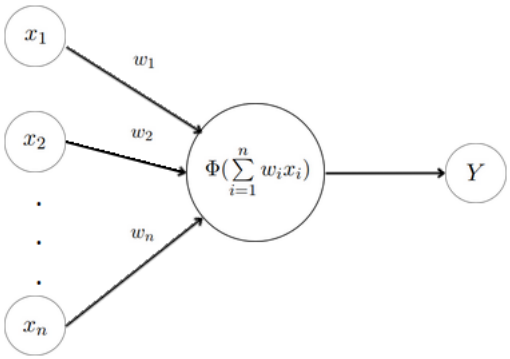


Figura 2.5: Representación gráfica de un perceptrón.

En estos modelos, en la última capa es donde se realiza la tarea final de clasificación, por lo que es posible utilizar la función logística para este último paso como función de activación si se desea predecir una probabilidad y se tiene un problema de clasificación binaria. Una función de activación en una red neuronal permite decidir si una neurona va a ser activada o no de acuerdo a la salida de la función elegida como función de activación. En la práctica se han encontrado funciones de activación que permiten tener un mejor rendimiento a la red neuronal, como lo son la funcion tangente hiperbólica, definida por:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}},$$

o la función ReLU, que es una abreviatura de Rectified Linear Unit y esta definida como

$$f(x) = max(0, x).$$

Además, si se plantea un problema de clasificación multiclase, donde la variable $Y$ puede tomar cualquiera de las $n$ clases que defina el problema y $X$ es el conjunto de atributos, contamos con la función softmax, definida por:

$$f(X)_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}},$$

que permite calcular las probabilidades de que una observación pertenezca a alguna de las $n$ clases que contenga el problema. La capa oculta es donde los datos de entrada son transformados a través de las funciones de activación, de tal forma que en cada capa se realiza una abstracción cada vez más compleja de los datos, para finalmente dar un valor de salida $Y$ que determinará la predicción sobre la tarea que se estudie.

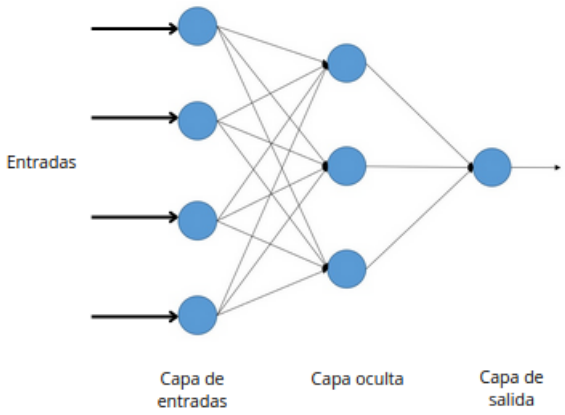


Figura 2.6: Representación gráfica de un perceptrón multicapa.

En general, las redes neuronales artificiales permiten realizar tareas de aprendizaje más complejas debido a su nivel de abstracción del problema, además de ser la base de nuevas vertientes dentro el aprendizaje computacional como lo es el aprendizaje profundo (deep learning) o el aprendizaje por refuerzo en tareas de visión computacional. El proceso de aprendizaje de una red neuronal utiliza conceptos similares vistos en esta tesis, como el proceso de optimización de los pesos $w_i$, que se realiza con una variante del algoritmo de gradiente de descenso (se le conoce como gradiente de descenso estocástico). Debido a lo anterior, el uso de redes neuronales es un proceso computacionalmente complejo, pues el proceso de entrenamiento de una red neuronal requiere demasiado tiempo, por lo que se recurre a algoritmos de optimización numérica o a soluciones que involucran hardware adicional, como las unidades de procesamiento gráfico, o GPUs.

### 2.2.1. Relación entre los Clasificadores Bayesianos Simples y los Perceptrones.

Existe una relación interesante entre los clasificadores Bayesianos simples y el perceptrón. Como se mencionó anteriormente, el clasificador Bayesiano simple estima la probabilidad de la clase $C$, a partir del teorema de Bayes, suponiendo que los atributos $A = \{a_1, a_2, ..., a_n\}$ cumplen con la propiedad de independencia condicional, es decir:

$$P(C|a_1, a_2, ..., a_n) = \frac{P(C)P(a_1|C)P(a_2|C)...P(a_n|C)}{P(A)},$$

donde $P(A)$ se considera como una constante de normalización, de modo que tenemos:

$$P(C|A) \sim P(C)P(a_1|C)P(a_2|C)...P(a_n|C).$$

De modo que podemos enunciar el problema de encontrar la clase con la máxima probabilidad, es decir, el problema de clasificación, como:

$$\arg\max_C [P(C)P(C|a_1, a_2, ..., a_n)].$$

Luego, aprovechando la propiedad de monotonía de la función logaritmo con respecto a $P(C|A)$, podemos escribir la expresión anterior como:

$$\arg\max_C[\log P(C) + \log P(a_1|C) + ... + \log P(a_n|C)].$$

Observemos que el término que nos interesa maximimzar es $Y(C) = \log P(C) + \sum_{i=1}^{n} \log P(a_i|C)$; entonces, si tenemos un problema de clasificación binaria, los valores de $C$ serán las clases $c_1$ y $c_2$, de modo que si evaluamos estas clases en la función $Y(C)$, tenemos:

$$Y(c_1) = \log P(c_1) + \sum_{i=1}^{n} \log P(a_i|c_1),$$
$$Y(c_2) = \log P(c_2) + \sum_{i=1}^{n} \log P(a_i|c_2).$$

Debido a lo anterior, si tenemos que $Y(c_2) > Y(c_1)$, entonces se decidirá por la clase $c_2$, y elegiremos $c_1$ de otra forma. Si observamos con detenimiento esta desigualdad, podemos ver que la siguiente expresión es equivalente:

$$y = \begin{cases} c_2, & \sum_{i=1}^{n} \log \frac{P(a_i|c_2)}{P(a_i|c_1)} + \log \frac{P(c_2)}{P(c_1)} > 0, \\ c_1, & \sum_{i=1}^{n} \log \frac{P(a_i|c_2)}{P(a_i|c_1)} + \log \frac{P(c_2)}{P(c_1)} \leq 0. \end{cases}$$

Ahora, tomando el caso del perceptrón, recordamos que su definición es la siguiente:

$$y = f(z) = \begin{cases} 1, & z + b > 0, \\ 0, & z + b \leq 0. \end{cases}$$

Podemos observar que ambas expresiones son similares. Esto es porque se está calculando una combinación lineal de las entradas $x_i$ (o atributos $a_i$) con pesos $w_i$, considerando al sesgo $b$ como la probabilidad a priori $P(C)$. Se ha demostrado que la separabilidad lineal está relacionada con la independencia condicional, es decir, los hiperplanos pueden clasificar información de manera óptima cuando las entradas tienen la propiedad de independencia condicional [11].

Por lo tanto, un perceptrón con la función logística $f(z) = \frac{1}{1+e^{-z}}$ puede calcular la probabilidad posterior cuando las entradas de la neurona son proporcionales a la probabilidad de la entrada en cuestion [11], con pesos:

$$w_i = \log \frac{P(x_i|c_2)}{P(x_i|c_1)},$$

y el sesgo $b$ expresado como:

$$b = \log \frac{P(c_2)}{P(c_1)}.$$

Por lo tanto, si se requiere resolver un problema más complejo donde los datos no son condicionalmente independientes (o linealmente separables), es necesario utilizar modelos más complejos, como las arquitecturas multicapas de las redes neuronales.

Teniendo en cuenta lo expuesto anteriormente, podemos darnos cuenta cómo los conceptos como el Teorema de Bayes y la independencia condicional nos permiten abordar estos problemas de clasificación, los cuales son los más frecuentes dentro del contexto de la computación, por lo que el estudio de los clasificadores Bayesianos simples y la regresión logística son conceptos accesibles y puntos importantes de partida para iniciar el estudio de esta área de la computación. Además, la comprensión y dominio de este tema de aprendizaje permite adentrarse a técnicas y algoritmos para tareas más complicadas como hacer predicciones utilizando redes Bayesianas, la detección de objetos de interés en una imagen o el uso de máquinas de soporte vectorial para el reconocimiento de escritura a mano.

# Capítulo 3

# Evaluación de modelos: Prueba de Friedman

Dentro del aprendizaje computacional, es común querer saber cuál es el modelo que mejor se ajusta al conjunto de datos con el que se cuenta. En general, se pueden plantear diferentes modelos dado un conjunto de datos, y estos modelos proporcionarán predicciones con cierta precisión. Una tarea importante al desarrollar o implementar un modelo de aprendizaje, es saber si la elección del modelo es la más adecuada y evaluar su desempeño. Esta evaluación nos dará información necesaria para poder decidir cuál es el modelo con mejor funcionamiento. En la práctica es común abordar esta tarea (ya sea teniendo un problema de regresión o de clasificación) utilizando el modelo y sus variantes con distintos conjuntos de datos, para después comprobar cuál es el que tiene mejor rendimiento.

Los datos recolectados de cada modelo tendrán como base las métricas utilizadas para evaluar la eficacia del modelo (como la precisión o accuracy en clasificación o $R^2$ en regresión), de modo que cada modelo se puede identificar como un tratamiento para los datos, por lo que podemos tomar en cuenta un diseño experimental aleatorio por bloques, donde cada bloque esta conformado por un conjunto de datos y cada tratamiento sería representado por el modelo definido a partir de cada conjunto de datos. De esta manera, nos interesa saber si hay diferencia alguna entre los modelos en cuestión; por lo tanto, podríamos plantear el siguiente juego de hipótesis, donde cada $\tau_i$ representa un modelo de aprendizaje:

$$H_0 : \tau_1 = \tau_2 = ... = \tau_k \text{ vs. } H_1 : \text{no todos los } \tau_k \text{ iguales.}$$

Este caso es similar al procedimiento para probar si los efectos de dos tratamientos no son iguales [13], por lo que el cálculo del estadístico $S$ de Friedman nos puede asistir en la decisión de escoger el modelo con mejor rendimiento. Sea $N$ el número de bloques o conjuntos de datos y sea $k$ el número de modelos de aprendizaje. Sea $r_i^j$ el rango jerárquico del $j$-ésimo algoritmo de aprendizaje en el $i$-ésimo conjunto de datos, con $j \leq k$ e $i \leq N$. La prueba de Friedman compara los promedios de los rangos de los algoritmos $R_j = \frac{1}{N} \sum_i r_i^j$. Entonces, dado que queremos saber si los modelos son equivalentes, el estadístico de Friedman se define como:

$$S = \frac{12N}{k(k+1)} [\sum_{i=1}^{k} R_i^2 - \frac{k(k+1)^2}{4}],$$

donde $S$ tiene una distribución $\chi^2$ con $k-1$ grados de libertad. Se ha mostrado [14] que el estadístico $S$ es conservador, de modo que se definió un estadístico derivado de $S$:

$$F_F = \frac{(N-1)S}{N(k-1)-S},$$

el cual se aproxima a una distribución $F$ con $k-1$ y $(k-1)(N-1)$ grados de libertad.

Para ilustrar el uso de la prueba de Friedman para la evaluación de modelos, supongamos que tenemos 4 clasificadores, que se probaron con 5 distintos conjuntos de datos. También supongamos que se recolectó la información respecto a sus métricas de rendimiento representada en la Tabla 3.1.

| | **Clasificador A** | **Clasificador B** | **Clasificador C** | **Clasificador D** |
|---|---|---|---|---|
| **Base de datos 1** | 0.8992 | 0.845 | 0.7707 | 0.7429 |
| **Base de datos 2** | 0.9253 | 0.8625 | 0.8941 | 0.6 |
| **Base de datos 3** | 0.8108 | 0.9210 | 0.911 | 0.8917 |
| **Base de datos 4** | 0.9974 | 0.9625 | 0.967 | 0.975 |
| **Base de datos 5** | 0.7904 | 0.655 | 0.555 | 0.535 |

Tabla 3.1: Resultados de la evaluación de 4 clasificadores diferentes.

Utilizaremos un nivel de confianza del 95 % (es decir, $\alpha = 0.05$) para probar entonces el siguiente juego de hipótesis:

$$H_0 : \tau_1 = \tau_2 = \tau_3 = \tau_4 \text{ vs } H_1 : \tau_1 \neq \tau_2 \neq \tau_3 \neq \tau_4.$$

Donde $\tau_i, i = 1, 2, 3, 4$ representan a cada clasificador utilizado. Con esta prueba pretendemos evaluar que tan parecidos son los clasificadores utilizados, de modo que rechazaremos la hipótesis nula $H_0$ si $F_F > F(k-1, (k-1)(N-1), \alpha)$, donde $F$ es el valor crítico de la distribución $F$ con $k-1$ grados de libertad en el numerador, $(k-1)(N-1)$ grados de libertad en el denominador y nivel de confianza $\alpha$.

En este caso, tenemos $N = 5$ conjuntos de datos, $k = 4$ clasificadores y $R_i$ es el rankeo promedio de cada clasificador. Este rankeo se encuentra enumerando por filas cada precisión respecto a cada clasificador, de modo que $R_1 = 1.2, R_2 = 2.4, R_3 = 3$ y $R_4 = 3.4$, entonces:

$$\begin{aligned} S &= 8.28 \\ \implies F_F &= 4.92. \end{aligned}$$

Con el nivel de significancia $\alpha = 0.05$, el valor crítico de $F$ es $F(3, 12, 0.05) = 3.49$, por lo que $F_F = 4.92 > 3.49 = F(3, 12, 0.05)$, de esta forma, rechazamos $H_0$, es decir, podemos afirmar con un 95 % de confianza que al menos un clasificador es diferente de los demás.

Debido a que se rechazó $H_0$, el siguiente paso sería realizar una prueba de Nemenyi para determinar cuál es el clasificador que tiene mejor rendimiento. La prueba de Nemenyi es una prueba post hoc, que sirve para mostrar las diferencias entre los rendimientos de los modelos de aprendizaje y se utiliza cuando se va a comparar a los modelos entre si.

En este paso necesitamos calcular las diferencias entre cada $R_i$. Si alguna de estas diferencias es mayor o igual al estadístico de prueba (denominado distancia crítica o CD), podemos afirmar con un nivel de significancia $\alpha$ que los clasificadores con mayor diferencia son diferentes entre si. La distancia crítica se calcula de la siguiente forma:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}},$$

donde $q_\alpha$ se obtiene de la Tabla 3.2 de valores críticos.

En este ejemplo, tenemos que $k = 4$, por lo que $q_\alpha = 2.569$. Entonces:

$$CD = 2.569 \sqrt{\frac{4(4+1}{6(5)}} = 2.098.$$

Tabla 3.2: Tabla de valores críticos para la prueba de Nemenyi [19].

| **k** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|
| **Nemenyi** | 1.96 | 2.343 | 2.569 | 2.728 | 2.85 | 2.949 | 3.031 | 3.102 | 3.164 |

Por lo tanto, el valor absoluto de las diferencias entre el ranking de cada clasificador son las siguientes:

$$|R_1 - R_2| = 1.2$$
$$|R_1 - R_3| = 1.8$$
$$|R_1 - R_4| = 2.2$$
$$|R_2 - R_3| = 0.6$$
$$|R_2 - R_4| = 1$$
$$|R_3 - R_4| = 0.4$$

Observemos que la diferencia $|R_1 - R_4| = 2.2 > 2.098 = CD$ y las demás diferencias son menores al valor de $CD$, entonces, con la prueba de Friedman y posteriormente con la prueba de Nemenyi, podemos afirmar que el mejor clasificador es el clasificador A con un nivel de significancia de $\alpha = 0.05$.

En general, van a existir diferentes opciones para realizar alguna prueba estadística que nos permita evaluar la selección de modelos de aprendizaje (como la prueba de Wilcoxon o la prueba de McNemar) o incluso se puede comparar el rendimiento de los modelos haciendo gráficas simultáneas que representen la métrica de interés. La prueba de Friedman, y la consecuente prueba de Nemenyi, son de las pruebas más accesibles para la selección de modelo, pues únicamente depende de las métricas de evaluación de los modelos, por lo que es posible realizarlas en la práctica cuando se cuenta de antemano con los modelos propuestos a partir de los datos con los que se cuenta, o si se pretende evaluar variantes en los algoritmos o en los parámetros de los modelos, para determinar si existe una diferencia significativa entre ellos.

# Capítulo 4

# Conclusiones

El estudio del aprendizaje computacional inicia desde que aprendemos en los cursos básicos de probabilidad y estadística conceptos como el Teorema de Bayes o regresión lineal. Es importante además, ser conscientes de que esta área de la computación es cada vez más común en las empresas públicas y privadas, pues la automatización de procesos y los pronósticos son algo fundamental en cualquier organización, de modo que el conocer cómo se llevan a cabo estas tareas da una ventaja a los estudiantes interesados.

Los modelos de regresión, en general, son considerados como el punto de partida para el estudio del aprendizaje computacional, por lo que suelen ser omitidos debido a que los conceptos de regresión lineal simple se toman por conocidos cuando se comienza el estudio de la literatura relacionada al aprendizaje computacional. Las inferencias que se pueden hacer sobre los elementos de los modelos de regresión ayudan a comprender la importancia de tener un modelo que pueda abstraer las características del problema en cuestión; además, ayudan a comprender el análisis de modelos para tareas distintas a la regresión, pues éstos modelos cuentan con sus propios parámetros que van a representar en algun grado las variables en la predicción final.

Por otro lado, las tareas de clasificación son más comunes dentro del estudio del aprendizaje computacional, ya que es más sencillo calcular clases que un número real, debido a su representación más accesible en las computadoras. Es por esto que usualmente las tareas de regresión se convierten en tareas de clasficación discretizando los atributos y la variable de respuesta del problema, ya que el tiempo para tomar una decisión utilizando el problema generalmente es limitado, por lo que se recurre a modelos que funcionen en la menor cantidad de tiempo. En este sentido, la aplicación del teorema de Bayes y la regresión logística son herramientas accesibles que pueden ser utilizadas en los cursos como un primer acercamiento a sus aplicaciones con algunos ejemplos sobre aprendizaje computacional. Además, con la incorporación de temas de otras áreas, como el álgebra lineal o la teoría de grafos, se puede observar de una manera tangible cómo estas áreas de las matemáticas se combinan para dar lugar a esta disciplina que es parte de la computación.

Es importante mencionar que la elección de modelos en un problema de aprendizaje son una parte importante del aprendizaje computacional, pues estas estrategias de evaluación van a permitir seleccionar el mejor modelo para el problema en cuestión, proporcionando argumentos estadísticos para validar esta selección.

El aprendizaje computacional, en la época reciente, ha avanzado a pasos agigantados a causa de la necesidad de automatizar tareas y al desarrollo de algoritmos nuevos que permiten ahorrar tiempo y trabajo dentro de alguna organización. También es importante destacar que aún existe

una gran cantidad de recursos para el desarrollo teórico del aprendizaje computacional, por lo que su importancia es vital tanto en el sector organizacional como en la academia. Es por esto que el propósito de este trabajo es que sirva como una guía para el inicio del estudio de esta área de la inteligencia artificial, pues aunque el tema se cubre de forma superficial en algunos aspectos y se dejan de lado temas que se alejan del área de probabilidad y estadística, las bases teóricas de los problemas más sencillos de regresión y clasificación pueden ser comprendidos y resueltos con los conceptos aprendidos a nivel licenciatura. Además, partiendo del presente trabajo, es posible continuar el estudio de otros paradigmas del aprendizaje computacional, pues el aprendizaje no supervisado y el aprendizaje por refuerzo son áreas donde las tareas se pueden solucionar utilizando como base los conceptos presentados en esta tesis.

# Apéndice A

# Apéndice

## A.1. Conceptos teóricos

### Variable aleatoria

Sea $(\Omega, \mathcal{F}, P)$ un espacio de probabilidad. Una variable aleatoria es una transformación $X$ del espacio muestral $\Omega$ al conjunto de números reales, esto es,

$$X : \Omega \longrightarrow \mathbb{R}$$

tal que, para cualquier número real $x$,

$$\{\omega \in \Omega : X(\omega) \leq x\} \in \mathcal{F}.$$

### Valor esperado

Sea $X$ una variable aleatoria discreta con posibles valores $x_1, x_2, ..., x_n, ...$ Sea $p(x_i) = P(X = x_i)$, $i = 1, 2, ..., n, ...$ Entonces el valor esperado de $X$ (o esperanza matemática de $X$), denotada por $E(X)$, esta definido por:

$$E(X) = \sum_{i=1}^{n} x_i p(x_i)$$

si la serie $\sum_{i=1}^{n} x_i p(x_i)$ converge absolutamente, es decir, si $\sum_{i=1}^{n} |x_i| p(x_i) < \infty$. A este número también se le conoce como el valor medio de $X$. Para una variable aleatoria continua $X$, el valor esperado de $X$ se define como:

$$E(X) = \int_{-\infty}^{\infty} x f(x)\, \mathrm{d}x.$$

Si todos los valores posibles $x_1, x_2, ..., x_n$ son equiprobables, entonces tenemos:

$$E(X) = \tfrac{1}{n} \sum_{i=1}^{n} x_i = \overline{X}.$$

Comúnmente $\overline{X}$ es llamada media aritmética.

## Varianza

Sea $X$ una variable aleatoria discreta con función de probabilidad $f(X)$. La varianza de $X$ se define como sigue:

$$V(X) = \sum_X (X-\mu)^2 f(X).$$

cuando esta suma es convergente y en donde $\mu$ es la esperanza o valor esperado de $X$. Para una variable aleatoria continua $X$ con función de densidad $f(x)$, se define:

$$V(X) = \int_{-\infty}^{\infty} (x-\mu)^2 f(x)\,\mathrm{d}x.$$

La varianza puede escribirse en una sola expresión como:

$$V(X) = E(X-\mu)^2.$$

## Función de distribución empírica

Sean $X_1, X_2, ...$ variables aleatorias independientes con función de distribución $F$. La función de distribución aleatoria:

$$F_n(x) = \tfrac{1}{n} \sum_{i=1}^{n} I_{(-\infty,x]}(X_i), -\infty < x < \infty$$

donde $I$ es la función indicadora y $F_n$ asigna $\frac{1}{n}$ a cada valor $X_i$ es llamada función de distribución empírica [15].

## Distribución normal

La distribución normal ocurre frecuentemente en el mundo natural. La distribución es simétrica alrededor de su media $\mu$ y tiene desviación estándar $\sigma$, que es tal que prácticamente toda la distribución (99.73 %) se encuentra en el rango $\mu - 3\sigma < x < \mu + 3\sigma$.

La función de frecuencia es:

$$f(x) = \tfrac{1}{\sigma(2\pi)^{\frac{1}{2}}} exp(\tfrac{(x-\mu)^2}{-2\sigma^2}),\ -\infty < x < \infty.$$

Usualmente escribimos $X \sim \mathcal{N}(\mu,\, \sigma^2)$ para decir que $X$ se distribuye normalmente con media $\mu$ y varianza $\sigma^2$. En el caso particular de que $\mu = 0$ y $\sigma^2 = 1$, entonces decimos que $X$ tiene una distribución normal estándar.

Para hacer el cambio a una variable normal estándar $Z$, utilizamos la siguiente expresión:

$$Z = \tfrac{X-\mu}{\sigma}.$$

## Distribución $\chi^2$

Sean $X_1, X_2, ..., X_n$ variables aleatorias independientes que tienen distribución normal estándar. Consideremos la variable aleatoria:

$$\chi^2 = X_1 + X_2 + ... + X_n,$$

$\chi^2$ es llamada "chi-cuadrada" y su función de distribución es:

$$P(\chi^2 \le x) = \tfrac{1}{2^{\frac{n}{2}}\Gamma(\frac{n}{2})} \int_0^x u^{\frac{n}{2}-1} e^{-\frac{u}{2}}\,\mathrm{d}u$$

y su función de densidad de probabilidad es:

$$f(x) = \begin{cases} \frac{1}{2^{\frac{n}{2}}\Gamma(\frac{n}{2})} x^{\frac{n}{2}-1} e^{-\frac{x}{2}} & x > 0, \\ 0 & x \le 0. \end{cases}$$

## Distribución $t$

Existen varias distribuciones $t$, pues depende de la forma de la curva, definida por:

$$f_v(t) = \frac{\Gamma(\frac{v+1}{2})}{(v\pi)^{1/2}\Gamma(\frac{v}{2})}(1+\frac{t^2}{v})^{-\frac{v+1}{2}}, -\infty < t < \infty.$$

la cual depende de $v$, el número de grados de libertad. En general, la distribución $f(t)$ se ve similar a una distribución normal estándar si $v$ es grande ($v \geq 30$).

## Teorema de Bayes

Suponga que $A_1, A_2, ..., A_n$ son eventos mutuamente excluyentes cuya unión es el espacio muestral $S$, es decir, uno de estos eventos debe ocurrir. Entonces si $A$ es un evento cualquiera, tenemos la siguiente expresión:

$$P(A_k|A) = \frac{P(A_k)P(A|A_k)}{\sum_{j=1}^{n} P(A_j)P(A|A_j)}.$$

## Matrices

Sean $K$ un campo, $n, m \in \mathbb{N}$. Una matriz de tamaño $m \times n$, con coeficientes en $K$ es una función $A : \{1, 2, ..., m\} \times \{1, 2, ..., n\} \longrightarrow K$ tal que para cada $i \in \{1, 2, 3, ..., m\}$ y $j \in \{1, 2, 3, ..., n\}$, $A(i, j) = a_{ij} \in K$. [16]

Denotamos a la matriz $A$ como:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}.$$

En este trabajo consideraremos a una matriz como un arreglo rectangular de números reales con $m$ filas y $n$ columnas.

Sea $A$ una matriz de tamaño $n \times n$. Si $I$ es una matriz que satisface la relación $IA = AI = A$, decimos entonces que $I$ es la matriz inversa de $A$, y de la misma forma, decimos que $A$ es una matriz invertible.

Sea $A$ una matriz de dimensiones $n \times m$ con $A = (a_{ij})$; se define la matriz transpuesta de $A$ como $A^T$ tal que $A^T = (a_{ji})$ para cada $i \in \{1, 2, 3, ..., m\}$ y $j \in \{1, 2, 3, ..., n\}$.

## Análisis convexo

Sea $f : \mathbb{R}^n \to \mathbb{R}^m$ una función. Decimos que $f$ es un campo escalar si $m = 1$ y $n > 1$.

El gradiente de un campo escalar está definido como el vector de sus derivadas parciales:

$$\nabla f = (\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, ..., \frac{\partial f}{\partial x_n}).$$

La norma de un campo escalar $f : \mathbb{R}^n \to \mathbb{R}$ se denota por $||f(x)||$ y se define como:

$$||f(x)|| = \sum_{i=1}^{n} [f_i(x)]^{\frac{1}{2}}.$$

Un conjunto $S \subseteq \mathbb{R}^n$ es convexo si para $x_1, x_2 \in S$, entonces $\lambda x_1 + (1-\lambda)x_2 \in S$, para cualquier $\lambda \in [0, 1]$. Es decir, un conjunto $S$ es convexo si para cualesquiera $x_1, x_2 \in S$, el segmento de recta que une a $x_1$ y a $x_2$ también pertenece al conjunto $S$ [9].

Sea $f : S \longrightarrow \mathbb{R}$ con $S \subseteq \mathbb{R}^n$ un subconjunto convexo de $\mathbb{R}^n$. La función $f$ es convexa si:

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda)f(x_2).$$

Para cualesquiera $x_1, x_2 \in S$ y $\lambda \in (0, 1)$.

## Grafos

Un grafo $G$ es un par ordenado $(V, E)$ que consiste en un conjunto $V$ de vértices y un conjunto $E$, disjunto de $V$, de aristas, junto con una función de incidencia $\phi_G$ que asocia cada arista de $G$ con un par ordenado de vértices de $G$. Si $e$ es una arista y $u, v$ son vértices tales que $\phi_G(e) = (u, v)$, entonces se dice que $e$ junta o une a $u$ con $v$, y los vértices $u$ y $v$ son llamados extremos de $e$.

Una arista con extremos iguales se le conoce como bucle, y con distintos extremos se le llama enlace. Dos o más enlaces con los mismos pares de extrmos se dice que son aristas paralelas.

Un grafo es simple si no tiene bulces ni aristas paralelas.

Un camino o ruta es un grafo simple cuyos vértices pueden ser ordenados en una secuencia lineal, de tal forma que dos vértices son adyacentes si asi lo son en la secuencia.

Un ciclo en 3 o mas vértices es un grafo simple cuyos vértices pueden ser ordenados en una secuencia cíclica, de tal forma que dos vértices son adyacentes si son consecutivos en la secuencia. Un grafo es acíclico si no contiene un ciclo.

Un grafo dirigido $D$ es un par ordenado $(V, A)$ que consiste en un conjunto de vértices $V$ y un conjunto $A$ de arcos o flechas, disjunto de $V$, junto con una función de incidencia $\Phi_D(a) = (u, v)$, que asocia a cada arco de $D$ un par ordenado no necesariamente distintos de $D$. En este caso, el vértice $u$ es la cola de $a$, el vértice $v$ es la cabeza de $a$ y los extremos de $a$ son los vértices $u$ y $v$ [17].

# A.2. Código utilizado

## A.2.1. Conjunto de datos de Regresión Lineal Simple

| n | TV | Sales | n | TV | Sales | n | TV | Sales | n | TV | Sales |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 230.1 | 22.1 | 51 | 199.8 | 11.4 | 101 | 222.4 | 11.7 | 151 | 280.7 | 16.1 |
| 2 | 44.5 | 10.4 | 52 | 100.4 | 10.7 | 102 | 296.4 | 23.8 | 152 | 121 | 11.6 |
| 3 | 17.2 | 9.3 | 53 | 216.4 | 22.6 | 103 | 280.2 | 14.8 | 153 | 197.6 | 16.6 |
| 4 | 151.5 | 18.5 | 54 | 182.6 | 21.2 | 104 | 187.9 | 14.7 | 154 | 171.3 | 19 |
| 5 | 180.8 | 12.9 | 55 | 262.7 | 20.2 | 105 | 238.2 | 20.7 | 155 | 187.8 | 15.6 |
| 6 | 8.7 | 7.2 | 56 | 198.9 | 23.7 | 106 | 137.9 | 19.2 | 156 | 4.1 | 3.2 |
| 7 | 57.5 | 11.8 | 57 | 7.3 | 5.5 | 107 | 25 | 7.2 | 157 | 93.9 | 15.3 |
| 8 | 120.2 | 13.2 | 58 | 136.2 | 13.2 | 108 | 90.4 | 8.7 | 158 | 149.8 | 10.1 |
| 9 | 8.6 | 4.8 | 59 | 210.8 | 23.8 | 109 | 13.1 | 5.3 | 159 | 11.7 | 7.3 |
| 10 | 199.8 | 10.6 | 60 | 210.7 | 18.4 | 110 | 255.4 | 19.8 | 160 | 131.7 | 12.9 |
| 11 | 66.1 | 8.6 | 61 | 53.5 | 8.1 | 111 | 225.8 | 13.4 | 161 | 172.5 | 14.4 |
| 12 | 214.7 | 17.4 | 62 | 261.3 | 24.2 | 112 | 241.7 | 21.8 | 162 | 85.7 | 13.3 |
| 13 | 23.8 | 9.2 | 63 | 239.3 | 15.7 | 113 | 175.7 | 14.1 | 163 | 188.4 | 14.9 |
| 14 | 97.5 | 9.7 | 64 | 102.7 | 14 | 114 | 209.6 | 15.9 | 164 | 163.5 | 18 |
| 15 | 204.1 | 19 | 65 | 131.1 | 18 | 115 | 78.2 | 14.6 | 165 | 117.2 | 11.9 |
| 16 | 195.4 | 22.4 | 66 | 69 | 9.3 | 116 | 75.1 | 12.6 | 166 | 234.5 | 11.9 |
| 17 | 67.8 | 12.5 | 67 | 31.5 | 9.5 | 117 | 139.2 | 12.2 | 167 | 17.9 | 8 |
| 18 | 281.4 | 24.4 | 68 | 139.3 | 13.4 | 118 | 76.4 | 9.4 | 168 | 206.8 | 12.2 |
| 19 | 69.2 | 11.3 | 69 | 237.4 | 18.9 | 119 | 125.7 | 15.9 | 169 | 215.4 | 17.1 |
| 20 | 147.3 | 14.6 | 70 | 216.8 | 22.3 | 120 | 19.4 | 6.6 | 170 | 284.3 | 15 |
| 21 | 218.4 | 18 | 71 | 199.1 | 18.3 | 121 | 141.3 | 15.5 | 171 | 50 | 8.4 |
| 22 | 237.4 | 12.5 | 72 | 109.8 | 12.4 | 122 | 18.8 | 7 | 172 | 164.5 | 14.5 |
| 23 | 13.2 | 5.6 | 73 | 26.8 | 8.8 | 123 | 224 | 11.6 | 173 | 19.6 | 7.6 |
| 24 | 228.3 | 15.5 | 74 | 129.4 | 11 | 124 | 123.1 | 15.2 | 174 | 168.4 | 11.7 |
| 25 | 62.3 | 9.7 | 75 | 213.4 | 17 | 125 | 229.5 | 19.7 | 175 | 222.4 | 11.5 |
| 26 | 262.9 | 12 | 76 | 16.9 | 8.7 | 126 | 87.2 | 10.6 | 176 | 276.9 | 27 |
| 27 | 142.9 | 15 | 77 | 27.5 | 6.9 | 127 | 7.8 | 6.6 | 177 | 248.4 | 20.2 |
| 28 | 240.1 | 15.9 | 78 | 120.5 | 14.2 | 128 | 80.2 | 8.8 | 178 | 170.2 | 11.7 |
| 29 | 248.8 | 18.9 | 79 | 5.4 | 5.3 | 129 | 220.3 | 24.7 | 179 | 276.7 | 11.8 |
| 30 | 70.6 | 10.5 | 80 | 116 | 11 | 130 | 59.6 | 9.7 | 180 | 165.6 | 12.6 |
| 31 | 292.9 | 21.4 | 81 | 76.4 | 11.8 | 131 | 0.7 | 1.6 | 181 | 156.6 | 10.5 |
| 32 | 112.9 | 11.9 | 82 | 239.8 | 12.3 | 132 | 265.2 | 12.7 | 182 | 218.5 | 12.2 |
| 33 | 97.2 | 9.6 | 83 | 75.3 | 11.3 | 133 | 8.4 | 5.7 | 183 | 56.2 | 8.7 |
| 34 | 265.6 | 17.4 | 84 | 68.4 | 13.6 | 134 | 219.8 | 19.6 | 184 | 287.6 | 26.2 |
| 35 | 95.7 | 9.5 | 85 | 213.5 | 21.7 | 135 | 36.9 | 10.8 | 185 | 253.8 | 17.6 |
| 36 | 290.7 | 12.8 | 86 | 193.2 | 15.2 | 136 | 48.3 | 11.6 | 186 | 205 | 22.6 |
| 37 | 266.9 | 25.4 | 87 | 76.3 | 12 | 137 | 25.6 | 9.5 | 187 | 139.5 | 10.3 |
| 38 | 74.7 | 14.7 | 88 | 110.7 | 16 | 138 | 273.7 | 20.8 | 188 | 191.1 | 17.3 |
| 39 | 43.1 | 10.1 | 89 | 88.3 | 12.9 | 139 | 43 | 9.6 | 189 | 286 | 15.9 |
| 40 | 228 | 21.5 | 90 | 109.8 | 16.7 | 140 | 184.9 | 20.7 | 190 | 18.7 | 6.7 |
| 41 | 202.5 | 16.6 | 91 | 134.3 | 11.2 | 141 | 73.4 | 10.9 | 191 | 39.5 | 10.8 |
| 42 | 177 | 17.1 | 92 | 28.6 | 7.3 | 142 | 193.7 | 19.2 | 192 | 75.5 | 9.9 |
| 43 | 293.6 | 20.7 | 93 | 217.7 | 19.4 | 143 | 220.5 | 20.1 | 193 | 17.2 | 5.9 |
| 44 | 206.9 | 12.9 | 94 | 250.9 | 22.2 | 144 | 104.6 | 10.4 | 194 | 166.8 | 19.6 |
| 45 | 25.1 | 8.5 | 95 | 107.4 | 11.5 | 145 | 96.2 | 11.4 | 195 | 149.7 | 17.3 |
| 46 | 175.1 | 14.9 | 96 | 163.3 | 16.9 | 146 | 140.3 | 10.3 | 196 | 38.2 | 7.6 |
| 47 | 89.7 | 10.6 | 97 | 197.6 | 11.7 | 147 | 240.1 | 13.2 | 197 | 94.2 | 9.7 |
| 48 | 239.9 | 23.2 | 98 | 184.9 | 15.5 | 148 | 243.2 | 25.4 | 198 | 177 | 12.8 |
| 49 | 227.2 | 14.8 | 99 | 289.7 | 25.4 | 149 | 38 | 10.9 | 199 | 283.6 | 25.5 |
| 50 | 66.9 | 9.7 | 100 | 135.2 | 17.2 | 150 | 44.7 | 10.1 | 200 | 232.1 | 13.4 |

### A.2.2. Regresión Lineal Simple: Programa en R

```
# Regresion Lineal Simple

# Importar el conjunto de datos
dataset = read.csv('tvmarketing.csv')

# Visualizar el conjunto de datos
library(ggplot2)
ggplot(data = dataset,
       aes(x = TV, y = Sales)) + ggtitle("Conjunto de datos")+
  theme(plot.title = element_text(hjust = 0.5))+
  geom_point()+
xlab('TV')+ylab('Ventas')
# Division del conjunto de datos
# install.packages('caTools')
library(caTools)
set.seed(123)
split = sample.split(dataset$Sales, SplitRatio = 0.8)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)


# Calculo del modelo de regresion
regressor = lm(formula = Sales ~ TV,
               data = training_set)
#Coeficiente de correlacion entre X y Y
#a partir de el conjunto de entrenamiento:
cor(training_set)
cor.test(training_set$TV, training_set$Sales)

#Intervalo de confianza para el coeficiente de correlacion
#r representa el coeficiente de correlacion
#n representa el numero de observaciones
#level representa el nivel de confianza
library(psychometric)
CIr(r = 0.7998, n = 160, level = 0.95)

#Histograma de residuales

ggplot(data = training_set, aes(x = regressor$residuals)) +
geom_histogram(bins = 15, fill = 'steelblue',
               color = 'black') +
labs(title = 'Histograma de residuales',
     x = 'Residuales', y = 'Frecuencia')

#Grafica de probabilidad normal
regressor.stdres = rstandard(regressor)
qqnorm(regressor.stdres,  ylab = "Residuales estandarizados",
 xlab = "Valor normal",  main =  " ")
qqline(regressor.stdres)
#Prueba de Kolmogorov-Smirnov
```

```
library(olsrr)
ols_test_normality(regressor)
#Intervalos de confianza para los parametros beta0 y beta1:
confint(regressor, level = 0.95)
# Predicting the Test set results
y_pred = predict(regressor, newdata = test_set)

#Calculo del error cuadratico medio
library(Metrics)
rmse(test_set$Sales, y_pred)
# Visualising the Training set results
library(ggplot2)
ggplot() +
  geom_point(aes(x = training_set$TV, y = training_set$Sales),
             colour = 'red') +
  geom_line(aes(x = training_set$TV,
                y = predict(regressor,
                newdata = training_set)),
            colour = 'blue') +
  ggtitle('TV vs Ventas (Conjunto de entrenamiento)') +
  xlab('TV') +
  ylab('Ventas')

# Visualising the Test set results
library(ggplot2)
ggplot() +
  geom_point(aes(x = test_set$TV, y = test_set$Sales),
             colour = 'red') +
  geom_line(aes(x = training_set$TV,
                y = predict(regressor, newdata = training_set)),
                colour = 'blue') +
  ggtitle('X vs Y(Test set)') +
  xlab('X') +
  ylab('Y')
```

### A.2.3. Regresión Lineal Múltiple: Mínimos Cuadrados

```
#importar librerias
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

#importar el conjunto de datos
dataset = pd.read_csv('ejemploRLM.csv')
X=dataset.iloc[:,:-1].values
y=dataset.iloc[:,-1].values
#Dividir el conjunto de datos en conjunto de entrenamiento y conjunto de prueba
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)

#Entrenar el modelo de regresion lineal multiple en el conjunto de entrenamiento
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)

#Prediccion de los resultados del conjunto de prueba
y_pred = regressor.predict(X_test)
np.set_printoptions(precision=2) #representa cualquier numero solo con 2 decimales
print(np.concatenate((y_pred.reshape(len(y_pred),1),
        y_test.reshape(len(y_test),1)),1))

#Hacer una sola prediccion:
#X_1=5,X_2=240,X_3=0.87,X_4=0.62,X_5=240
print(regressor.predict([[5,240,0.87,0.62,240]]))

#Ver los coeficientes del modelo
print(regressor.coef_)
print(regressor.intercept_)
```

### A.2.4. Regresión Lineal Múltiple: Algoritmo de Gradiente de Descenso

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("ejemploRLM.csv")
X = df.iloc[:,:-1].values
y = df.iloc[:,-1].values
m = len(y)
print("Total no. of training examples (m) = %s \n," %m)

#mostrar las primeras 5 observaciones
for i in range(5):
    print("x = ", X[i,], ', y = ', y[i])

#funcion para normalizar los atributos
def feature_normalize(X):
    mu = np.mean(X, axis = 0)
    sigma = np.std(X,axis=0,ddof=1)
    X_norm = (X-mu)/sigma
    return X_norm, mu, sigma

#funcion de costo
def compute_cost(X,y,theta):
    predictions = X.dot(theta)
    errors = np.subtract(predictions,y)
    sqrErrors = np.square(errors)
    J = 1/(2*m)* np.sum(sqrErrors)
    return J

#funcion que realiza el algoritmo de gradiente de descenso
def gradient_descent(X,y,theta,alpha,iterations):
    cost_history=np.zeros(iterations)
    for i in range(iterations):
        predictions = X.dot(theta)
        errors = np.subtract(predictions,y)
        sum_delta = (alpha / m) * X.transpose().dot(errors)
        theta = theta - sum_delta
        cost_history[i] = compute_cost(X,y,theta)

    return theta,cost_history


X, mu, sigma = feature_normalize(X)
print("mu = ", mu)
print("sigma = ", sigma)
print("X_norm = ", X)

mu_testing = np.mean(X,axis = 0)
print(mu_testing)
```

```
sigma_testing = np.std(X, axis = 0, ddof = 1)
print(sigma_testing)

#Se agrega la primera columna de unos a la matriz de atributos X
X = np.hstack((np.ones((m,1)),X))

theta = np.zeros(len(X[0]))
iterations = 400
alpha = 0.15

theta, cost_history = gradient_descent(X,y,theta,alpha,iterations)
print("Valores finales de los parametros =", theta)

#prediccion: X_1=5,X_2=240,X_3=0.87,X_4=0.62,X_5=240
normalize_test_data = (([5,240,0.87,0.62,240])-mu)/sigma
normalize_test_data = np.hstack((np.ones(1), normalize_test_data))
predict = normalize_test_data.dot(theta)
print("Prediccion: ", predict)
```

### A.2.5. Regresión Lineal Múltiple: Backwards Elimination

```
#importar librerias
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

#importar el conjunto de datos
dataset = pd.read_csv('ejemploRLM.csv')
X=dataset.iloc[:,:-1].values
y=dataset.iloc[:,-1].values

#Dividir el conjunto de datos en conjunto de entrenamiento y conjunto de prueba
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test= train_test_split(X,y,test_size=0.2, random_state=0

#Entrenar el modelo de regresion lineal multiple en el conjunto de entrenamiento
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X,y)

#Prediccion de los resultados del conjunto de prueba
y_pred = regressor.predict(X_test)
np.set_printoptions(precision=4) #redondea a 4 decimales
print(np.concatenate((y_pred.reshape(len(y_pred),1),
                      y_test.reshape(len(y_test),1)),1))


#Ver los coeficientes del modelo
print(regressor.coef_)
print(regressor.intercept_)

#backwards elimination
# Building the optimal model using Backward Elimination
import statsmodels.api as sm
X = np.append(arr = np.ones((len(X), 1)).astype(int), values = X, axis = 1)
#X = np.append(arr = X, values = np.ones((len(X),1)).astype(int), axis = 1)
X_opt = X[:, [0, 1, 2, 3, 4, 5]]
X_opt = X_opt.astype(np.float64)
regressor_OLS = sm.OLS(endog = y, exog = X_opt).fit()
print(regressor_OLS.summary())
#Removemos la variable X_5 por tener p-value mayor al nivel de significancia
X_opt = X[:, [0, 1,2, 3, 4]]
X_opt = X_opt.astype(np.float64)
regressor_OLS = sm.OLS(endog = y, exog = X_opt).fit()
print(regressor_OLS.summary())
#Removemos la variable X_2 por tener p-value mayor al nivel de significancia
X_opt = X[:, [0,1, 3, 4]]
X_opt = X_opt.astype(np.float64)
regressor_OLS = sm.OLS(endog = y, exog = X_opt).fit()
print(regressor_OLS.summary())
#Removemos la variable X_4 por tener p-value mayor al nivel de significancia
```

```
X_opt = X[:, [0, 1,3]]
X_opt = X_opt.astype(np.float64)
regressor_OLS = sm.OLS(endog = y, exog = X_opt).fit()
print(regressor_OLS.summary())
#Removemos la variable X_3 por tener p-value mayor al nivel de significancia
X_opt = X[:, [0, 1]]
X_opt = X_opt.astype(np.float64)
regressor_OLS = sm.OLS(endog = y, exog = X_opt).fit()
print(regressor_OLS.summary())
```

# Bibliografía


[1] RINCÓN, LUIS, *Curso intermedio de probabilidad*, Facultad de Ciencias, UNAM, 2010.

[2] CASELLA,GEORGE, *Statistical Inference*, Duxbury, 2002.

[3] YASSER, H., *Advertising Sales Dataset | Kaggle*, [En línea] Disponible en `https://www.kaggle.com/datasets/yasserh/advertising-sales-dataset` (Visitada el 10.11.22)

[4] MITCHELL, TOM M., *Machine Learning*, McGraw-Hill, 1997.

[5] DRAPER, NORMAN R., *Applied Regression Analysis*, Wiley, 1998.

[6] MONTGOMERY, DOUGLAS C., *Introducción al análisis de regresión lineal*, Compañia Editorial Continental, 2006.

[7] LILLIEFORS, HUBERT W., *On the Kolmogorov-Smirnov Test for Normality with Mean and Variance Unknown*, Journal of the American Statistical Association, 2012.

[8] INFANTE GIL, SAID, *Métodos Estadísticos: Un enfoque interdisciplinario*, Trillas, 1990.

[9] MOKHTAR S. BAZARAA, *Nonlinear Programming: Theory and Algorithms*, Wiley-Interscience, 2006.

[10] MURPHY, KEVIN P., *Machine Learning: A probabilistic perspective*, The MIT Press, 2012.

[11] SUCAR, LUIS ENRIQUE, *Probabilistic Graphical Models: Principles and Aplications*, Springer, 2021.

[12] T. LE, CHAP, *Introductory Biostatistics*, Wiley-Interscience, 2003.

[13] HOLLANDER, WOLFE, CHICKEN, *Nonparametric Statistical Methods*, Wiley & Sons, 2014.

[14] Demsar, Janez, *Statistical Comparisons of Classifiers over Multiple Data Sets*, Journal of Machine Learning Research 7, 2006.

[15] RINCÓN, LUIS, *Una introducción a la estadística inferencial*, Facultad de Ciencias, UNAM, 2019.

[16] ANGOA, IBARRA, LINARES, ANDRADE, *Introducción al álgebra lineal*, Facultad de Ciencias Físico-Matemáticas y Dirección General de Publicaciones, 2019.

[17] BONDY, J.A., MURTY, U.S.R., *Graph Theory*, Springer, 2008.

[18] SONAWE, AJINKYA, *Building the Naïve Bayes Classifier from scratch in Python*, [En línea] Disponible en `https://blog.goodaudience.com/building-the-naïve-bayes-classifier-from-scratch-in-python-b0717fa022d8` (Visitada el 7.12.21)

[19] WALLIS, DIÓGENES, *Comparing classifiers (Friedman and Nemenyi tests*, [En línea] Disponible en `https://medium.com/mlearning-ai/comparing-classifiers-friedman-and-nemenyi-tests-32294103ee12` (Visitada el 2.12.21)