
# PROYECTO AI

## WebMining: Aprendizaje Supervisado, No Supervisado y Por Refuerzo.

**Integrantes:**

Cesar Eduardo Elías del Hoyo

José Luis Sandoval Pérez

Diego Emanuel Saucedo Ortega

Carlos Daniel Torres Macías

**Universidad Autónoma de Aguascalientes**

Aguascalientes, Ags, 17 de mayo, 2024

# Análisis

Para esta práctica, se implentaron todas las técnicas de Machine Learning a partir de un propio dataset obtenido mediante WebMining. Para ello, daremos instroducción a cada uno de los métodos de aprendizaje empleados

## APRENDIZAJE SUPERVISADO

El aprendizaje es una subcatergoría del machine learning que se define por el uso de conjunto de datos etiquetados para entrenar algoritmos que clasifican los datos o predicen los resultados con precisión. Esto significa que cada ejemplo de entrenamiento incluye una entrada y una salida deseada.

El objetivo es el modelo que aprenda a mapear las entradas a las salidas correctas basándose en los ejemplos proporcionados. Una vez entrenado, el modelo puede predecir las salidas para nuevas entradas no vistas

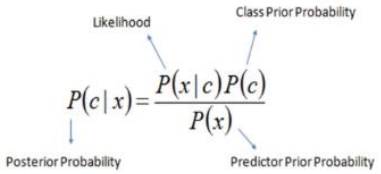
Para el aprendizaje supervisado, realizamos su comprobación mediante los siguientes métodos:

### Gaussian Naive Bayes

Gaussian Naive Bayes es un tipo de método Naive Bayes en el que se consideran atributos continuos y las características de los datos siguen una distribución gaussiana a lo largo del conjunto de datos. Gaussian Naive Bayes es un tipo de algoritmo de clasificación que funciona con características continuamente distribuidas de forma normal y está basado en el algoritmo Naive Bayes. Antes de profundizar en este tema, debemos obtener una comprensión básica de los principios en los que se basa Gaussian Naive Bayes.

Para clasificar es utilizada la siguiente relación entre las probabilidades de pertenecer a una clase o a otra:

## Naive Bayes Classifier




$$P(c|X) = P(x_1|c)\times P(x_2|c)\times\cdots\times P(x_n|c)\times P(c)$$

Esta fórmula se puede explicar en el siguiente procedimiento paso a paso que describe el algoritmo perfectamente:

1. **Calcular las probabilidades previas** $P(Ck)$ **para cada clase** $Ck$:

$$P(C_k) = \frac{número\ de\ instancias\ de\ C_k}{número\ total\ de\ instancias}$$

2. **Calcular la media** $\mu k,i$ **y la varianza** $\sigma k,i2$ **para cada característica** $i$ **en cada clase** $Ck$:

$$u_{k,i} = \frac{1}{N_k}\sum_{j=1}^{N_k} X_{j,i}$$

$$\sigma^2{}_{k,i} = \frac{1}{N_k}\sum_{j=1}^{N_k}(X_{j,i} - u_{k,i})^2$$

3. **Calcular la probabilidad condicional** $P(xi|Ck)$ **usando la distribución normal**:

$$P(X_i \mid C_k) = \frac{1}{\sqrt{2\pi\sigma^2{}_{k,i}}}\exp\left(-\frac{(X_{j,i} - u_{k,i})^2}{2\sigma^2{}_{k,i}}\right)$$

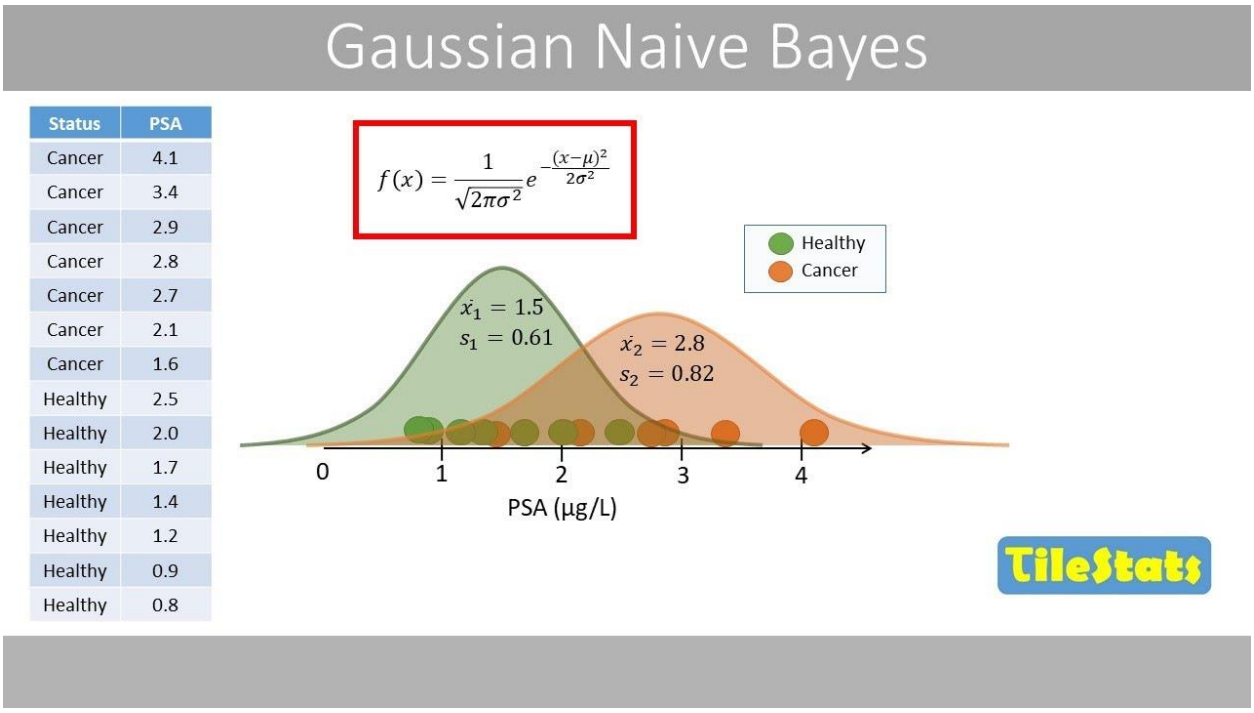
4. **Calcular la probabilidad posterior para cada clase** $Ck$:

$$P(C_k \mid x) \propto P(C_k)\prod_{i=1}^{n} P(X_i \mid C_k)$$

5. **Asignar la clase con la mayor probabilidad posterior**:

$$\hat{y} = \arg max_k P(C_k \mid x)$$

Estas formulas nos permiten obtener resultados como el ejemplo mostrado en la siguiente imagen




## K-Nearest Neighbors

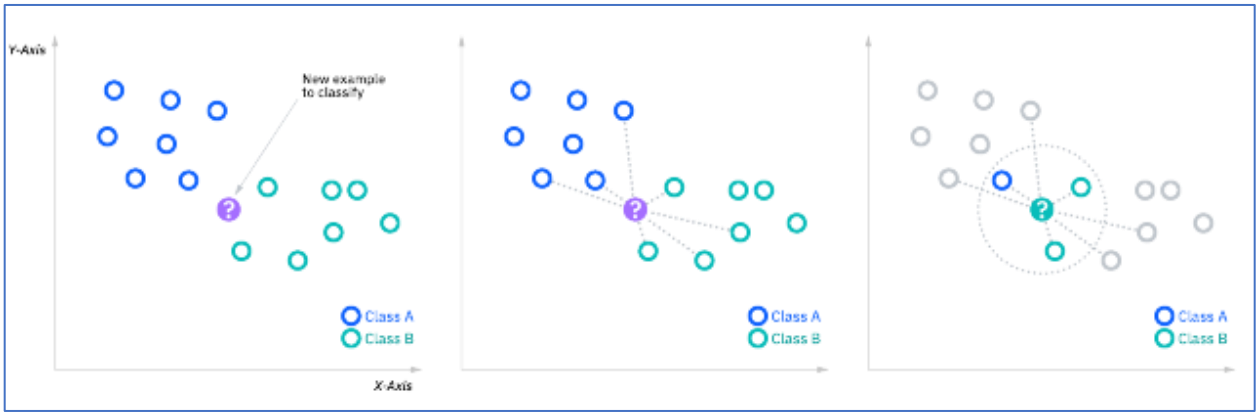
K-Nearest Neighbors es un algoritmo de aprendizaje supervisado utilizado principalmente para problemas de clasificación y, en menor medida, regresión. El principio básico del K-NN es que una muestra se clasifica según la mayoría de sus vecinos más cercanos.

Este algoritmo tiene un funcionamiento que se aplica de la siguiente manera:

1. **Selección de $k$:** Decidir el número de vecinos $k$.
2. **Cálculo de distancias:** Calcular la distancia entre la muestra a clasificar y todas las muestras en el conjunto de entrenamiento (usualmente se usa la distancia euclidiana).
3. **Identificación de vecinos:** Seleccionar los $k$ vecinos más cercanos.

4. **Clasificación o regresión:** Para clasificación, asignar la clase más común entre los vecinos. Para regresión, calcular el promedio de los valores de los vecinos.

Para los problemas de clasificación, se asigna una etiqueta de clase sobre la base de un voto mayoritario, es decir, se utiliza la etiqueta que se representa con más frecuencia alrededor de un punto de datos determinado




Para su implementación en algoritmo, se requiere de una serie de pasos y procedimientos formulados que describen el método:

- **Seleccionar el número de vecinos $k$.**
- **Para cada punto de prueba $x$:**
  - Calcular la distancia entre $x$ y todos los puntos del conjunto de entrenamiento.
  - Seleccionar los $k$ puntos del conjunto de entrenamiento que están más cerca de $x$.
  - Asignar la clase que es más frecuente entre los $k$k vecinos más cercanos.

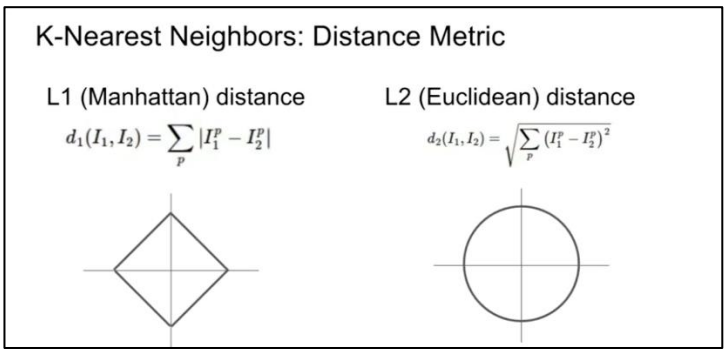
**Fórmula de Distancia Euclidiana:**

La distancia euclidiana entre dos puntos $x = (x_1, x_2, \ldots, x_n)$ y $y = (y_1, y_2, \ldots, y_n)$ en un espacio n-dimensional se calcula como:

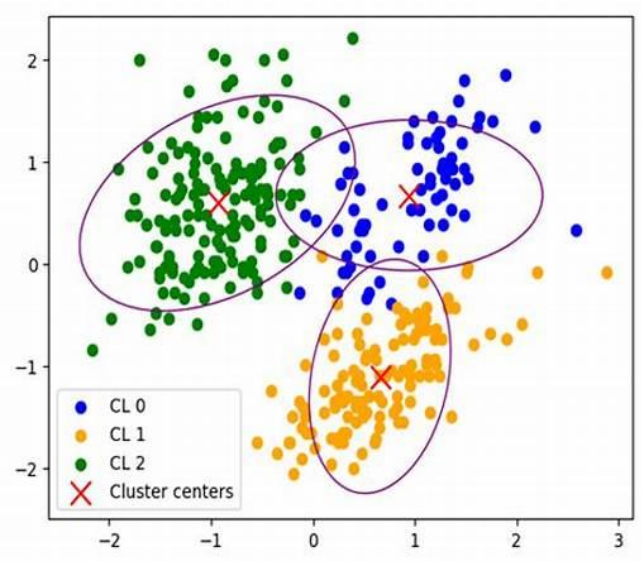
$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

En general, K-NN puede ser utilizado para tareas como la clasificación de página web en categorías temáticas, recomendación de artículos similares y detección de contenido duplicado

La fórmula expuesta anteriormente se representa de la siguiente manera




En este caso, existen dos fórmulas que aplican para este algoritmo, pero el más usado es el caso Euclidiano. Esto dentro de la programación aplicada con K-NN se muestra de esta manera

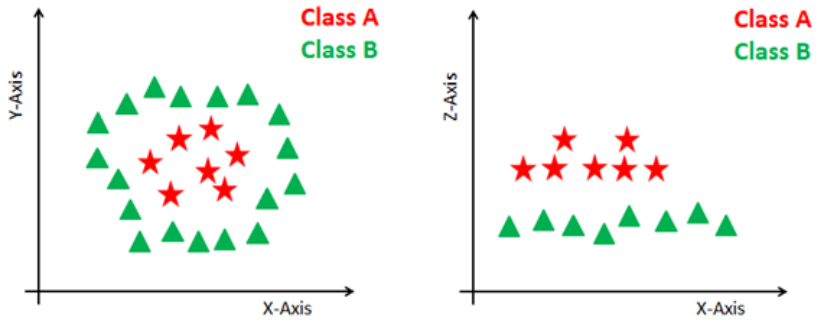



## Support Vector Machines (SVM)

Algoritmo que separa a los puntos del dataset mediante hiperplanos buscando el máximo margen de separación, es por ello que es conocido como un algoritmo discriminatorio. SVM busca un hiperplano óptimo que ayude a clasificar nuevos datos.

Su principal objetivo es segregar las clases de forma que la separación de estas sea la mayor. Para ello realiza los siguientes pasos:

1. Generar un hiperplano que segregue la mayor cantidad de datos, primero entre dos diferentes clasificaciones.
2. Si existe el problema de datos inseparables, SVM hace uso de un plano diferente para buscar una diferenciacion, como se muestra en la imagen a continuación:




3. Una parte esencial del algoritmo es el kernel, quien es el encargado de realizar las diferentes transformaciones de los datos a fin de encontrar mejores seperaciones, es decir convierte los problemas no separables en separables, realizando así un mejor algoritmo de clasificación.

| | |
|---|---|
| $\sum_{i=1}^{n} \lambda_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j x_i x_j$ | $\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i x_j$ |
| Dot product distance or similarity measurement formula. | Kernel looks version |
| $\sum_{i=1}^{n} \lambda_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j f(x_i x_j)$ | $\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i x_j)$ |
| Constraints from derivatives | |
| $w = \sum_{i=1}^{n} \alpha_i y_i x_i$ <br> $\sum_{i=1}^{n} \alpha_i y_i = 0$ | |

Así la decisión del clasificador se obtiene por medio de las diversas transformaciones y separaciones que se realizan a través de las iteraciones del algoritmo, definiendo las fronteras óptimas para los nuevos datos a analizar.

# APRENDIZAJE NO SUPERVISADO

El aprendizaje no supervisado es un tipo de aprendizaje automático (machine learning) donde el modelo se entrena utilizando datos que no están etiquetados ni categorizados. A diferencia del aprendizaje supervisado, en el cual los datos de entrenamiento incluyen tanto las entradas como las salidas deseadas (etiquetas), el aprendizaje no supervisado se ocupa de encontrar estructuras o patrones ocultos en los datos sin ninguna guía explícita sobre qué debe aprender

El aprendizaje no supervisado es especialmente útil cuando se tienen grandes volúmenes de datos y se quiere explorar su estructura sin tener que etiquetar manualmente cada instancia. Además, puede ser utilizado como una etapa previa al aprendizaje supervisado, ya que puede ayudar a identificar características relevantes o a crear nuevas representaciones de los datos que mejoren el rendimiento de los modelos supervisados.

## K-Means

El algoritmo k-means es un método de agrupamiento que divide un conjunto de datos en k grupos o clusters. Los datos se agrupan de modo que los puntos en el mismo clúster sean más similares que los puntos en otro.

Del universo de algoritmos de aprendizaje no supervisado, K-means es probablemente el más reconocido. La razón por la que existe este método es porque hoy en día la cantidad total de datos creados, capturados, copiados y consumidos globalmente es de aproximadamente 100 Zettabytes y seguirá creciendo. Con el algoritmo k- means es posible recopilar grandes cantidades de información similar en un mismo lugar, hecho que ayuda a encontrar patrones y hacer predicciones en grandes conjuntos de datos.

El algoritmo es iterativo y se compone de los siguientes pasos:

1. **Inicialización:** Selecciona $K$ centroides iniciales aleatoriamente.

2. **Asignación de Clústeres:** Asigna cada punto al clúster cuyo centroide esté más cercano. Esto se hace calculando la distancia euclidiana entre el punto $xi$xi y cada centroide $cj$:

$$d(x_i, c_i) = \sqrt{\sum_{k=1}^{m} (x_{ik} - c_{jk})^2}$$

donde $m$ es el número de características.

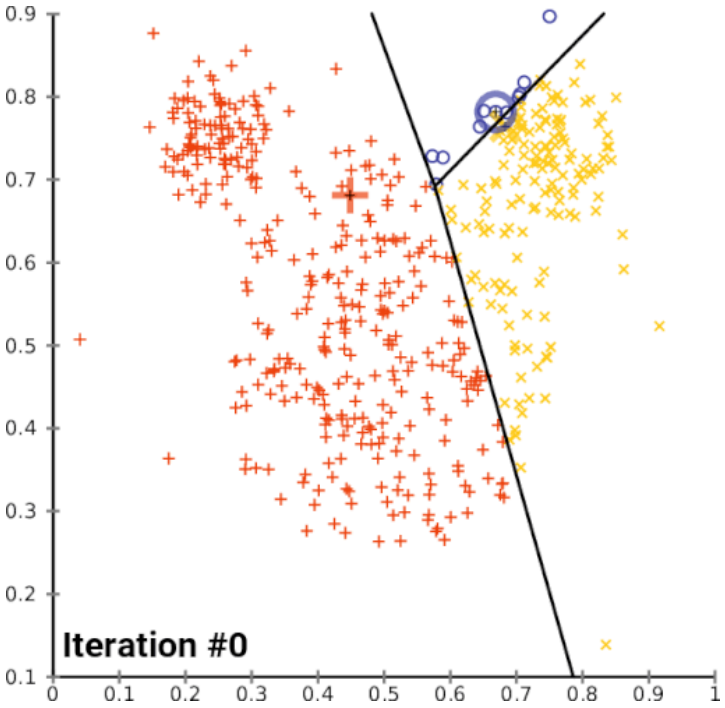
3. **Actualización de Centroides:** Recalcula los centroides como la media de todos los puntos asignados a cada clúster.

$$c_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$$

donde $Cj$ es el conjunto de puntos en el clúster $j$

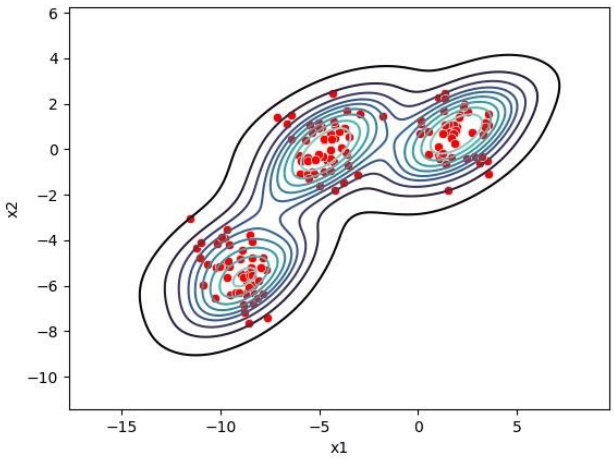
4. **Convergencia:** Repite los pasos 2 y 3 hasta que los centroides no cambien significativamente.

En este gif de ejemplo se ve bien cómo se mueven los centroides y cambian los grupos con las distintas iteraciones. Recordamos que un centroide es un punto de datos (imaginario o real) en el centro de un clúster



# Mean Shift

Mean Shift es un modelo de aprendizaje unsupervisado que apunta al descubrimiento de burbujas en una densidad de datos lisa. Está basada en el posicionamiento de centroides que funcionan como candidatos y el algoritmo es el encargado de ajustar las medias con respecto a la distancia entre datos para mejorar su agrupamiento, estos centros a su vez pasan por un proceso de eliminación para evitar que junten.




Pasos importantes:

- Crear una ventana de datos para cada punto del dataset.
- Desplazar cada ventana considerando la mayor densidad de las regiones de los datos cambiando entre los distintos centros. Este proceso es repeitdo hasta encontrar la máxima densidad de los centros.
- Seleccionar los centroides a utilizar eliminando aquellos centros cuyas regiones estan sobrepuestas, siendo el punto con mayor cantidad de puntos el que se conserva.
- Asignar los puntos al grupo donde fueron deslizados.

$$m(x) = \frac{\sum_{x_i \in N(x)} K(x_i - x) x_i}{\sum_{x_i \in N(x)} K(x_i - x)}$$

# APRENDIZAJE POR REFUERZO

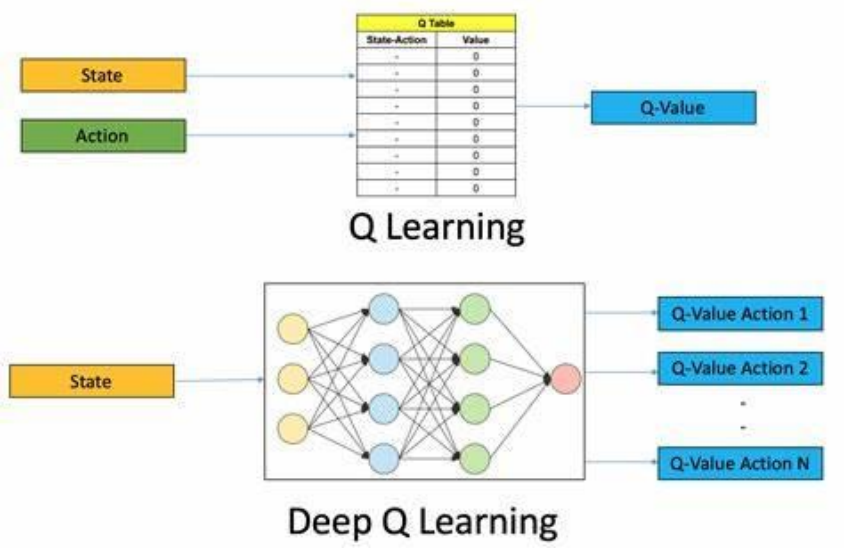
**Q-Learning**

El algoritmo aprende a mejorar su comportamiento a través del tiempo interactuando iterativamente con el entorno.

Estos entornos a lso que son expuestos se denominan estado, de las cuales el algoritmo tomar a la opción más gratificante con el fin de que las recompensas o castigos mejoren el rendimiento.

Algunos componentes clave son:

- Valores de accion: definen los estados de acción y estiman que tanto beneficio ofrece ejercer una acción especifica en un estado.
- Recompensa: A través de la vida del agente desde un estado inicial hasta el siguiente. Tras cada paso el agente observa una recompensa al ambiente. Esto seguira hasta que alcance un estado final donde no es posible realizar más transiciones.
- Diferencial del tiempo: Estima el valor de un estado aplicado a un tiempo especificado en la interaccion del agente.
- Curso de acción: Determinar qué es lo que se desea maximizar o minimizar, cual es la función objetivo determinada para el agente y de qué forma repercuten las acciones a su resultado.



# Implementación

Para esta actividad realizamos la implementación del WebMining en la búsqueda de datos en la web, nosotros tomamos información de una página web. De esta manera, al obtener los datos aplicamos los métodos de aprendizaje supervisado, no supervisado y por refuerzo para su análisis e interpretación de la información recibida

## Extracción, preparación y transformación de los datos

Insertamos el html de la pagina en un elemento htmlSoup para poder analizar el html

```
url = 'https://www.kaggle.com/datasets/aminizahra/tips-dataset'

options = Options()
options.add_argument('--user-agent=Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36')
options.add_argument('--headless')

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=options)

driver.get(url)

driver.find_elements(By.XPATH, '//title')
```

```
[<selenium.webdriver.remote.webelement.WebElement (session="0a8d25b960c67a09ac23fbbcd0ac6b3c", element="f.88008E97B808B26684E335A0DD33AC7A.d.CA2FC3216491E5E7BCCD972B7C11038C.e.57")>]
```

```
# comenzamos a buscar las etiquetas de la tabla
rows = driver.find_elements(By.XPATH,'//span[@class="sc-iKOmoZ ca-DKfq"]')
rows[0].text
```

## Ajustamos el DataSet

### Ajustamos el dataset

Dejamos todas la variables como numeros

```
df.info()
```

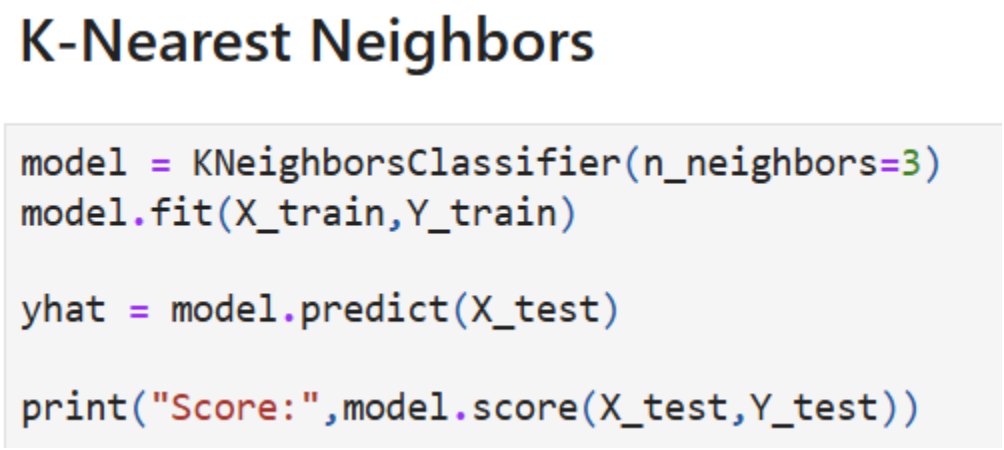
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Bill         50 non-null     float64
 1   Tip          50 non-null     int64  
 2   Sex          50 non-null     int64  
 3   Smoking      50 non-null     int64  
 4   Day          50 non-null     int64  
 5   Time         50 non-null     int64  
 6   Size         50 non-null     int64  
 7   PricePerson  50 non-null     float64
dtypes: float64(2), int64(6)
memory usage: 3.3 KB
```

Una vez que se obtienen los datos ordenados, aplicamos los métodos en orden de aprendizaje supervisado, no supervisado y por refuerzo

## APRENDIZAJE SUPERVISADO

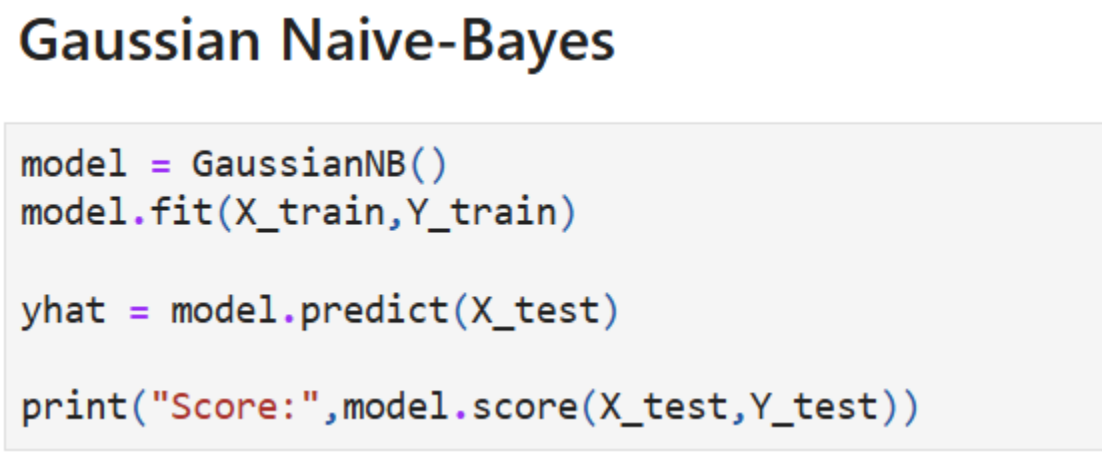
### K-Nearest Neighbors

Su implementación en Python se ve de la siguiente manear




### Gaussian Naive-Bayes

Su aplicación en Pyhton se muestra de la siguiente manera




### Support Vector Machines

Para el lenguaje de Python se aplica de igual manera

## Support Vector Machines

```
model = SVC()
model.fit(X_train,Y_train)

yhat = model.predict(X_test)

print("Score:",model.score(X_test,Y_test))
```

## APRENDIZAJE NO SUPERVISADO

### K-Means

Para el lenguaje de Python se aplica de igual manera

## KMeans

```
model = KMeans(init = "k-means++", n_clusters = 3, n_init = 12)
model.fit(X)

labels = model.labels_
labels
```

### MeanShift

Para el lenguaje de Python se aplica de igual manera

## MeanShift

```
model = MeanShift()
model.fit(X)

labels = model.labels_
labels
```

```
plt.scatter(X[:, 0], X[:, 1], c=model.labels_, cmap='rainbow')
plt.scatter(model.cluster_centers_[:, 0], model.cluster_centers_[:, 1], color='black', marker='x')
plt.show()
```

## APRENDIZAJE POR REFUERZO

Para el aprendizaje por refuerzo, se implementa el código de la siguiente manera

# Aprendizaje por refuerzo

Ingresamos los datos de la primera columna y establecemos una meta, para que dichos datos aLcancen, dicho valor.

```
# inicializamos una matriz de ceros
n_states = 16
n_actions = 4
goal_state = 15

Q_table = np.zeros((n_states, n_actions))

learning_rate = 0.8
discount_factor = 0.95
exploration_prob = 0.2
epochs = 1000

for epoch in range(epochs):
    current_state = np.random.randint(0, n_states)  # Start from a random state

    while current_state != goal_state:
        if np.random.rand() < exploration_prob:
            action = np.random.randint(0, n_actions)  # Explore
        else:
            action = np.argmax(Q_table[current_state])  # Exploit

        next_state = (current_state + 1) % n_states

        reward = 1 if next_state == goal_state else 0

        Q_table[current_state, action] += learning_rate * \
            (reward + discount_factor *
             np.max(Q_table[next_state]) - Q_table[current_state, action])

        current_state = next_state
print("Learned Q-table:")
print(Q_table)
```

# Evaluación y Prueba

En este proyecto, se buscó determinar la propina de un mesero utilizando un dataset obtenido a través de web mining. A continuación, se describen las técnicas de Machine Learning (ML) aplicadas y se analizan los resultados obtenidos con cada una de ellas. Los modelos utilizados incluyen técnicas de aprendizaje supervisado, no supervisado y por refuerzo.

## Extracción y Preparación de Datos

El dataset fue extraído del sitio [Kaggle](https://www.kaggle.com/datasets/aminizahra/tips-dataset) utilizando Selenium para la navegación y BeautifulSoup para el análisis del HTML. La información obtenida fue organizada en un DataFrame de Pandas, convirtiendo todas las variables en numéricas para facilitar su procesamiento por los modelos de ML.

## Modelos de Aprendizaje Supervisado

- 1. **K-Nearest Neighbors (KNN)**
  - Entrenamiento y Evaluación: El modelo KNN se entrenó con 75% de los datos y se probó con el 25% restante.
  - Precisión: El modelo obtuvo un score de 0.69 en el conjunto de prueba.
  - Conclusión: KNN mostró un rendimiento moderado, pudiendo predecir la propina justa en aproximadamente el 69 % de los casos.
- **2. Gaussian Naive-Bayes**
  - Entrenamiento y Evaluación: Se utilizó el mismo conjunto de datos de entrenamiento y prueba que en KNN.
  - Precisión: El modelo también obtuvo un score de 0.69 en el conjunto de prueba.
  - Conclusión: Gaussian Naive-Bayes tuvo un rendimiento similar al de KNN, lo que indica una capacidad comparable para la predicción de propinas.
- **3. Support Vector Machines (SVM)**

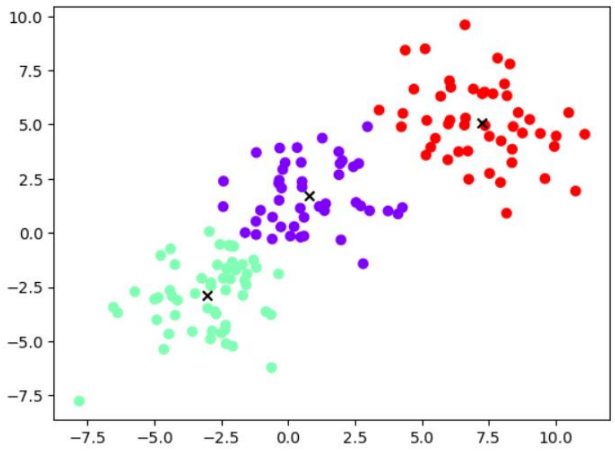
- Entrenamiento y Evaluación: Se aplicaron los mismos datos de entrenamiento y prueba.
- Precisión: El modelo SVM obtuvo un score de 0.77.
- Conclusión: SVM mostró un rendimiento superior a KNN y Naive-Bayes, logrando una mejor precisión en la predicción de propinas justas.

Resultados de desempeño Aprendizaje Supervisado:

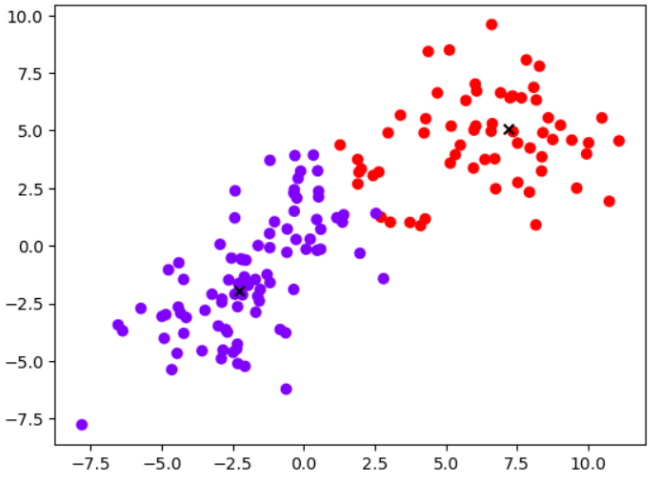
| Modelo | Score |
|---|---|
| Gaussian Naive-Bayes | 0.6923 |
| Support Vector Machines | 0.7692 |
| K-Nearest Neighbors | 0.6923 |

## Modelos de Aprendizaje No Supervisado

- **1. K-Means Clustering**
  - Aplicación y Resultados: Se agruparon los datos en 3 clusters utilizando el algoritmo K-Means.
  - Interpretación: La gráfica resultante mostró una buena separación de los datos en función de características similares, del dataset podemos interpretar que a partir de lo que se paga por persona individual la calidad de la propina se puede dividir en 3: los primeros de un precio por persona bajo indican una propina baja, aquellos con un precio razonable propician una propina moderada y a un mayor gasto por persona en cada mesa, se puede esperar una mejor propina.




- **2. Mean Shift Clustering**
  - Aplicación y Resultados: Se aplicó el algoritmo Mean Shift, resultando en una agrupación de los datos en varios clusters. Mediante MeanShift se pueden distinguir dos grupos: los de propina baja y propina alta. Esto corrobora que un mayor precio individual propicia una mejor propina.




  - Interpretación: Mean Shift identificó patrones en los datos, pero requiere un análisis detallado para comprender la naturaleza de cada grupo y su relación con las propinas.

## Aprendizaje por Refuerzo

Configuración y Resultados: Se implementó un modelo de aprendizaje por refuerzo con una tabla Q que aprendió a maximizar la recompensa para alcanzar un estado objetivo.

Conclusión: Este enfoque demostró ser útil para escenarios donde se busca optimizar una métrica específica, aunque requiere un mayor número de iteraciones y ajustes en los parámetros para mejorar su precisión.

# Conclusiones

El aprendizaje no supervisado es una rama crucial del aprendizaje automático que permite a los modelos descubrir estructuras y patrones en grandes conjuntos de datos sin la necesidad de etiquetas o guías explícitas. Este aprendizaje es fundamental en la exploración inicial de datos, permitiendo a científicos de datos y analistas identificar características relevantes y reducir la dimensionalidad de los datos, algo útil para modelos supervisados posteriores.

En el ámbito del análisis de datos en la web, el Web Mining se destaca como una herramienta poderosa que aprovecha la vasta cantidad de información disponible en internet para extraer conocimiento valioso. Web Mining se divide en tres áreas principales: Web Content Mining, Web Structure Mining y Web Usage Mining. Cada una de estas áreas aborda diferentes aspectos de la web, desde el contenido y su estructura hasta los patrones de uso de los usuarios, proporcionando una comprensión integral y multifacética del comportamiento en línea y las tendencias del mercado.

El algoritmo Gaussian Naive Bayes, basado en la suposición de que los datos siguen una distribución normal, es una herramienta eficaz para la clasificación de datos con atributos continuos. Este método se utiliza por su simplicidad y eficiencia, dando resultados robustos en diversas aplicaciones prácticas.

Por otro lado, el algoritmo K-Means es uno de los métodos de agrupamiento más reconocidos en el aprendizaje no supervisado. Su capacidad para dividir grandes conjuntos de datos en grupos o clústeres similares es invaluable en la identificación de patrones y tendencias dentro de datos masivos. Este algoritmo iterativo ajusta continuamente los centroides de los clústeres para mejorar la homogeneidad dentro de cada grupo, permitiendo así una mejor interpretación y análisis de los datos.

En la implementación práctica de estos métodos, como en el análisis de peligros en diversas ciudades a partir de datos web, el uso de técnicas de

aprendizaje no supervisado como PCA (Análisis de Componentes Principales) y clustering jerárquico, además de K-Means, permite una comprensión profunda y detallada de los datos recolectados. Estas técnicas facilitan la extracción de información relevante y la identificación de patrones significativos, mejorando así la capacidad de toma de decisiones basada en datos.

En resumen, el aprendizaje no supervisado y el Web Mining son herramientas esenciales en el análisis de datos moderno. Permiten la exploración y el descubrimiento de patrones ocultos en grandes volúmenes de datos no etiquetados, optimizando tanto la comprensión como la utilización de la información para diversas aplicaciones, desde la predicción de comportamientos hasta la mejora de experiencias de usuario y la identificación de tendencias de mercado.

## Referencias


- *¿Qué es el aprendizaje supervisado? | IBM*. (s. f.). https://www.ibm.com/es-es/topics/supervised-learning
- *¿Qué es el aprendizaje no supervisado? | IBM*. (s. f.). https://www.ibm.com/mx-es/topics/unsupervised-learning
- Rataplansky. (2024, 11 febrero). *Aprendizaje no supervisado: Una guía completa sobre esta técnica de aprendizaje automático - Prompts para IA*. Prompts Para IA. https://prompt.uno/aprendizaje-automatico/aprendizaje-no-supervisado/
- Fernández, A. (2023, 14 abril). *¿Qué es Web Mining? ¿Para qué sirve?* ingenieroSEO. https://albertofdez.com/blog/seo/que-es-web-mining-para-que-sirve/
- *SPSS Statistics Subscription - Classic*. (s. f.). https://www.ibm.com/docs/es/spss-statistics/saas?topic=analysis-hierarchical-cluster-statistics
- *¿Qué son los clasificadores Naive Bayes? | IBM*. (s. f.). https://www.ibm.com/es-es/topics/naive-bayes
- *¿Qué es KNN? | IBM*. (s. f.). https://www.ibm.com/mx-es/topics/knn
- Sanz, F. (2023, 22 marzo). *Algoritmo K-Means Clustering – aplicaciones y desventajas*. The Machine Learners. https://www.themachinelearners.com/k-means/
- Ramírez, L. (2023, 5 enero). *Algoritmo k-means: ¿Qué es y cómo funciona?* Thinking For Innovation. https://www.iebschool.com/blog/algoritmo-k-means-que-es-y-como-funciona-big-data/
- Admin. (2019a, marzo 11). *K-Means Clustering: Agrupamiento con Minería de datos*. ESTRATEGIAS DE TRADING. https://estrategiastrading.com/k-means/

- Tanner, G. (s. f.). *Mean shift*. Machine Learning Explained. https://ml-explained.com/blog/mean-shift-explained
- GeeksforGeeks. (2024a, marzo 21). *Q-Learning in Python*. GeeksforGeeks. https://www.geeksforgeeks.org/q-learning-in-python/
- *sklearn.neighbors.KNeighborsClassifier*. (s. f.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
- *sklearn.naive_bayes.GaussianNB*. (s. f.-b). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html
- GeeksforGeeks. (2023, 1 septiembre). *Classifying data using Support Vector Machines(SVMs) in Python*. GeeksforGeeks. https://www.geeksforgeeks.org/classifying-data-using-support-vector-machinessvms-in-python/