MACHINE LEARNING

Indice

1. Introduccion
2. Aprendizaje Supervisado
    a. Clasificacion
        i. Modelos
        ii. Metricas y evaluaciones
    b. Regresion
        i. Modelos
        ii. Metricas y evaluaciones
3. Aprendizaje no Supervisado
4. Bibliografia

Introduccion

El aprendizaje automático es un subcampo de la inteligencia artificial que se enfoca en desarrollar algoritmos y modelos, capaces de aprender a partir de los datos y generar predicciones o tomar decisiones basadas en estos. Estos modelos pueden analizar grandes cantidades de datos, identificar patrones y realizar tareas específicas sin necesidad de una programación específica.

El aprendizaje automático se divide en varias categorías:

1. Aprendizaje supervisado.
2. Aprendizaje no supervisado.
3. Aprendizaje por refuerzo.

Aprendizaje supervisado

El aprendizaje supervisado es una técnica de aprendizaje automático en la cual se utiliza un conjunto de datos etiquetados para entrenar un modelo y realizar predicciones (algoritmos de regresion) o clasificaciones (algoritmos de clasificacion) sobre nuevos datos. En el aprendizaje supervisado, el conjunto de datos de entrenamiento consiste en pares de entrada y salida, donde la entrada es un conjunto de características o atributos y la salida es la etiqueta o el valor objetivo correspondiente. El objetivo del modelo de aprendizaje supervisado es aprender la relación entre las características de entrada y las salidas para poder realizar predicciones precisas sobre nuevos datos no vistos previamente.

Estos pueden ser utilizados para realizar predicciones sobre etiquetas de clase o variables numericas.

Algoritmos de clasificacion

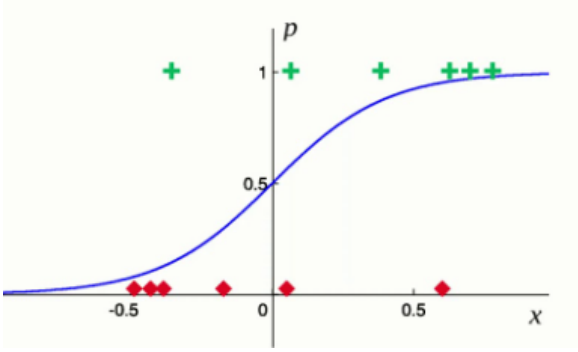
En estos se busca clasificar etiquetas, por ej. si un mail es spam o no; dentro de estos podemos mencionar:

1) Regresion logistica: Es un metodo de regresion utilizado para resolver problemas de variables binarias; el objetivo de la regresión logística es predecir la probabilidad de que una instancia pertenezca a una clase específica. La variable dependiente o la variable objetivo en la regresión logística se codifica como una variable binaria, donde generalmente se utiliza la codificación 0 para una clase y 1 para la otra clase.

El algoritmo de regresión logística utiliza una función logística (también conocida como función sigmoide) para modelar la relación entre las variables independientes y la variable dependiente. La función logística toma una combinación lineal de las variables independientes y la transforma en un valor entre 0 y 1, que representa la probabilidad de pertenecer a una clase. Dado un conjunto de características, el modelo de regresión logística calcula la probabilidad de pertenecer a una clase específica y puede asignar una clase basada en un umbral de probabilidad.

Su formula y grafica:

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



¿Como se interpeta esta funcion?

En la función logística, a medida que la entrada se acerca a infinito positivo, la salida de la función se aproxima a 1, lo que indica una alta probabilidad de pertenecer a la clase positiva. Por otro lado, a medida que la entrada se acerca a infinito negativo, la salida de la función se aproxima a 0, lo que indica una baja probabilidad de pertenecer a la clase positiva.

El umbral de decisión de 0.5 se utiliza comúnmente para asignar clases en la regresión logística. Si la probabilidad calculada por la función logística es mayor que 0.5(50% de probabilidad), se asigna la clase positiva (1). Si la probabilidad es menor que 0.5, se asigna la clase negativa (0).

- Implementacion con sklearn: https://github.com/fowardelcac/Machine-learning-con-sklearn/blob/main/Supervisado/Clasificacion/a_Regresion_logistica.py

- Arbol de desicion

Algoritmo utilizado tanto para problemas de clasificación como de regresión. Es una estructura en forma de árbol compuesta por nodos y ramas, donde cada nodo representa una característica o atributo, y cada rama representa una decisión o una conexión entre características.

El proceso de construcción de un árbol de decisión implica dividir el conjunto de datos en función de las características más importantes. El algoritmo busca la característica que proporciona la mayor ganancia de información o la reducción de la impureza en el conjunto de datos. La impureza se refiere a la mezcla de diferentes clases en un conjunto de datos y puede medirse utilizando métricas como la entropía o el índice de Gini

El árbol de decisión se construye de forma recursiva dividiendo los datos en subconjuntos más pequeños hasta que se alcance un criterio de parada, como alcanzar una profundidad máxima,

una pureza mínima en los nodos o un número mínimo de muestras requeridas en los nodos terminales.

El proceso de predicción implica seguir las ramas del árbol de decisión basándose en las características de los datos hasta llegar a un nodo terminal, donde se asigna una clase o se produce una salida numérica.

Los árboles de decisión tienen varias ventajas, como la capacidad de manejar datos numéricos y categóricos, la interpretabilidad y la capacidad de capturar relaciones no lineales entre las características. Sin embargo, también pueden ser propensos al sobreajuste y pueden no ser adecuados para conjuntos de datos muy grandes o con alta dimensionalidad.

Entropia y indice de gini

Supongamos que tenemos una caja de juguetes y cada juguete tiene un color, si queremos saber qué tan mezclados están los colores en la caja, podemos hacerlo de dos formas diferentes: la entropía y el índice de Gini.

La entropía es una *medida de desorden en la caja*. Si todos los juguetes son del mismo color, la entropía es baja y significa que la caja está muy ordenada. Pero si los juguetes tienen muchos colores diferentes y están mezclados, la entropía es alta y significa que la caja está desordenada. En otras palabras, *la entropía mide qué tan incierto o variado es el contenido de la caja.*

Por otro lado, el índice de Gini *es una medida de la impureza en la caja.* Si todos los juguetes son del mismo color, el índice de Gini es bajo y significa que la caja está muy ordenada y pura. Pero si los juguetes tienen muchos colores diferentes y están mezclados, el índice de Gini es alto y significa que la caja está impura. En resumen, el índice de Gini *mide qué tan mezclados están los colores en la caja y qué tan impura es la mezcla.*

La principal diferencia entre la entropía y el índice de Gini radica en cómo se calculan. *La entropía se basa en el concepto de la teoría de la información y utiliza la proporción de cada color en la caja para calcular qué tan incierto es el contenido*. *El índice de Gini, por otro lado, se basa en la probabilidad de que dos juguetes seleccionados al azar tengan diferentes colores y utiliza la proporción de cada color para calcular qué tan impura es la mezcla.*

En cuanto a la aplicación en los árboles de decisión, tanto la entropía como el índice de Gini se utilizan como criterios para decidir cómo dividir los datos en los nodos del árbol. El objetivo es seleccionar la división que reduzca la entropía o el índice de Gini de manera más significativa, lo que se traduce en obtener grupos más puros y discriminantes en el árbol de decisión.

Tanto la entropía como el índice de Gini son medidas válidas para evaluar la impureza en los árboles de decisión, y la elección entre ellos puede depender del problema y las preferencias del usuario.

- Implementacion con sklearn: https://github.com/fowardelcac/Machine-learning-con-sklearn/blob/main/Supervisado/Clasificacion/b_Arbol_de_desicion.py

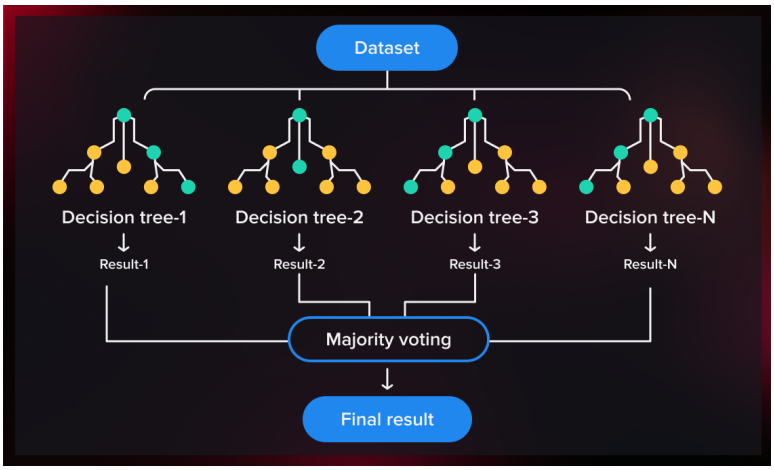
- Random Forest

Se encuentra formado por distintos arboles de desicion los cuales realizan sus propias predicciones; este enfoque se basa en combinar las predicciones de varios modelos de árboles de decisión para obtener una predicción final más precisa.

El ensamble de árboles de decisión se basa en la idea de que varios modelos individuales, aunque imperfectos por sí mismos, pueden combinar sus fortalezas y compensar sus debilidades. Cada árbol de decisión se entrena de forma independiente en un subconjunto de los datos disponibles y genera una predicción individual. Luego, las predicciones de todos los árboles se combinan mediante votación, donde la clase con más votos se considera la predicción final del modelo ensamblado.

La razón por la cual este enfoque puede ser más preciso que un solo árbol de decisión radica en que los árboles individuales pueden cometer errores, pero es menos probable que cometan los mismos errores en la misma dirección todo el tiempo. Al combinar las predicciones de múltiples árboles, los errores individuales tienden a "cancelarse" entre sí, mientras que las predicciones correctas se refuerzan. Esto puede llevar a una predicción final más precisa y robusta.

Tanto los arboles de desicion como el algoritmo Random forest, ambos pueden ser utilizados para clasificacion tanto como para problemas de regresion.



- Implementacion con sklearn: https://github.com/fowardelcac/Machine-learning-con-sklearn/blob/main/Supervisado/Clasificacion/c_Random_forest.py

- KNN

El algoritmo KNN se puede utilizar tanto para problemas de clasificación como para problemas de regresión. En el caso de la clasificación, KNN asigna una etiqueta a una nueva instancia en función de las etiquetas de las instancias vecinas más cercanas. La etiqueta más común entre los vecinos determina la clase a la que se asigna la instancia.

Es un algoritmo no paramétrico, lo que significa que no hace suposiciones sobre la distribución de los datos, en cambio, se basa en las instancias de entrenamiento para realizar las predicciones.

También es importante destacar que KNN es un algoritmo basado en instancias. En lugar de realizar un proceso de entrenamiento explícito para aprender un modelo, KNN utiliza todo el conjunto de entrenamiento como su base de conocimiento. Cuando se realiza una predicción, se busca en el conjunto de entrenamiento las instancias más cercanas a la instancia a clasificar o regresar, y se basa en ellas para realizar la predicción.

El valor de K(número de vecinos más cercanos) es un parámetro importante en el algoritmo KNN y debe ser elegido cuidadosamente. Un valor pequeño de K puede resultar en un modelo más sensible al ruido o a valores atípicos en los datos, mientras que un valor grande de K puede suavizar las fronteras de decisión y producir un sesgo hacia las clases mayoritarias en el conjunto de datos.

¿Como se calcula la distancia? La distancia más comúnmente utilizada es la distancia euclidiana, pero también se pueden utilizar otras medidas de distancia, como la distancia de Manhattan o la distancia de Minkowski.

Distancia Euclidiana

$$\sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}$$

El funcionamiento básico del algoritmo KNN es el siguiente:

1. Selecciona un valor de K, que representa el número de vecinos más cercanos que se tomarán en cuenta para realizar una predicción.
2. Calcula la distancia entre la instancia a clasificar y todas las demás instancias del conjunto de entrenamiento.
3. Selecciona los K vecinos más cercanos según la distancia calculada en el paso anterior.
4. Para el caso de la clasificación, se asigna a la instancia a clasificar la etiqueta más común entre sus vecinos cercanos. En otras palabras, se utiliza una votación mayoritaria para determinar la clase de la instancia

- Implementacion con sklearn: https://github.com/fowardelcac/Machine-learning-con-sklearn/blob/main/Supervisado/Clasificacion/d_KNN.py

Metricas utilizadas

Precisión (Accuracy): Mide la proporción de instancias correctamente clasificadas sobre el total de instancias. Es una medida general de la precisión del modelo, pero puede ser engañosa si hay clases desbalanceadas.

$$Accuracy = \frac{Predicciones\ correctas}{Total\ de\ Predicciones}$$

Matriz de Confusión (Confusion Matrix): Es una tabla que muestra el número de instancias clasificadas correctamente e incorrectamente para cada clase. A partir de la matriz de confusión, se pueden derivar otras métricas como:

- Verdaderos Positivos (True Positives, TP): Número de instancias correctamente clasificadas como positivas.
- Verdaderos Negativos (True Negatives, TN): Número de instancias correctamente clasificadas como negativas.
- Falsos Positivos (False Positives, FP): Número de instancias incorrectamente clasificadas como positivas.
- Falsos Negativos (False Negatives, FN): Número de instancias incorrectamente clasificadas como negativas.

TN	FP
FN	TP

Precisión (Precision): Mide la proporción de instancias positivas correctamente clasificadas sobre el total de instancias clasificadas como positivas. Indica cuán confiable es el modelo al clasificar las instancias positivas.

$$Precision = \frac{Verdaderos\ Positivos}{Verdaderos\ Positivos + Falsos\ Positivos}$$

Exhaustividad o Sensibilidad (Recall o Sensitivity): Mide la proporción de instancias positivas correctamente clasificadas sobre el total de instancias realmente positiva. Indica cuántas instancias positivas se capturan correctamente por el modelo.

$$Recall = \frac{Verdaderos\ Positivos}{Verdaderos\ Positivos + Falsos\ Negativos}$$

F1-Score: Es una métrica que combina la precisión y la exhaustividad en una sola medida. Es especialmente útil cuando hay un desbalance entre las clases.

$$F1\ score = \frac{2}{\frac{1}{precision} + \frac{1}{recall}}$$

Algoritmos de Regresion

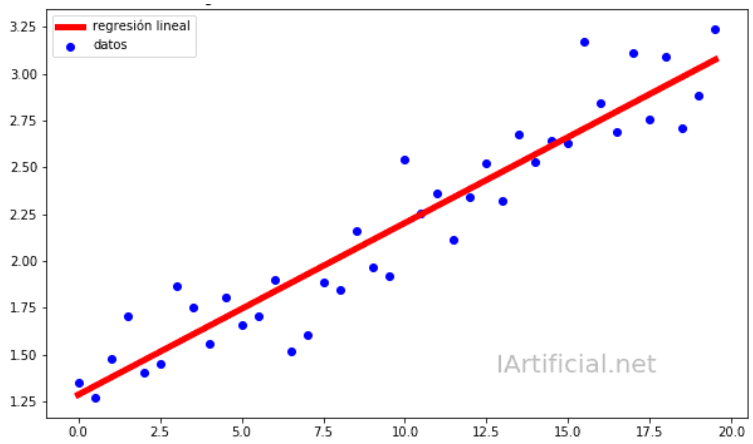
La variable a predecir es un numero, por ej: estimar el precio de distintas casas. Dentro de estos podemos mencionar:

- Regresion lineal

Algoritmo utilizado para predecir una variable dependiente continua a partir de una o más variables independientes. El objetivo de la regresión lineal es encontrar la mejor línea recta que se ajuste a los datos y minimice la diferencia entre los valores observados y los valores predichos.

En la regresión lineal, se asume que existe una relación lineal entre las variables independientes (variables predictoras) y la variable dependiente (la variable que se desea predecir). Esta relación lineal se expresa mediante una ecuación de la forma:

$$y = wx + b$$



El proceso de entrenamiento de un modelo de regresión lineal implica encontrar los valores óptimos para los coeficientes que minimizan la diferencia entre los valores observados y los valores predichos. Esto se puede lograr mediante métodos como el método de mínimos cuadrados o el descenso de gradiente.

Existen variantes de la regresión lineal, como la regresión lineal múltiple, que permite trabajar con múltiples variables independientes, y técnicas de regularización como la regresión ridge o la regresión Lasso, que ayudan a evitar el sobreajuste del modelo.

- Implementacion en sklearn: https://github.com/fowardelcac/Machine-learning-con-sklearn/blob/main/Supervisado/Regresion/a_Regresion_lineal.py

Ademas, podemos incluir algoritmos como los Arboles de desicion, Random forest, KNN, SVM etc.

Metricas utilizadas

Error cuadrático medio (MSE): El MSE calcula la media de los errores al cuadrado entre las predicciones del modelo y los valores reales. Cuanto menor sea el valor de MSE, mejor será el ajuste del modelo a los datos.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

Raíz del error cuadrático medio (RMSE): El RMSE es la raíz cuadrada del MSE y proporciona una medida de la discrepancia promedio entre las predicciones del modelo y los valores reales. Al igual que el MSE, se busca minimizar el valor del RMSE.

$$\mathrm{RMSE}(y, \hat{y}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Error absoluto medio (MAE): El MAE calcula la media de los errores absolutos entre las predicciones del modelo y los valores reales. A diferencia del MSE, el MAE no tiene en cuenta la magnitud de los errores, solo su promedio absoluto.

$$\mathrm{MAE}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Coeficiente de determinación (R^2): El coeficiente de determinación, también conocido como R cuadrado, proporciona una medida de qué tan bien se ajusta el modelo a los datos en comparación con un modelo de línea base. R^2 varía entre 0 y 1, donde 0 indica que el modelo no explica la variabilidad de los datos y 1 indica que el modelo se ajusta perfectamente a los datos. Un valor más alto de R^2 indica un mejor ajuste del modelo.

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

Donde $\bar{y}$ es la media de los valores reales de la variable dependiente.

Aprendizaje no supervisado

A diferencia del aprendizaje supervisado, donde se proporciona un conjunto de datos etiquetados para entrenar al modelo, el aprendizaje no supervisado se enfoca en extraer patrones, estructuras o características ocultas en conjuntos de datos sin la guía de etiquetas o respuestas predefinidas.

Algunos ejemplos comunes de algoritmos de aprendizaje no supervisado incluyen:

1. Clustering: Estos algoritmos agrupan los datos en grupos o clústeres basados en la similitud de las características. Algunos algoritmos populares son el K-means, DBSCAN y el algoritmo de agrupamiento jerárquico.
2. Reducción de dimensionalidad: Estos algoritmos buscan reducir la cantidad de variables o características en un conjunto de datos, preservando al mismo tiempo la información relevante. Principal Component Analysis (PCA) y t-SNE son ejemplos conocidos de algoritmos de reducción de dimensionalidad.

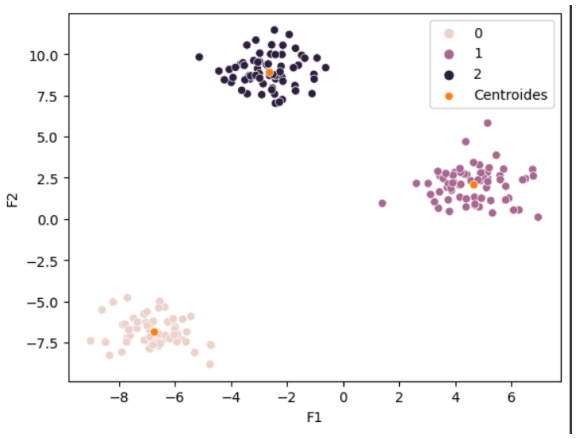
3. Reglas de asociación: Estos algoritmos buscan descubrir relaciones o patrones frecuentes entre diferentes elementos en un conjunto de datos. El algoritmo Apriori es uno de los más conocidos en este campo.
4. Detección de anomalías: Estos algoritmos identifican patrones o instancias inusuales o anómalas en un conjunto de datos. Algunos algoritmos populares son One-Class SVM y el algoritmo de detección de anomalías basado en densidad (DBSCAN).

Clustering

- K-means

Este algoritmo busca particionar los datos en k clusters basados en la similitud de sus caracteristicas, el objetivo del K-Means es minimizar la suma de las distancias entre los puntos y el centroide al que pertenecen. Funciona de la siguiente manera:

1. Inicialización: Se seleccionan k centroides iniciales de forma aleatoria en el espacio de características. Cada centroide representa el centro de un clúster.
2. Asignación de muestras: Cada muestra del conjunto de datos se asigna al clúster cuyo centroide esté más cercano en términos de su distancia, esta se puede calcular utilizando diversas métricas, siendo la distancia euclidiana la más comúnmente utilizada.
3. Actualización de centroides: Se recalcula el centroide de cada clúster utilizando las muestras asignadas a él. El nuevo centroide se define como el promedio de las características de todas las muestras asignadas a ese clúster.
4. Paso 2 y 3 se repiten iterativamente hasta que los centroides converjan o se alcance un criterio de parada predefinido. La convergencia ocurre cuando los centroides ya no cambian significativamente de una iteración a otra.
5. El resultado final es un conjunto de k clústeres, donde cada muestra está asignada a un clúster específico.



- Implementeacion con sklearn: https://github.com/fowardelcac/Machine-learning-con-sklearn/blob/main/No supervisado/a_K_means.py

Reduccion dimensionalidad

- PCA

Cuando en un dataset contamos con una gran cantidad de atributos, podemos decir que es un conjunto de datos con una gran dimensionalidad, lo que puede generar distintos problemas; para esto aplicamos algoritmos de reduccion de dimensionalidad como PCA(analisis de componentes principales).

Su objetivo principal es transformar un conjunto de variables correlacionadas en un nuevo conjunto de variables no correlacionadas llamadas componentes principales.

El algoritmo PCA se compone de los siguientes pasos:

1. **Normalización de los datos:** Si es necesario, los datos se normalizan para que todas las variables tengan la misma escala. Esto es importante ya que PCA es sensible a las escalas de las variables. Se recomienda utilizar StandarScaler ya que PCA es sensible a las escalas de las variables y el escalamiento estándar asegura que todas las variables tengan una media de 0 y una desviación estándar de 1. Esto ayuda a garantizar que todas las variables tengan un impacto similar en la matriz de covarianza y en los componentes principales resultantes.

2. **Cálculo de la matriz de covarianza:** Se calcula la matriz de covarianza que muestra las relaciones de covarianza entre las diferentes variables del conjunto de datos. La covarianza mide cómo varían conjuntamente dos variables.
3. **Cálculo de los autovectores y autovalores:** Se calculan los autovectores y autovalores de la matriz de covarianza. Los autovectores representan las direcciones principales de variabilidad en los datos, y los autovalores indican la cantidad de varianza explicada por cada autovector.
4. **Selección de los componentes principales:** Los autovectores se ordenan en función de sus autovalores asociados, de mayor a menor. Luego, se seleccionan los componentes principales en función de la cantidad de varianza que explican. La suma acumulada de los autovalores normalizados se utiliza a menudo para tomar esta decisión.
5. **Transformación de los datos:** Se proyecta el conjunto de datos original en el espacio de los componentes principales seleccionados. Esto implica multiplicar la matriz de datos por la matriz de autovectores correspondientes a los componentes principales seleccionados.

El resultado final del algoritmo PCA es un nuevo conjunto de variables (los componentes principales) que son combinaciones lineales de las variables originales y que están ordenadas en función de su capacidad para explicar la varianza en los datos. Estas nuevas variables son ortogonales entre sí, lo que significa que están no correlacionadas.

PCA se basa en el supuesto que la varianza es igual a informacion, entonces tenemos que encontrar como covarian estas variables, la covarianza es una medida de cómo dos variables numéricas se mueven en conjunto. Mide la relación lineal entre las variables y muestra si varían en la misma dirección (covarianza positiva) o en direcciones opuestas (covarianza negativa). Una covarianza cercana a cero indica que no hay una relación lineal aparente entre las variables.

El PCA tiene varias aplicaciones, como la reducción de dimensionalidad, la visualización de datos en un espacio de menor dimensión y la eliminación de la multicolinealidad en los datos antes de aplicar otros algoritmos de aprendizaje automático.

Bibliografia

- chat.openai.com
- https://www.iberdrola.com/innovacion/machine-learning-aprendizaje-automatico#:~:text=El Machine Learning es una,elaborar predicciones (análisis predictivo).
- https://es.wikipedia.org/wiki/Aprendizaje_supervisado
- https://aprendeia.com/algoritmo-regresion-logistica-machine-learning-teoria/
- https://www.youtube.com/watch?v=SeM4Rtoa4EU
- Gutiérrez-García, J.O. [Código Máquina]. (2023, 24 de Abril). Entendiendo la Entropía en Machine Learning y Ciencia de Datos con Python [Video]. YouTube. [Incluye aquí la URL del video]
- https://towardsdatascience.com/understanding-random-forest-58381e0602d2
- https://serokell.io/blog/random-forest-classification
- https://www.youtube.com/watch?v=FHHuo7xEeo4
- Gutiérrez-García, J.O. [Código Máquina]. (2022, 20 de Agosto). Métricas para Clasificadores de Machine Learning: Matriz de Confusión, Precision, Accuracy, Recall, F1 [Video]. YouTube. [Incluye aquí la URL del video]
- https://www.iartificial.net/regresion-lineal-con-ejemplos-en-python/
- https://www.freecodecamp.org/espanol/news/aprendizaje-automatico-una-introduccion-al-error-cuadratico-medio-y-las-lineas-de-regresion/
- https://www.youtube.com/watch?v=b1NGM3IbRcI
- Repositorio de mi autoria: https://github.com/fowardelcac/Machine-learning-con-sklearn/tree/main