*Modelos Analíticos para Ciencia de Datos II*

Empecemos repasando algunos conceptos de la sesión anterior. En la clase de Modelos Analíticos para Ciencia de Datos I, comentamos que el Machine Learning, es un método de análisis de datos que automatiza la construcción de “Modelos Analíticos”. Entonces dentro de este contexto, el ML son diferentes algoritmos que nosotros podemos desarrollar, en función de la problemática del negocio como así también, del tipo de aprendizaje que queramos aplicar.

En esta sesión, vamos a profundizar en el Tipo de Aprendizaje Supervisado, sus características, particulares y los principales algoritmos que podemos encontrar.

Recordemos …. *¿Cuál es el objetivo principal del Aprendizaje Supervisado?*  🤔

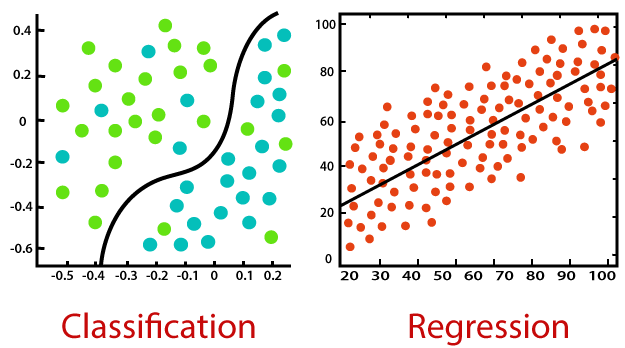
Predecir las respuestas que habrá en el futuro, gracias al entrenamiento del algoritmo con datos conocidos del pasado (datos históricos).

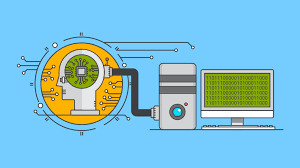
**Aprendizaje Supervisado**

Los problemas que resuelve el aprendizaje supervisado de forma general son de dos tipos:

* **Problemas de clasificación:** Se trata de problemas que necesitan predecir la clase más probable de un elemento, en función de un conjunto de variables de entrada. Para este tipo de algoritmos, la variable target o respuesta es una variable de **tipo categórica**.
* **Problemas de regresión**: Los algoritmos de regresión en vez de predecir categorías, predicen valores numéricos. Es decir, la variable target en un problema de regresión es de **tipo cuantitativa**.

Entonces, ¿*Cómo sé si tengo que utilizar un algoritmo de clasificación o de regresión?* Depende del tipo de problema que plantea mi variable a predecir.



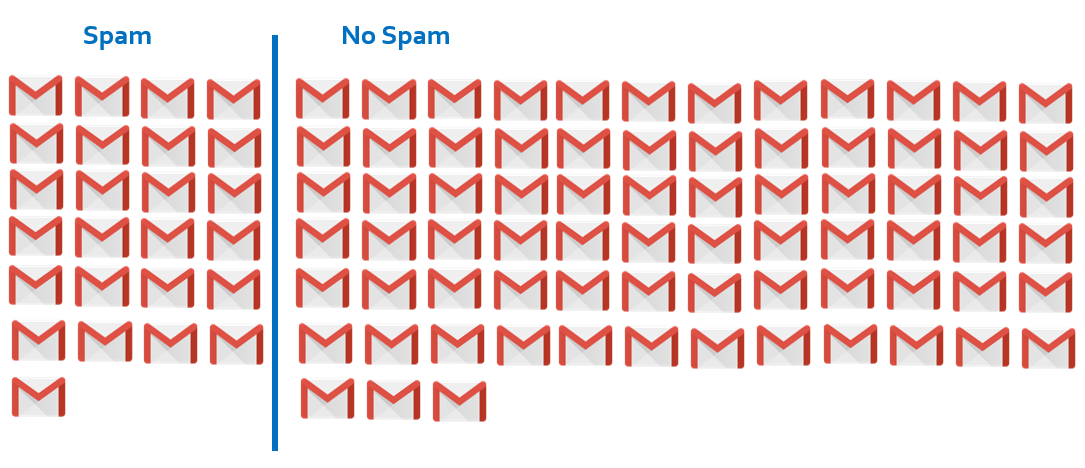
Como hemos comentado anteriormente, los algoritmos de clasificación intentan predecir una categoría, por ejemplo:

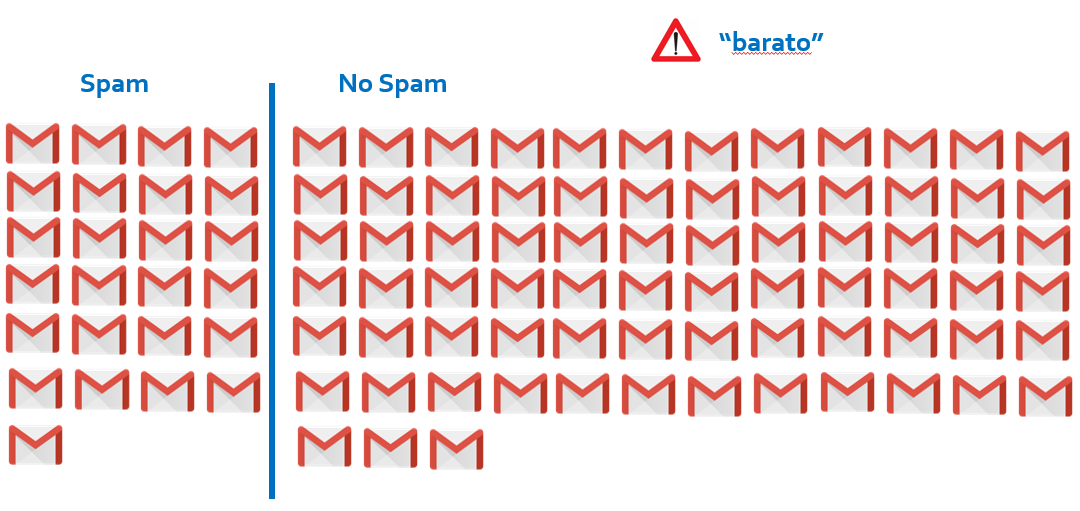
* Enfermo/No enfermo.
* Sobrevive/No sobrevive.
* Baja/ No baja, entre otros.

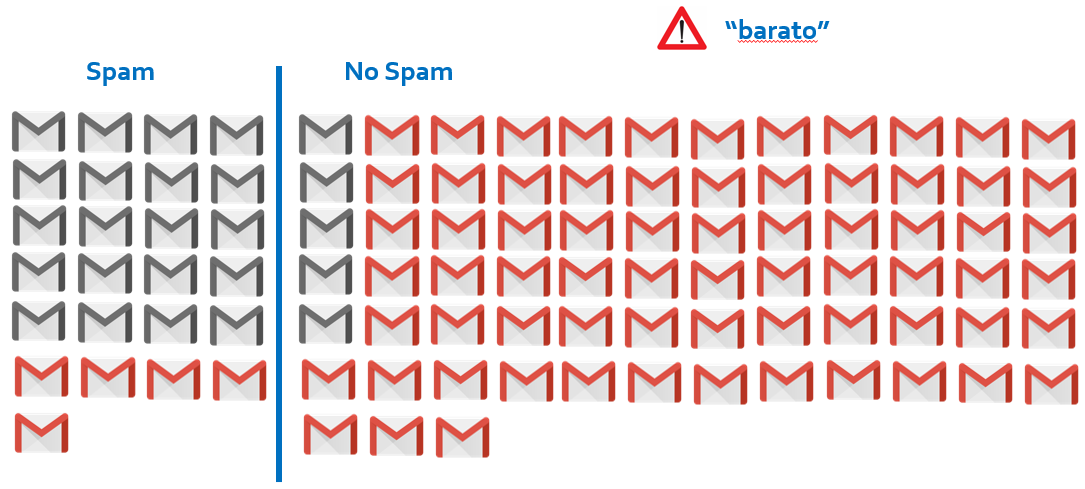
**Ejemplo de un Algoritmo Clasificador de correos Spam**

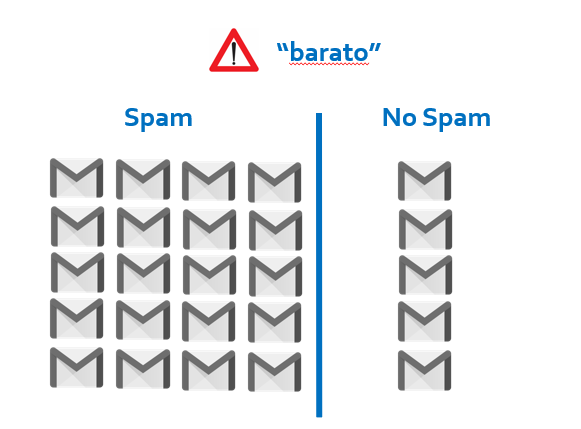
**Objetivo**: Clasificar los correos que llegan a la bandeja en Spam y No Spam

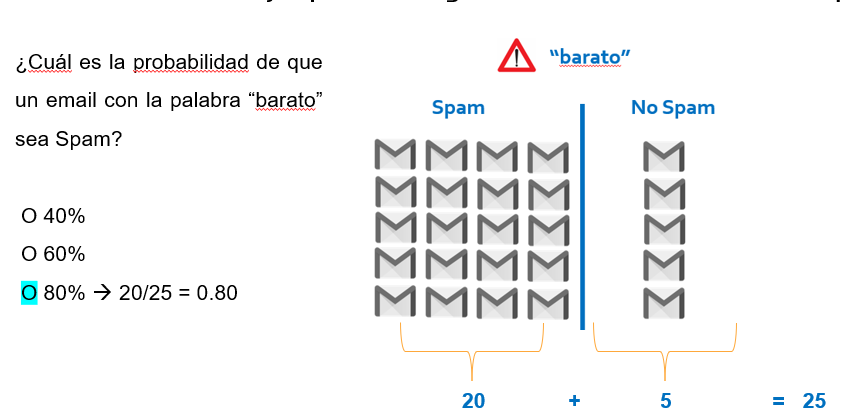










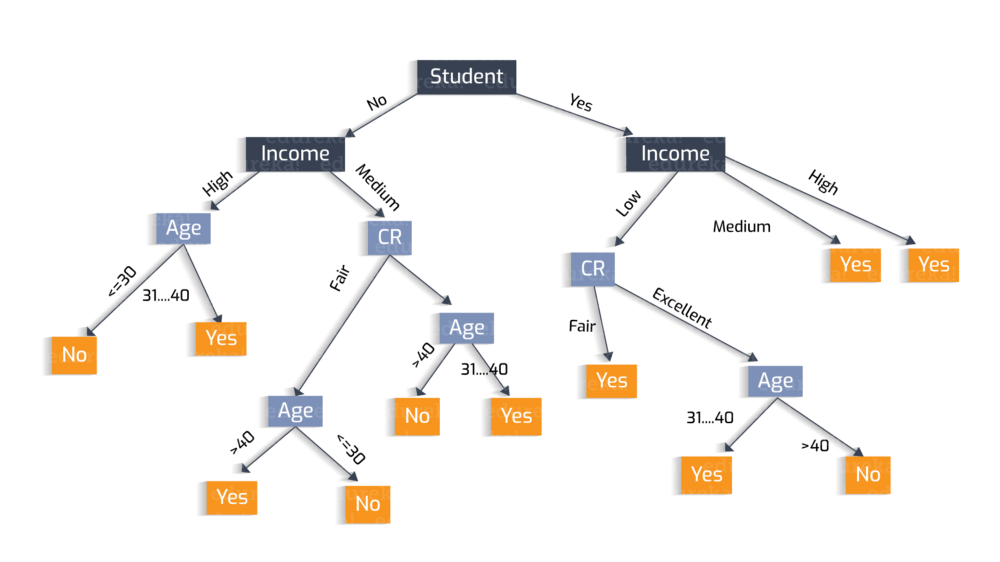


Existen múltiples algoritmos de clasificación, en esta sección comentaremos aquellos más populares. Resulta importante mencionar, que en el Módulo 5 del curso titulado: *Algoritmos de Machine Learning*, abordaremos esta temática en mayor profundidad. Empecemos 😊

**Árboles de Decisión:**

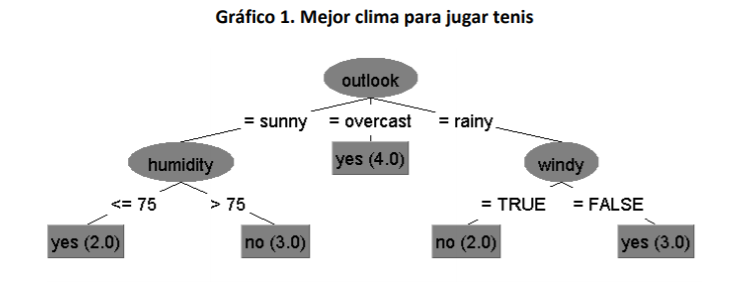
El aprendizaje basado en árboles de decisión está ampliamente extendido en la actualidad, y múltiples modelos hacen diferentes implementaciones de los mismos.

Las primeras versiones de estos modelos fueron implementados por Leo Breiman, que desarrolló algoritmos como ID3, C4.5, random forest, entre otros. Resulta importante mencionar que los árboles de decisión, se utilizan para problemas de **Clasificación** y **Regresión**.



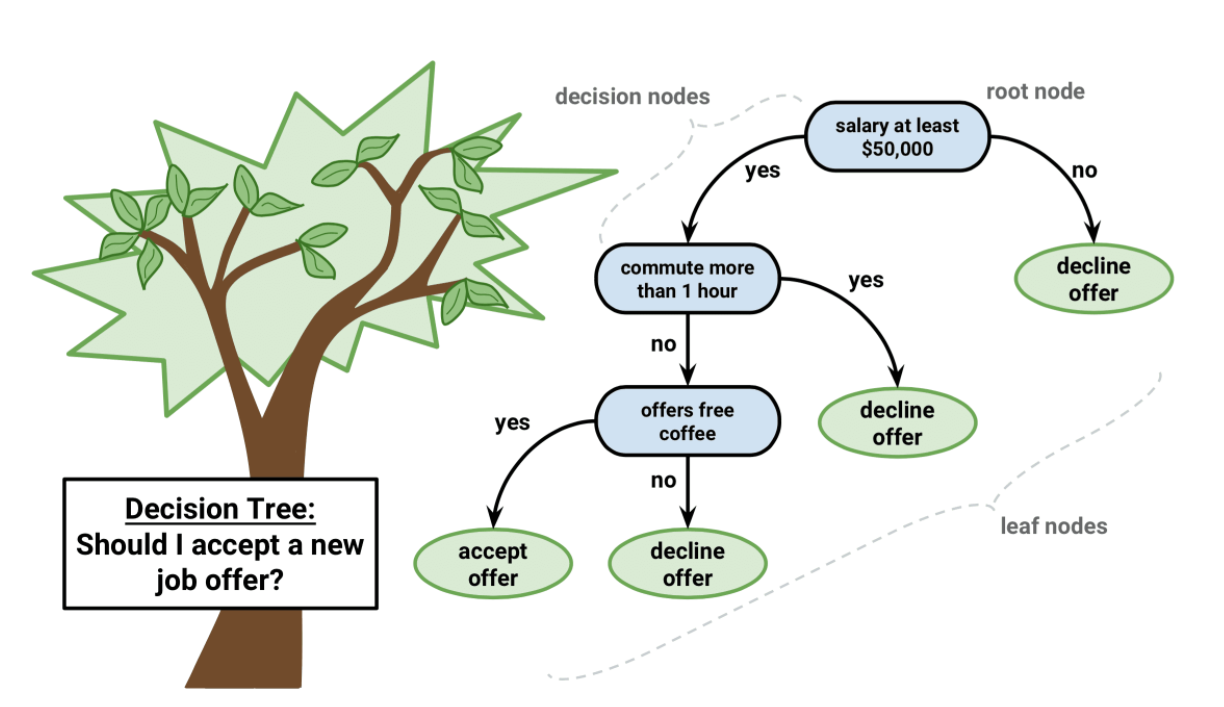
Los árboles de decisión aprenden de los datos generando reglas de tipo ***if-else***. Van separando la data en grupos cada vez más pequeños de subsets de mi dataset original. Cada división se la conoce con el nombre de nodo. Cuando un nodo no conduce a nuevas divisiones se le denomina hoja, para luego ser considerada como ramas del árbol.

*Ejemplo:*



*Terminología:*

* **Nodo raíz**: Representa a toda la población o muestra y esto se divide en dos o más conjuntos homogéneos.
* **División**: Es un proceso de división de un nodo en dos o más subnodos.
* **Nodo de decisión:** Cuando un subnodo se divide en subnodos adicionales, se llama nodo de decisión.
* **Nodo de hoja / terminal:** Los nodos sin hijos (sin división adicional) se llaman Hoja o nodo terminal.
* **Poda (Pruning):** Consiste en la reducción del tamaño de los árboles de decisión eliminando nodos.
* **Rama / Subárbol**: Una subsección del árbol de decisión se denomina rama o subárbol.
* **Nodo padre e hijo**: Un nodo, que se divide en subnodos se denomina nodo principal de subnodos, mientras que los subnodos son hijos de un nodo principal.



*Ventajas del Algoritmo*:

* Caja blanca (conjunto de reglas con booleanos), sus resultados son fáciles de entender e interpretar.
* Relativamente robusto.
* Funcionan relativamente bien con grandes conjuntos de datos.
* Combinaciones de los mismos pueden dar resultados muy certeros sin perder explicabilidad, por ejemplo, Random Forest.

*Desventajas del Algoritmo*:

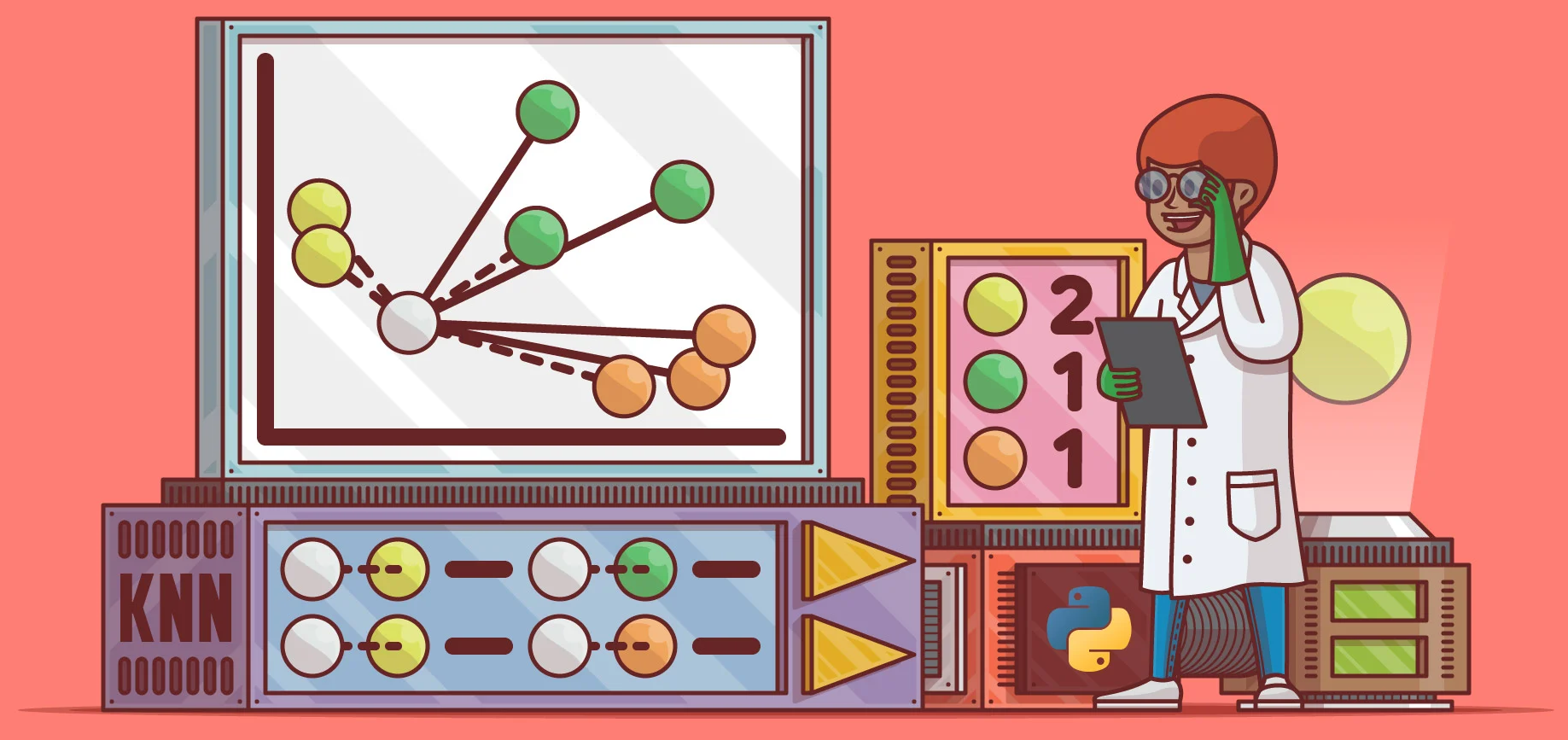
* Tienden al sobreajuste u overfitting de los datos, por lo que el modelo al predecir nuevos casos no estima con el mismo índice de acierto.
* Se ven influenciadas por los outliers, creando árboles con ramas muy profundas que no predicen bien para nuevos casos.
* Crear árboles demasiado complejos puede conllevar que no se adapten bien a los nuevos datos.
* Se pueden crear árboles sesgados si una de las clases es más numerosa que otra es decir, si hay desbalance de clases.

*Link de Interés*:

* https://sitiobigdata.com/2019/12/14/arbol-de-decision-en-machine-learning-parte-1/
* <https://www.iartificial.net/arboles-de-decision-con-ejemplos-en-python/#:~:text=Los%20%C3%A1rboles%20de%20decisi%C3%B3n%20son,(las%20ramas)%20representan%20soluciones>.
* <https://www.cienciadedatos.net/documentos/py07_arboles_decision_python.html>

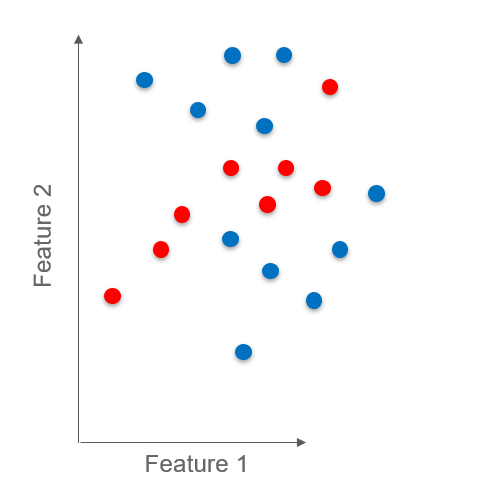
**KNN: K-Nearest-Neighbor (Vecinos Cercanos)**

Este algoritmo, puede usarse para clasificar nuevas muestras (valores discretos) o para predecir (regresión, valores continuos). Sirve esencialmente para clasificar valores, buscando los puntos de datos “más similares” (por cercanía) aprendidos en la etapa de entrenamiento del modelo y haciendo conjeturas de nuevos puntos basado en esa clasificación o regresión.

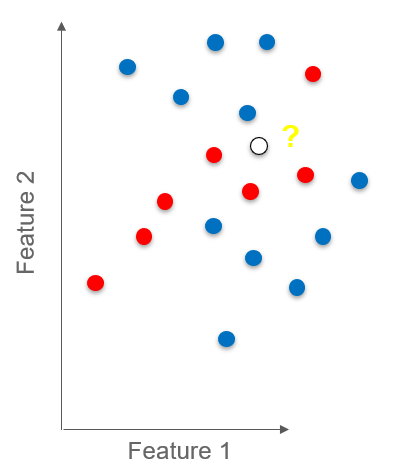


**Concepto**: La idea es realmente sencilla, el algoritmo clasifica cada dato nuevo en el grupo que corresponda, según tenga k vecinos más cerca de un grupo o de otro. Es decir, calcula la distancia del elemento nuevo a cada uno de los existentes, y ordena dichas distancias de menor a mayor para ir seleccionando el grupo al que pertenecer.

Entonces, supongamos el siguiente escenario: Tenemos un Dataset con 2 Features, en el cual cada instancia puede pertenecer a una de dos clases: “Rojo” o “Azul”.

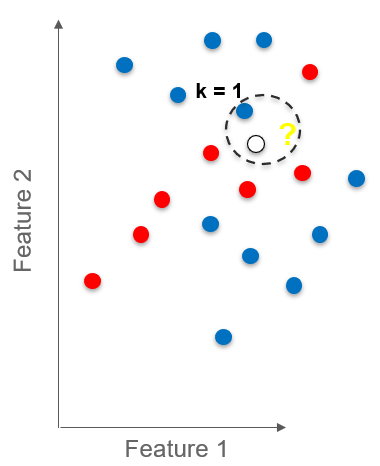


Dada una nueva instancia, de la cual no sabemos cuál es su clase, vamos a recurrir a sus vecinos cercanos para clasificarla. La pregunta seria entonces, ¿La clasificamos como rojo o como azul?



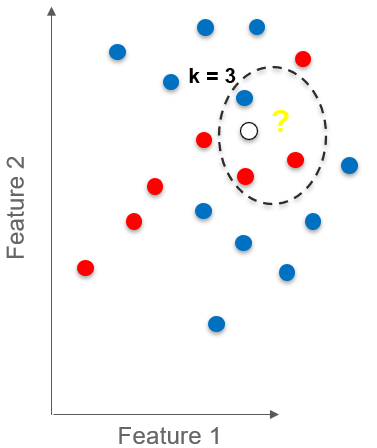
Si tomamos K=1, solo miraremos al vecino más cercano.

Aclaración: K es el nro de vecinos. La predicción será: Azul.



Si elegimos otro valor de k, por ejemplo k > 1, nuestra clasificación cambiará significativamente.

Por ejemplo, con k = 3 tenemos dos vecinos Rojos y uno Azul. Por lo tanto en base a este escenario, la clasificación será: Rojo.



*Link de Interés*:

* <https://themachinelearners.com/algoritmo-knn/>
* <https://www.youtube.com/watch?v=s-9Qqpv2hTYhttps://www.cienciadedatos.net/documentos/py07_arboles_decision_python.html>
* https://www.aprendemachinelearning.com/clasificar-con-k-nearest-neighbor-ejemplo-en-python/

**Regresión Logística**

La Regresión Logística o también conocida como Logistic Regression, es una técnica de aprendizaje automático que proviene del campo de la estadística. A pesar de su nombre no es un algoritmo para aplicar en problemas de regresión, en los que se busca un valor continuo, sino que es un método para problemas de clasificación, en los que se obtienen un valor binario entre 0 y 1.

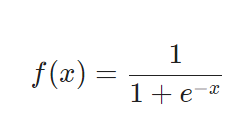
Por ejemplo, un problema de clasificación es identificar si una operación dada es fraudulenta o no, asociándole una etiqueta “fraude” a unos registros y “no fraude” a otros. Entonces, la Regresión Logística describe y estima la relación entre una variable binaria dependiente y las variables independientes.

En general, este algoritmo se puede utilizar para varios problemas de clasificación, como la detección de spam, predicción de la diabetes, si un cliente determinado comprará un producto en particular o si se irá con la competencia, hay muchos más ejemplos en donde se puede aplicar este algoritmo.

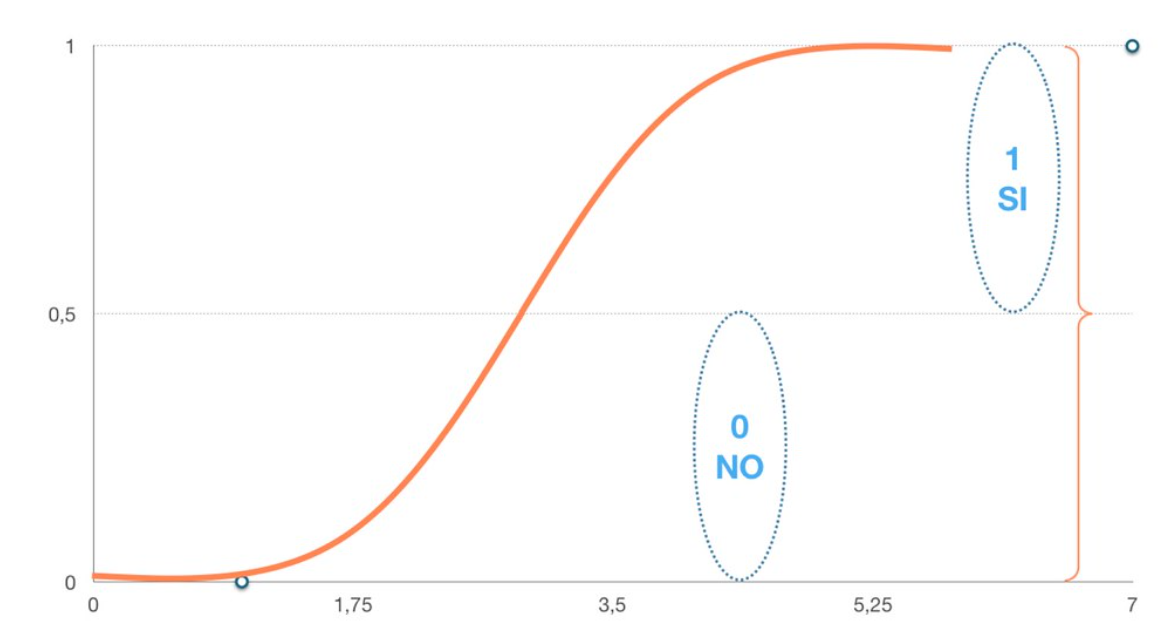


Por su parte, la Regresión Logística lleva el nombre de la función utilizada en el núcleo del método, la *Función Logística* es también llamada función *Sigmoide*. Esta función es una curva en forma de S que puede tomar cualquier número de valor real y asignar a un valor entre 0 y 1.

La ecuación que define la función sigmoidea es la siguiente:



Si la curva va a infinito positivo la predicción se convertirá en 1, y si la curva pasa el infinito negativo, la predicción se convertirá en 0. Si la salida de la función Sigmoide es mayor que 0.5, podemos clasificar el resultado como 1 o SI, y si es menor que 0.5 podemos clasificarlo como 0 o NO. Por su parte si el resultado es 0.75, podemos decir en términos de probabilidad como, hay un 75% de probabilidades de que el paciente sufra cáncer.



*Link de Interés:*

* [*https://www.analyticslane.com/2018/07/23/la-regresion-logistica/#:~:text=El%20algoritmo%20de%20regresi%C3%B3n%20log%C3%ADstica,obtiene%20un%20resultado%20u%20otro*](https://www.analyticslane.com/2018/07/23/la-regresion-logistica/)*.*
* [*https://www.aprendemachinelearning.com/regresion-logistica-con-python-paso-a-paso/*](https://www.aprendemachinelearning.com/regresion-logistica-con-python-paso-a-paso/)
* [*https://aprendeia.com/aprendizaje-supervisado-logistic-regression/*](https://aprendeia.com/aprendizaje-supervisado-logistic-regression/)
* [*https://www.iartificial.net/como-usar-regresion-logistica-en-python/*](https://www.iartificial.net/como-usar-regresion-logistica-en-python/)

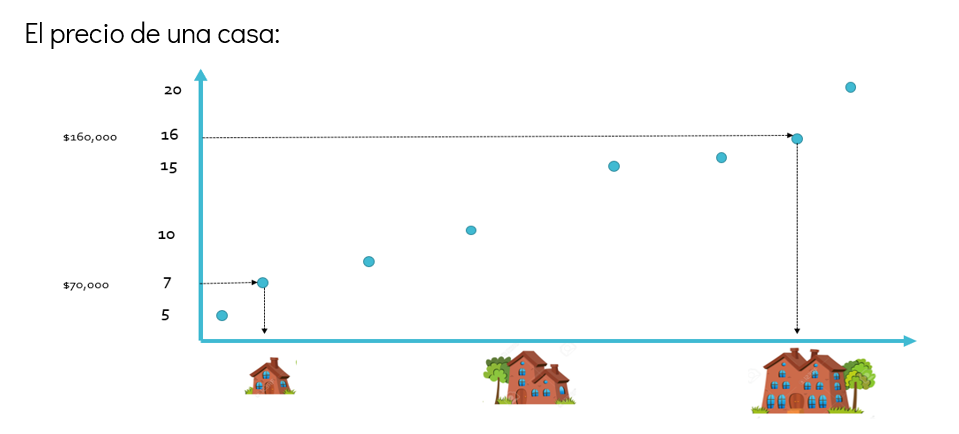
¿Los Algoritmos de Clasificación como Árboles de Decisión, KNN y la Regresión Logística son los únicos que existen? Por supuesto que no! Existen muchos más como ser por ejemplo: Support Vector Machines (SVM), Ramdom Forest, entre otros. 🤓

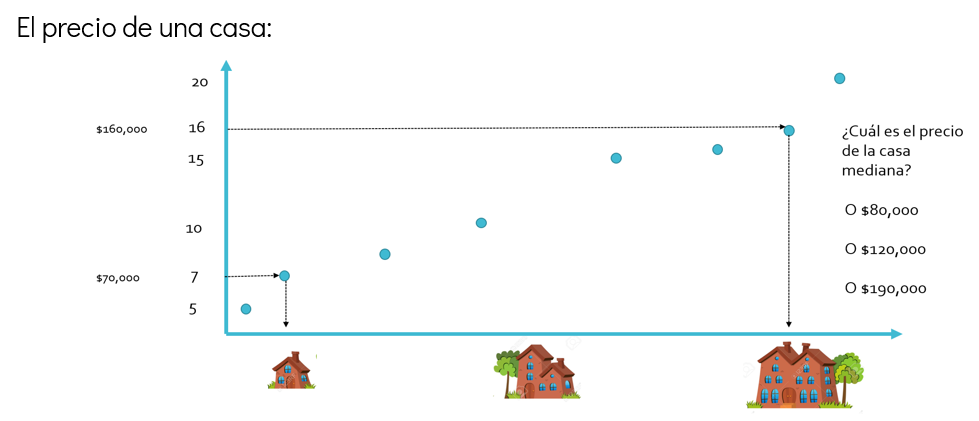
**Aprendizaje Supervisado - Regresión**

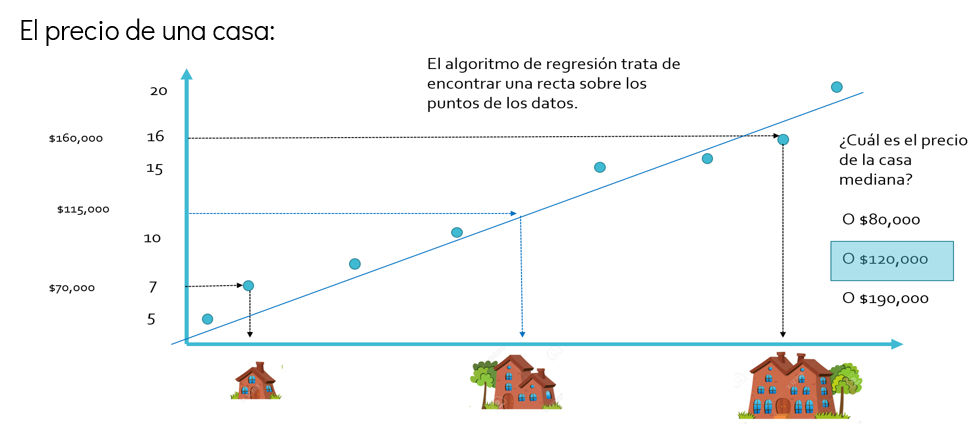
Los algoritmos de Regresión, intentan predecir una variable de tipo numérica o cuantitativa como por ejemplo: el precio de una casa.











*Link de Interés:*

* *https://pythondiario.com/2018/01/introduccion-al-machine-learning-8.html.*
* *https://www.datasource.ai/es/data-science-articles/una-guia-para-principiantes-sobre-la-regresion-lineal-en-python-con-scikit-learn*
* [*https://machinelearningparatodos.com/regresion-lineal-en-python//*](https://machinelearningparatodos.com/regresion-lineal-en-python/)

***Aclaración:*** Es importante recordar, que muchos de los algoritmos que se utilizan para **clasificación** también pueden utilizarse para problemas de **regresión**, como ser por ejemplo: Árboles o KNN. Cabe aclarar que por supuesto utilizan diferentes librerías. 😉