Machine Learning

Aprendizaje Supervisado

Rama de la inteligencia artificial la cual se basa en modelos matemáticos que, a partir de un conjunto grande de datos, aprenden.

Existen 3 tipos de aprendizaje

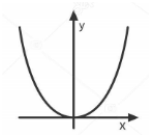
1.- Busca generar modelos con el fin de predecir o clasificar un nuevo conjunto de datos entrante

2.- Genera insight de los datos, ya sea segmentándolos en clusters o reduciendo sus dimensiones a los datos más representativos

3.- El objetivo es el mismo que en el supervisado pero se puede ir ajustando y optimizando con intervención humana por medio de “resultados y castigo”. Un aprendizaje de prueba y error

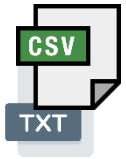
Herramientas y conocimientos básicos

Estadística



Álgebra Lineal

Creatividad



Estructuras y tipos de datos

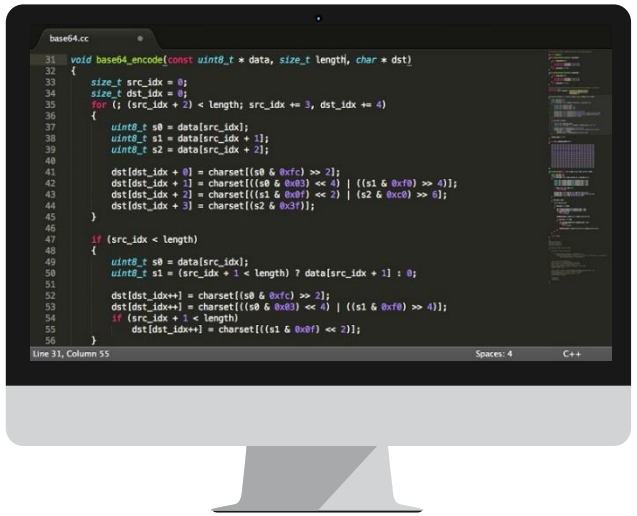
Motivación!

Anaconda Navigator

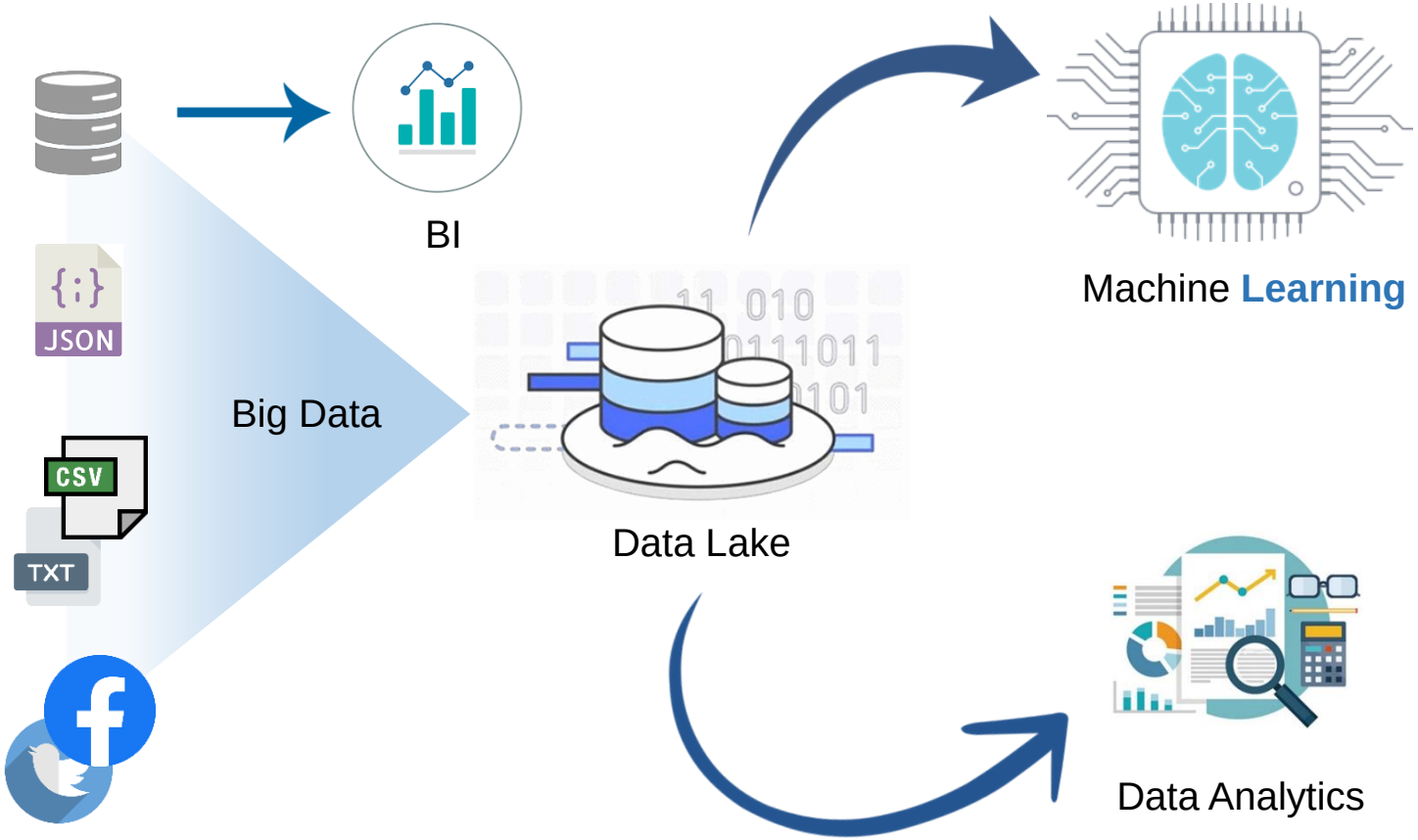
Programación Básica
Python o R

Stack Overflow

(spoiler: los ejemplos van a ser en python)



Ecosistema de los Datos



Roles

Data Engineer, Data Analytics, Data Scientist, Machine Learning Engineer, BI, DBA, Big Data Consultant

Estos son algunos de los roles que trabajan en este ecosistema, sus alcances varían bastante dependiendo la empresa donde vayan a ejercer

(No es extraño encontrar empresas donde busquen que un rol cubra todos los demás)

Preprocesamiento de Datos

Los modelos de Machine Learning esperan datos numéricos como datos de entrada, por ese motivo hay que realizar una limpieza y transformación de los mismos. Es un proceso tedioso pero determinante para la efectividad de su funcionamiento.

Limpieza y Manipulación

En esta etapa se realiza la eliminación de datos NaN (en caso de que fuera necesario), duplicados, filtrado, etc...

Visualización

Se realiza para analizar las distintas distribuciones estadísticas de nuestros datos, analizar la consistencia, identificar outliers, etc…

Feature Engineering

Es necesaria esta etapa para transformar todos nuestros datos a numéricos para después alimentar al modelo

Análisis de datos de AirBnB

Preprocesamiento

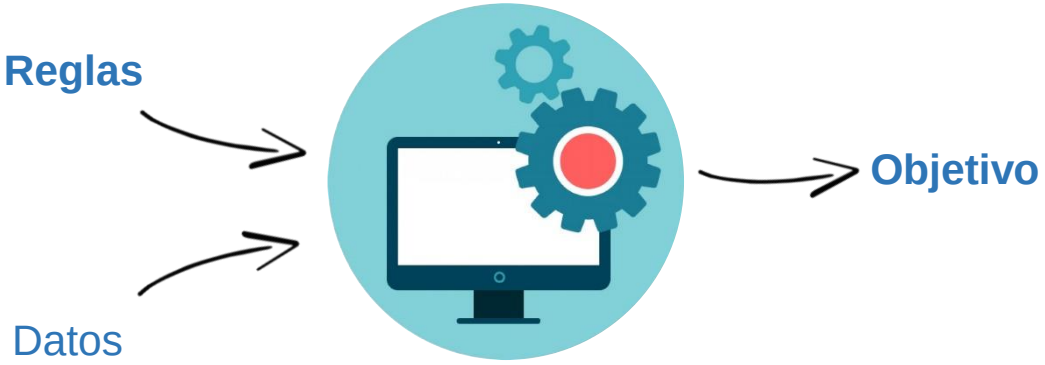
Vamos a presentar algunos ejemplos de esta etapa limpiando y visualizando datos de Airbnb, en este dataset tenemos los precios de los alquileres en la zona de New York.

Vamos a estar utilizando la plataforma de Kaggle, donde tenemos a disposición un Notebook online para poder ejecutar nuestro código y poder compartirlo. También vamos a encontrar varios Datasets para poder practicar y seguir aprendiendo. Es recomendable crear una cuenta y compartir el conocimiento con la comunidad.

Link a Kaggle

Link a GitHub

Programación Clásica



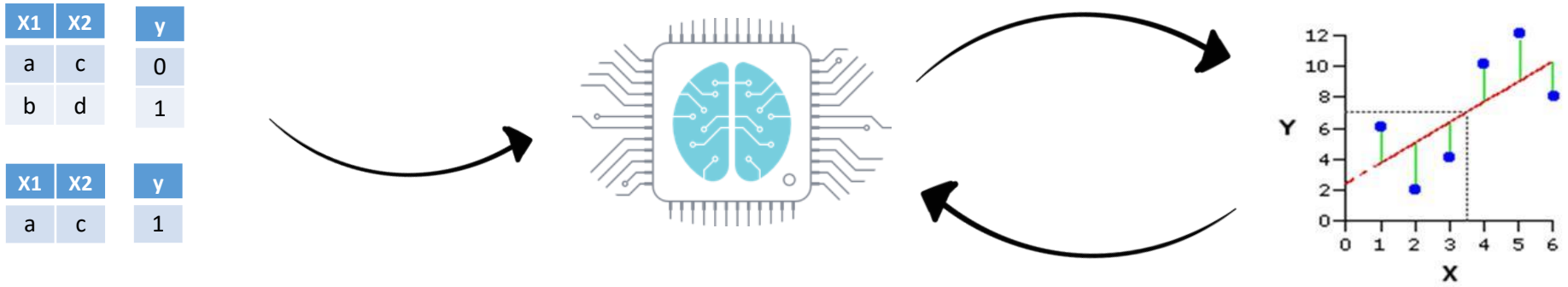
Machine Learning

La gran diferencia entre la programación clásica y Machine Learning es que los algoritmos los va a brindar nuestro modelo con solo pasarle nuestro conjunto de datos (*X)* y el objetivo (*Y)*. Nosotros generamos el modelo y vamos a poder delimitar su alcance a través de sus **hiperparametros.**

Esta definición aplica para el aprendizaje **Supervisado**, ya que en el aprendizaje **No Supervisado** no tenemos datos objetivo (*Y*), ya veremos ejemplos en la parte 2.

Machine Learning

Durante el proceso del desarrollo de nuestros modelos, primero tenemos que dividir el dataset en Entrenamiento y Testeo, seleccionamos nuestro modelo, lo entrenamos y evaluamos sus resultados con el conjunto de test. Existen distintas métricas para medir el error nuestros modelos (R^2 ,RMSE, matriz de confusión, etc..), eso nos va a brindar un **score** del modelo. Dependiendo el **score**, el modelo puede tener un ***Underfitting** u ***Overfitting**.



Train y Test

Realizamos una división de los datos en X_train, y_train, X_test, y_test
Generalmente esta división es de un **80%** train y **20%** para test.

Modelo

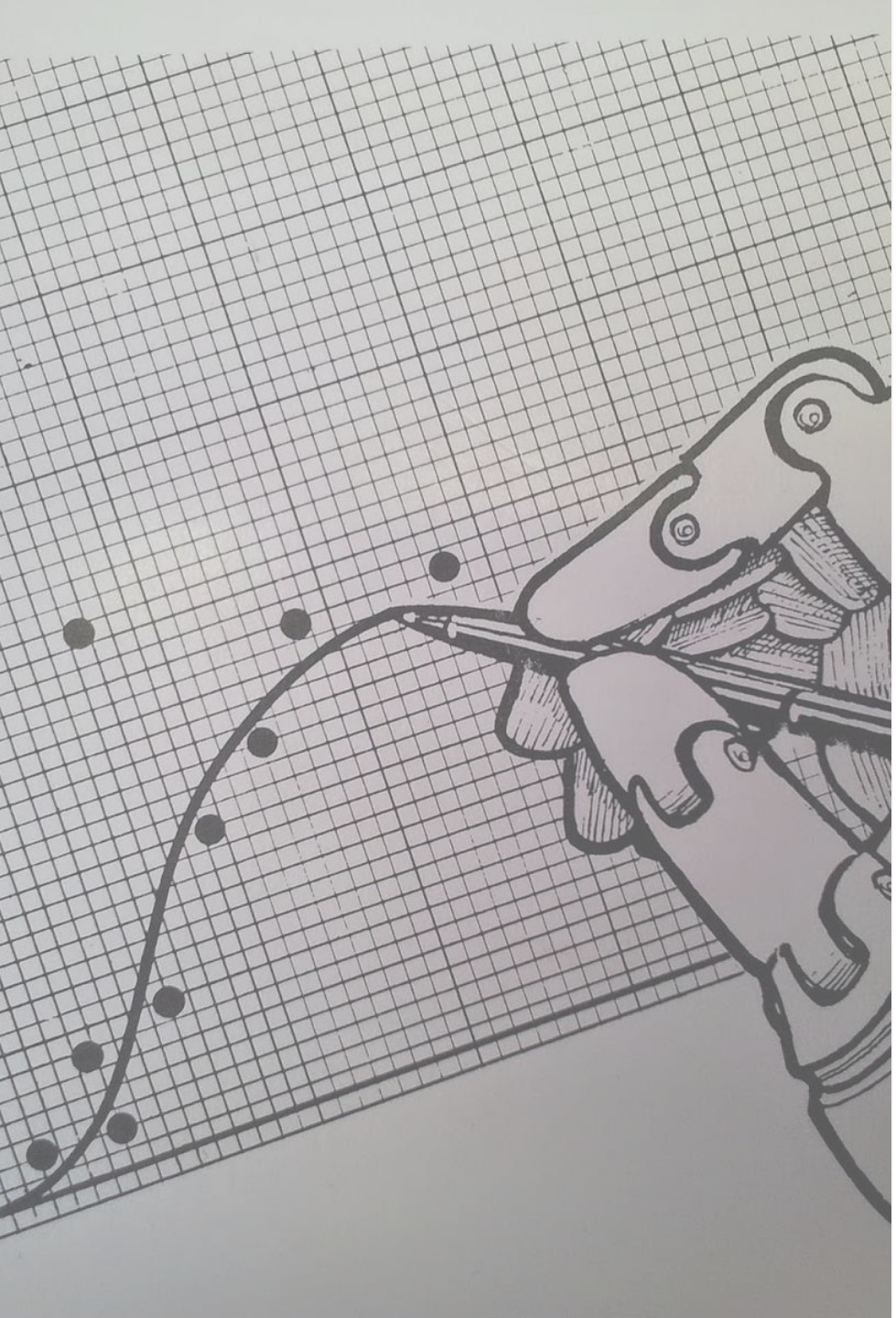
En esta etapa vamos a elegir el modelo y los hiperparámetros que vamos a utilizar (Linear Regression, SVM, Random Forest, etc..) y lo entrenamos.

Evaluación y optimización

En esta etapa medimos el error del modelo utilizando alguna métrica como R^2, MSE, RMSE, etc…
Para decir que nuestro modelo es bueno, su **score** tiene que estar entre el 75% y 98%
Si el modelo da un porcentaje muy alto, puede estar **overfitted***

***Overfitting**: Sobreajuste del modelo, no puede predecir correctamente un conjunto de datos distintos de los que se utilizaron para entrenarlo.

***Underfitting**: El modelo no llega a realizar una predicción correcta, esto se va a notar por el bajo score. Tan efectivo como lanzar una moneda.



Aprendizaje Supervisado

Regresión

Vamos a ver algunos algoritmos de regresión para poder **predecir valores continuos** como precios de un inmueble, cuánto tiempo tarda un vehículo en llegar a destino o estimar las ventas del próximo año.

La cantidad de soluciones que pueden desarrollarse depende de la creatividad de cada uno. Para eso es necesario tener conocimiento del área donde se vaya a aplicar el modelo (medicina, educación, biología, etc...).

Regresión Lineal

Se centra en el estudio de las relaciones entre una **variable dependiente** de una o más **variables independientes**

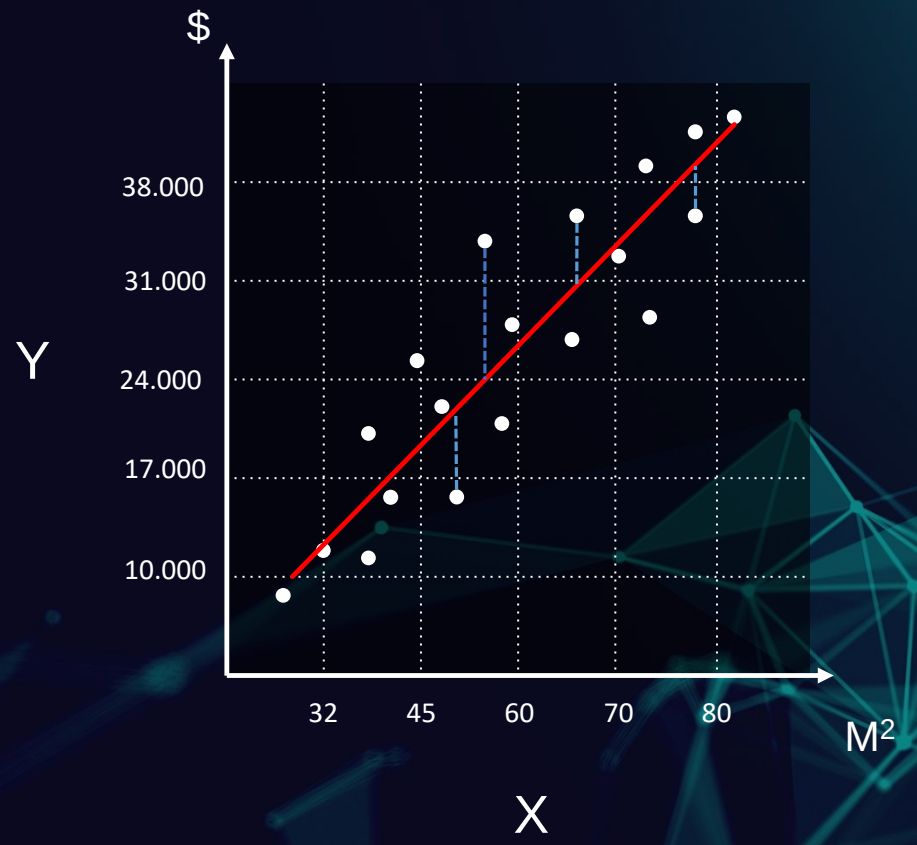
Busca la recta que, sin pasar por todos los puntos, represente los valores mas cercanos a la realidad

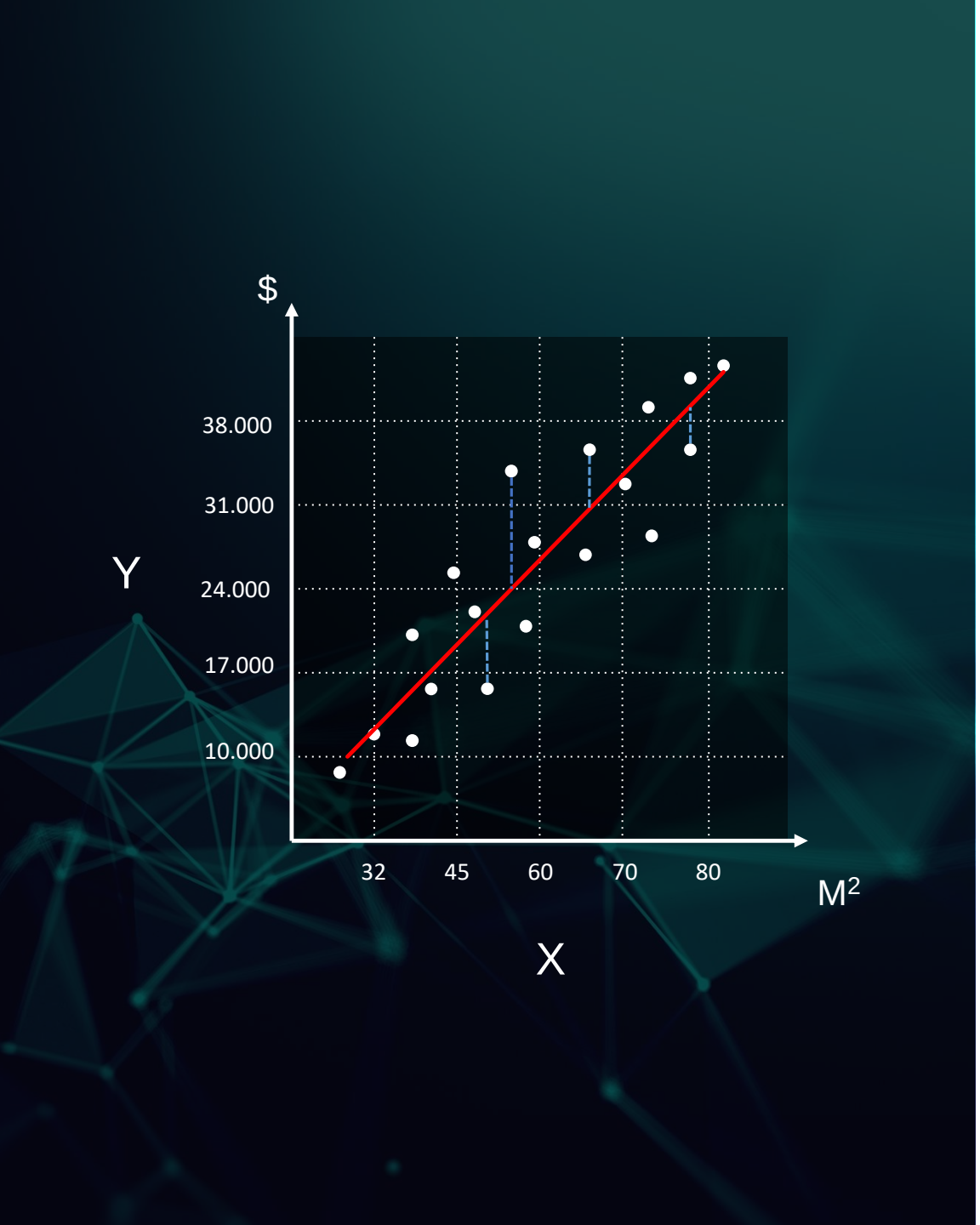
En este ejemplo vemos la relación lineal entre el precio de alquiler de un departamento y la cantidad de metros cuadrados

Término independiente

Variable Dependiente ← $Y\ (\$) = b_0 + b_1 * X_1\ (M^2)$ → Variable Independiente

Pendiente





Minimizando el Error

Para calcular el error tenemos que tomar la diferencia entre el valor predicho (y_i) y el valor actual ($\hat{y}_i$), en esto se basa la **función de coste:** Error Cuadrático Medio (Mean Square Error)

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

Una vez hecho esto tenemos que buscar el error mínimo de nuestra función de coste, para ello utilizamos una derivada del MSE, el **Error Cuadrático Medio Mínimo** (Root Mean Square Error)

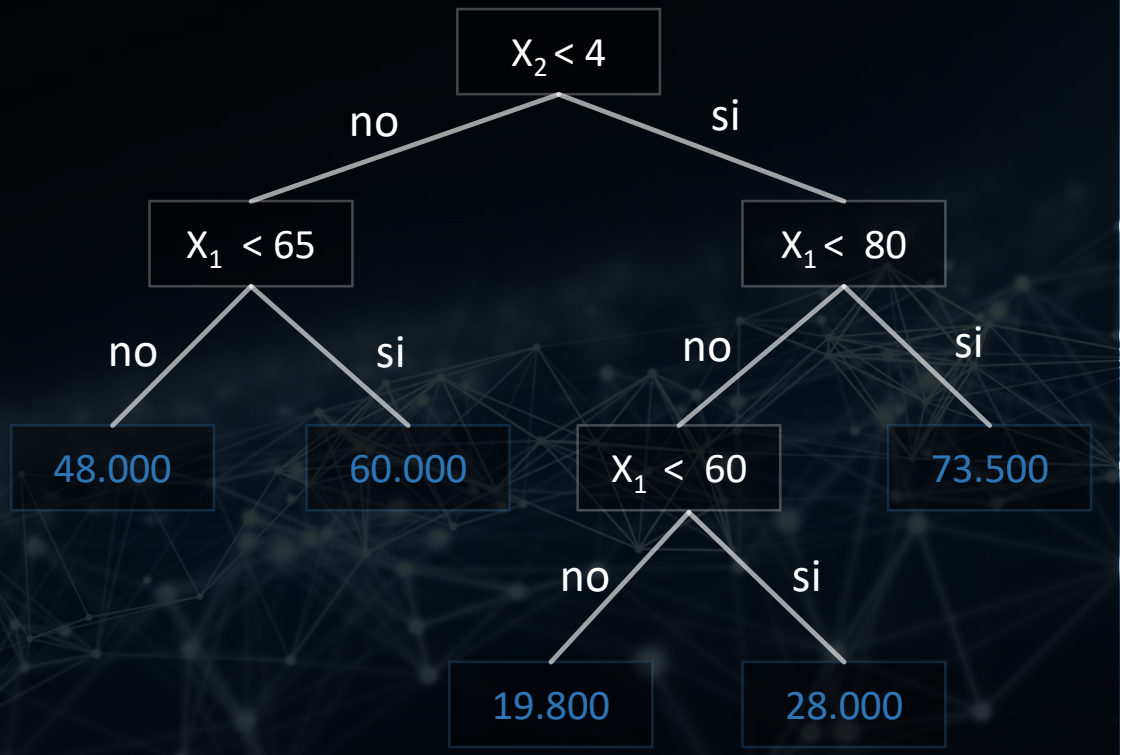
$$\mathbf{RMSE} = \sqrt{\sum_{i=1}^{n}\frac{(y_i - \hat{y}_i)^2}{n}}$$

Regresión Lineal Múltiple

Para poder predecir el precio de una casa no nos alcanza sólo con los metros cuadrados, tenemos que tener en cuenta muchas otras variables para ser más efectivos.
Cada variable representa una **dimensión**, por ese motivo no es posible representarla gráficamente más allá de una tercera dimensión.

De todas formas esto no es un problema, ya que la ecuación sigue siendo la misma, sólo que con más variables independientes.

$$Y (\$) = b_0 + b_1 * X_1 + b_2 * X_2 \ldots + b_n * X_n$$



Regresión con Árboles de Decisión

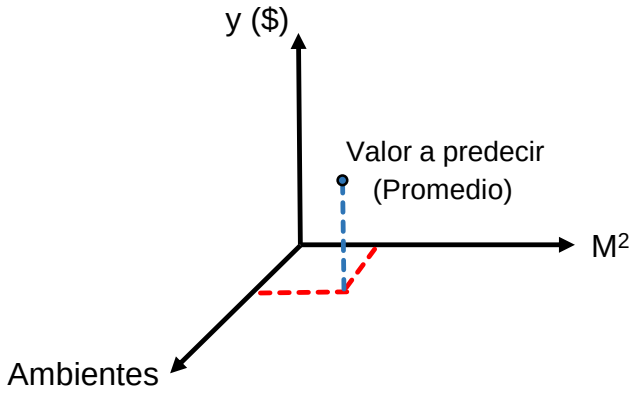
Este algoritmo va a ir evaluando distintas condiciones aplicadas a los datos para poder predecir cúal será el resultado de un conjunto de datos entrante.

En este caso tenemos 5 **hojas** en nuestro árbol, estos nodos son los rangos donde van a caer nuestros nuevos datos a predecir.

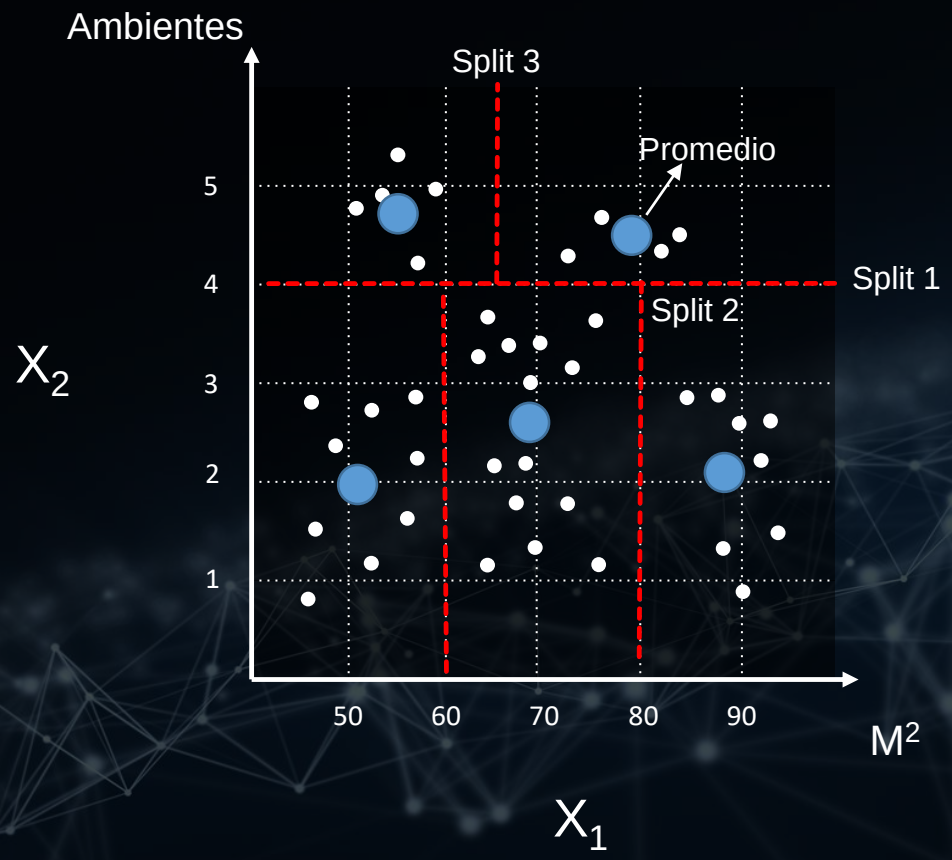
Esta cantidad va a depender de la **profundidad** del árbol que indiquemos al momento de entrenarlo. Tenemos que tener cuidado al momento de seleccionar este valor, ya que si es muy grande se puede producir el **overfitting** del modelo.

Regresión con Árboles de Decisión

Dependiendo como configuremos los hiperparámetros de nuestro árbol (profundidad, criterio del error, etc...) se van a ir realizando distintas divisiones en nuestro conjunto de datos que tengan características en común, cada una de estas divisiones son las llamadas **hojas del árbol**



En este ejemplo nuestro valor a predecir se encuentra en una tercera dimensión. Para poder predecirlo se toma el promedio de todos los puntos vecinos que se encuentran en el nodo hoja



Bosques Aleatorios

Es la técnica de entrenar varios árboles de decisión en conjunto, este tipo de modelo suele ser muy efectivo y potente. También suelen consumir muchos recursos de cómputo.

Se toma un subconjunto K de datos de entrenamiento para entrenar cada árbol con un conjunto distinto.

Se elige la cantidad de árboles (Ntree) que vamos a construir, se entrenan, cada uno realiza la predicción para su conjunto de datos y se le aplica el promedio de los resultados.

Existen otros algoritmos de **aprendizaje conjunto**, como gradient boosting, xgboost, voting classifier.
Es recomendable investigar sobre éstos ya que son técnicas muy efectivas.

Evaluando la Regresión

Para poder evaluar el score de nuestros algoritmos de regresión vamos a utilizar el factor R^2.
Anteriormente vimos cómo medir error de nuestra función lineal a través del MSE. Vamos a retomar esta función de coste que es la diferencia entre los valores reales y los predichos

$$SS_{res} = \sum_i (yi - \hat{y}i)^2$$

Suma de los cuadrados de los residuos

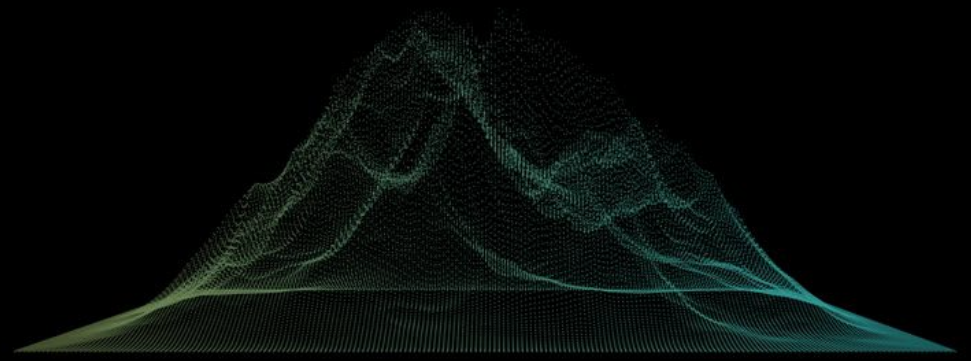
Vamos a calcular la diferencia de los valores reales y la media

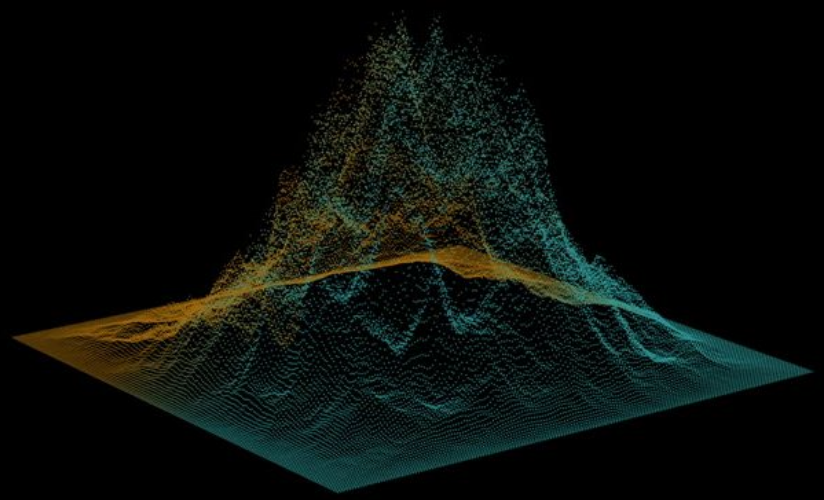
$$SS_{tot} = \sum_i (yi - \bar{y}i)^2$$

Suma de los cuadrados de los totales

A través de estos 2 términos podemos definir el factor R^2 con el cual podemos medir el score de nuestro modelo. Mientras este más cerca de 1, mejor será nuestro modelo.

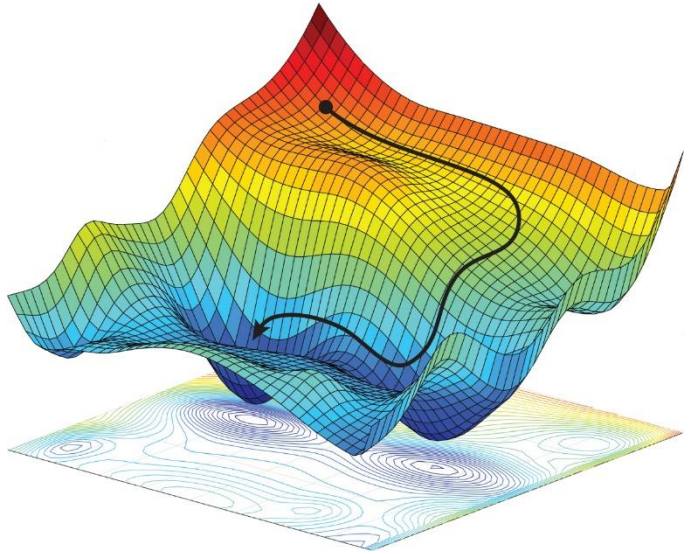
$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$





Optimización

Uno de los algoritmos de optimización más potentes es el Descenso del gradiente, el cual a través de una función de coste busca el error mínimo. El algoritmo comienza en un punto aleatorio y va derivando la función de forma iterativa hasta encontrar un punto donde la pendiente sea próxima a nula. Este punto puede ser un mínimo local (de una porción de la función de coste) o global (de la función de coste total).



Otro de los algoritmos de optimización es GridSearch. Este nos permite setear un conjunto de hiperparametros de nuestro algoritmo de ML y va a ir comparando diferentes configuraciones y brindando un puntaje para obtener la mejor configuración para nuestro modelo.

Radiación solar

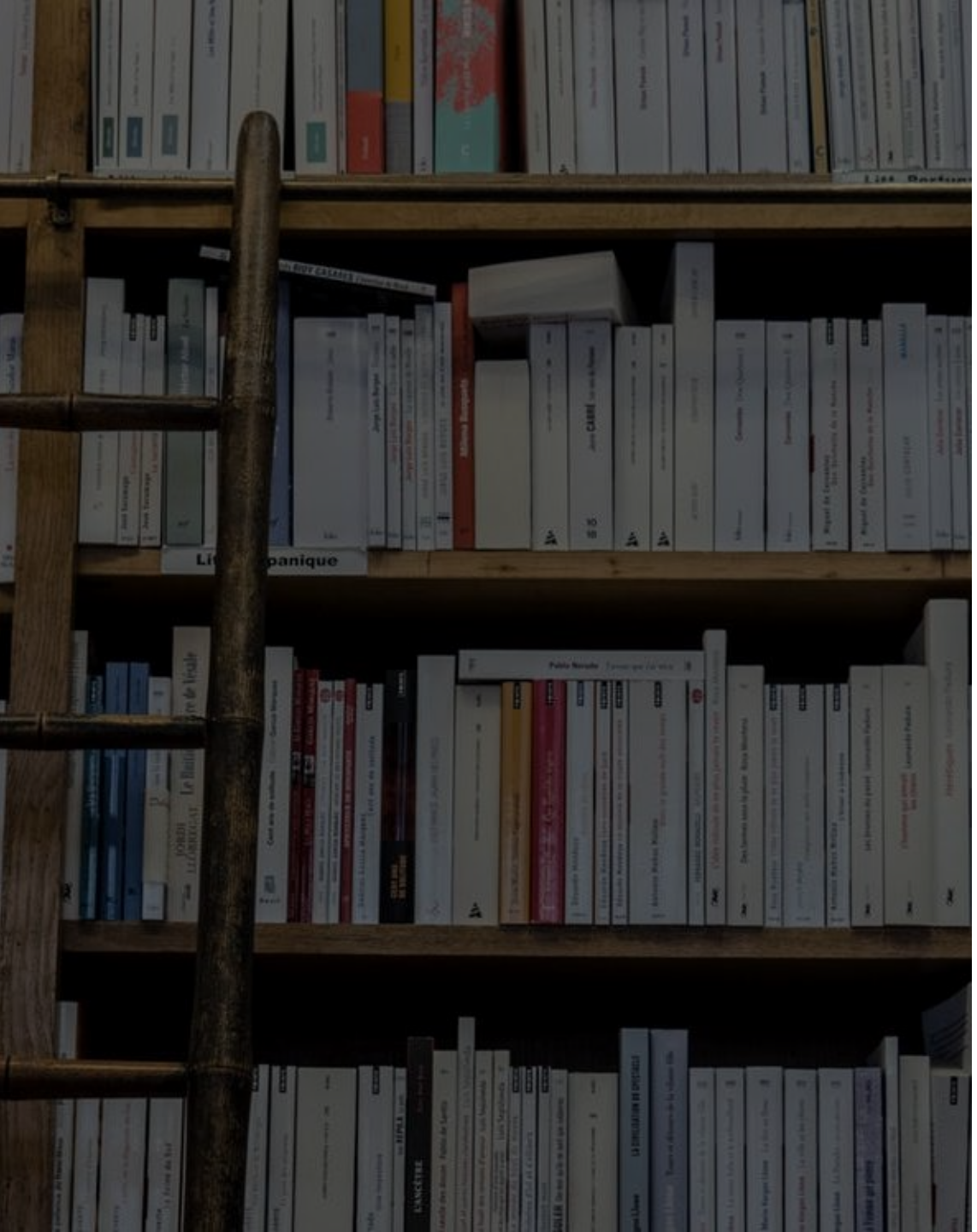
Regresión

Vamos a intentar armar un modelo que prediga la cantidad de radiación solar en watts por m^2 en base a la temperatura, humedad, horario del día y distintos factores que tengan una relación con la radiación.
Esto nos podría servir para calcular cuánta energía recibirá un panel solar

Aplicaremos distintas técnicas de regresión vistas anteriormente y vamos a medir qué tan bueno es nuestro modelo y aplicar técnicas de optimización

Link a Kaggle

Link a GitHub



Aprendizaje Supervisado
Clasificación

Los modelos de clasificación buscan predecir a qué clase pertenece un conjunto de datos entrante. Esto nos permite clasificar textos a través NLP, tipos de cliente, productos, enfermedades, películas, imágenes, etc…

Pueden usarse varios modelos, tanto para clasificación como para regresión, como por ejemplo los árboles de decisión, KNN, Bosques Aleatorios.

Regresión Logística

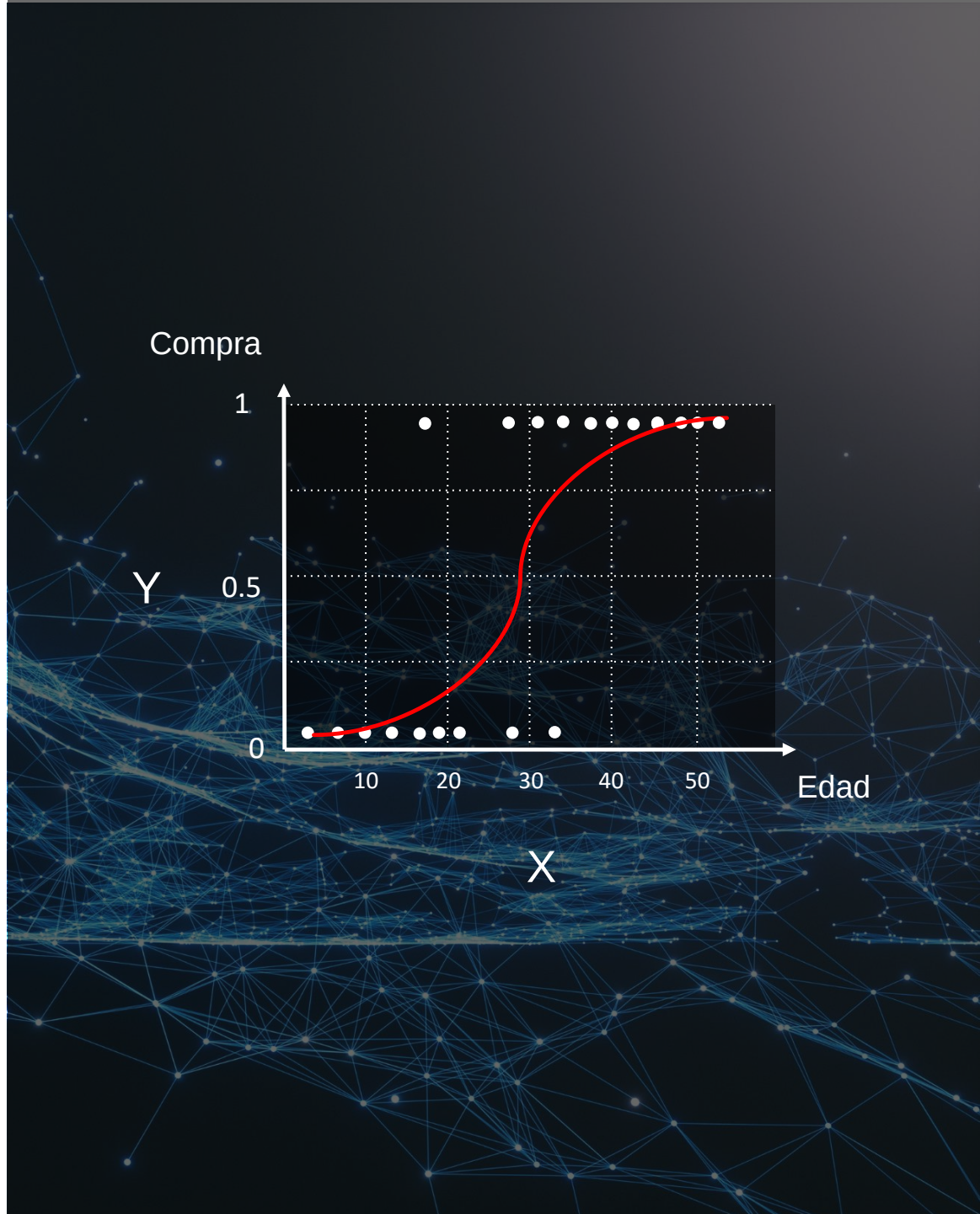
Este modelo tiene una relación con la regresión lineal ya que se le aplica la **función sigmoide** a la recta para transformarla y poder realizar las clasificaciones

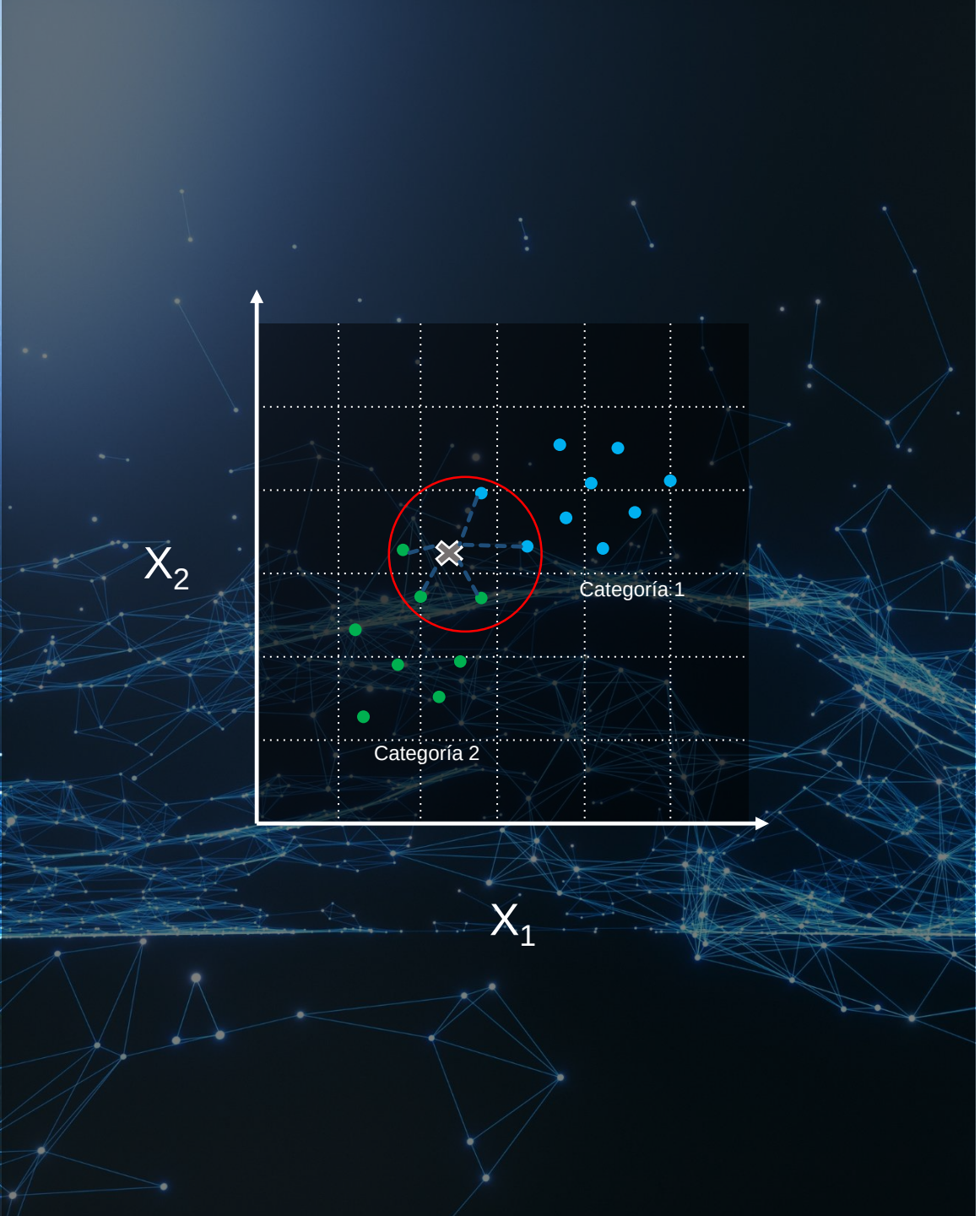
En este caso, nuestro objetivo *Y* pasa a ser un valor de probabilidad de 0 a 1. Por ejemplo nos sirve para determinar si un cliente en base a su edad realizó una compra o no.

$$\ln\left(\frac{p}{1+p}\right) = b_0 + b_1 * X_1$$

Función Sigmoide aplicada a Y

Regresión lineal





K-Nearest Neighbors
(KNN)

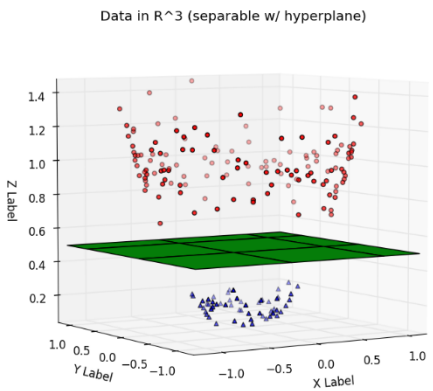
Este algoritmo se basa en buscar los ”vecinos más cercanos” a nuestro nuevo dato.
Determina un número k de vecinos, toma la distancia a esos puntos (suele ser la Euclidiana), entre esos K se cuenta cuántos pertenecen a cada categoría y se determina la categoría con más vecinos.

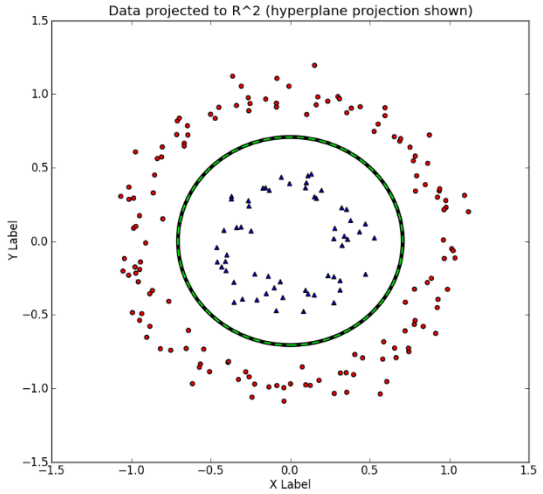
En este ejemplo se toma un número K = 5, se determina la distancia entre los puntos y vemos que tenemos 3 puntos verdes contra 2 puntos azules. Por votación nuestro nuevo dato cae dentro de la categoría verde.

Este es uno de los algoritmos más simples dentro de la clasificación.

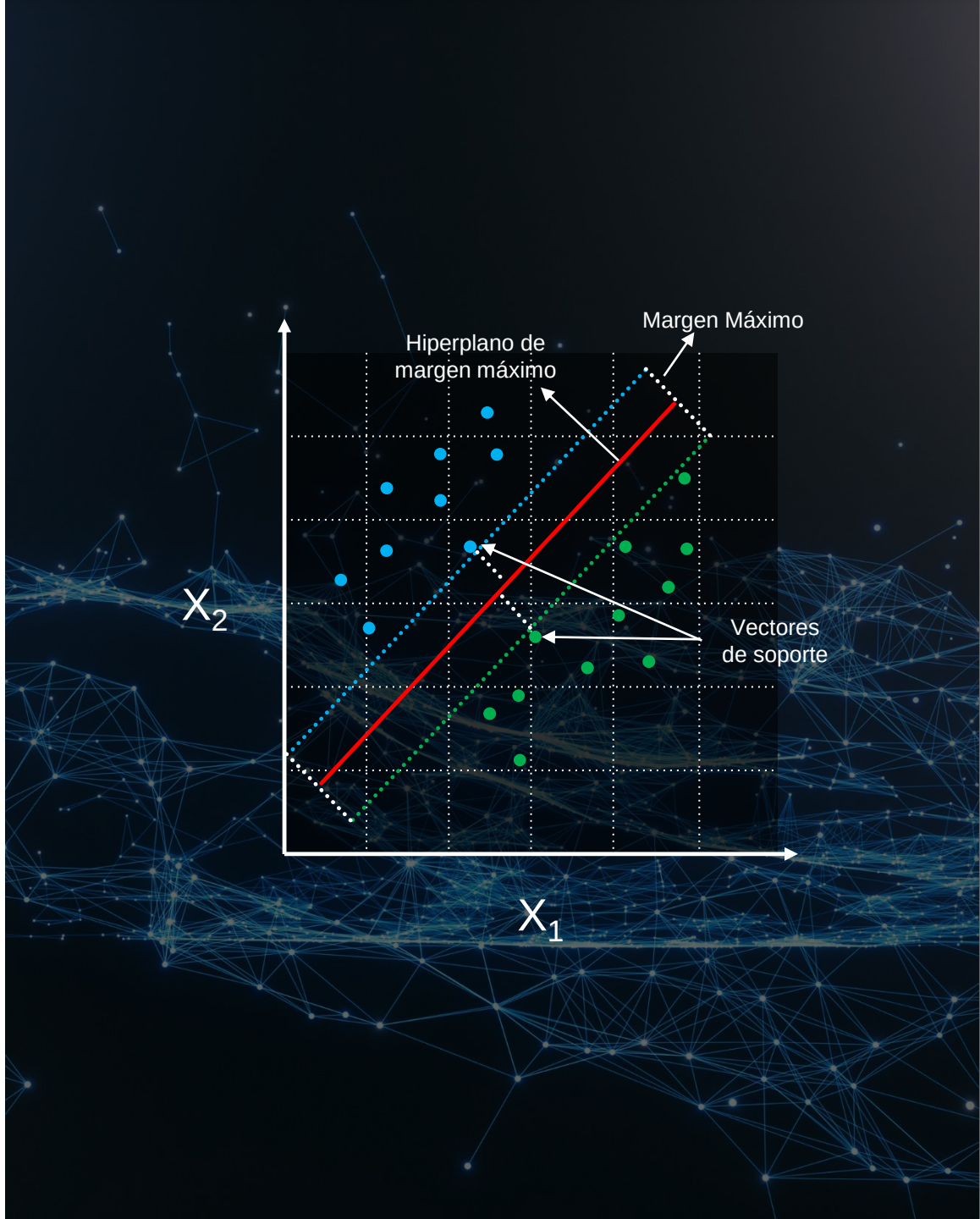
Support Vector Machine

Este modelo clasifica un conjunto de datos a través de una línea divisora.
Se basa en determinar un hiperplano óptimo que separe las clases buscando el mayor margen entre las instancias a través de los vectores de soporte. Estos vectores son los puntos más cercanos a la línea divisora.





Es un algoritmo muy potente, ya que en caso de tener los datos agrupados de forma no lineal, podemos pasar a otra dimensión para encontrar una separación.



Evaluando la Clasificación

Una de las formas de visualizar el desempeño de nuestro modelo es a través de una matriz de confusión, la cual está representada por los valores reales y los valores predichos

En nuestro ejemplo podemos ver que tenemos 950 gatos clasificados como gatos (Verdaderos Positivos / TP), 14 perros clasificados como gatos (Falsos Positivos / FP), 20 gatos clasificados como perros (Falsos Negativos / FN) y 840 perros clasificados como perros (Verdaderos Negativos / TN)

De esta forma podemos evaluar cómo se comportó nuestro algoritmo, en este caso vemos que tuvo un muy buen rendimiento a la hora de clasificar

Otras métricas de evaluación son Precisión y Recall, es recomendable investigarlas

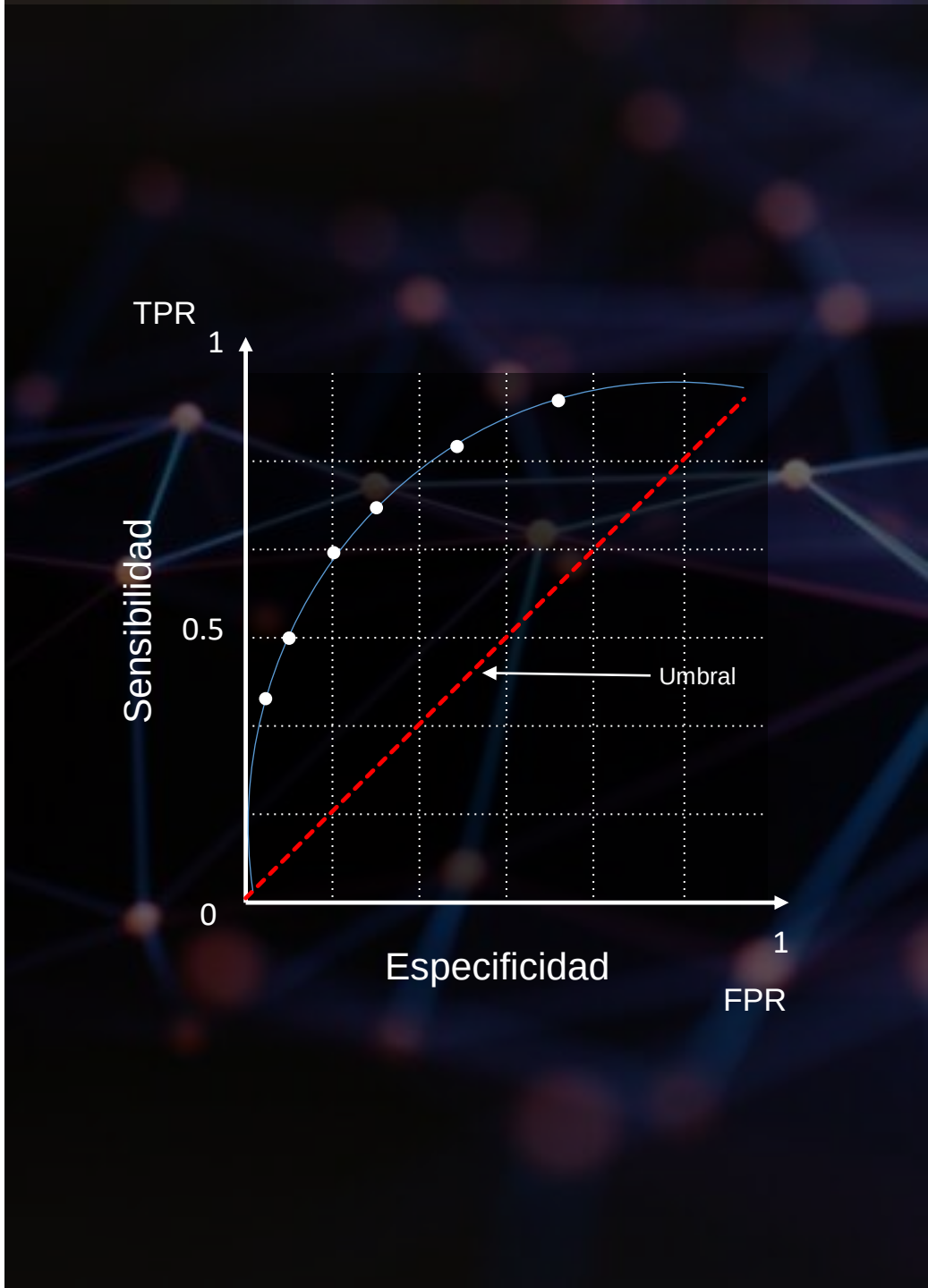
Área bajo la curva ROC

Una curva ROC es una herramienta gráfica que nos sirve para entender la eficacia de un modelo de clasificación. Está basada en la matriz de confusión.

Sensibilidad (TPR) = TP / (TP+FN)

Especificidad (FPR) = TP / (TP+FN)

Dependiendo del punto de la curva donde nos paremos, tendremos más TPR y menos FPR.
Mientras mayor sea el área bajo la curva, mejor será nuestro modelo.
Se tomará el punto óptimo que será aquel que se encuentra más arriba a la izquierda. De esta forma mediremos el score de nuestros modelos de clasificación.



Enfermedades Cardíacas

Clasificación

Vamos a armar un modelo que pueda clasificar pacientes con problemas cardíacos y aquellos sanos dependiendo de su edad, sexo, presión, dolores, etc...

Aplicaremos distintas técnicas de clasificación vistas anteriormente y algunas otras que utilizamos en regresión como los arboles de decisión y bosques aleatorios.
Vamos a medir qué tan bueno es nuestro modelo a través de la métrica de la curva ROC y aplicaremos Grid Search para ver si mejora el score del modelo.

Link a Kaggle

Link a GitHub

Este documento es un guía introductorio a Machine Learning con Python, orientado al aprendizaje supervisado.
Espero que les sea de utilidad a los que recién empiezan y a los que ya están con proyectos de Machine Learning y Ciencia de Datos. Dejo mis datos de contacto y las redes en las que participo .Para los que tengan dudas, voy a estar subiendo notebooks con distintos análisis de datos y modelos.
También estoy armando otras publicaciones sobre Aprendizaje no Supervisado, Aprendizaje por Refuerzo y **Deep Learning**.

Gracias!

Cristian **Benitez**

Data Scientist en Datastar

chbenitez@gmail.com

kaggle.com/chbenitez

github.com/chbenitez92