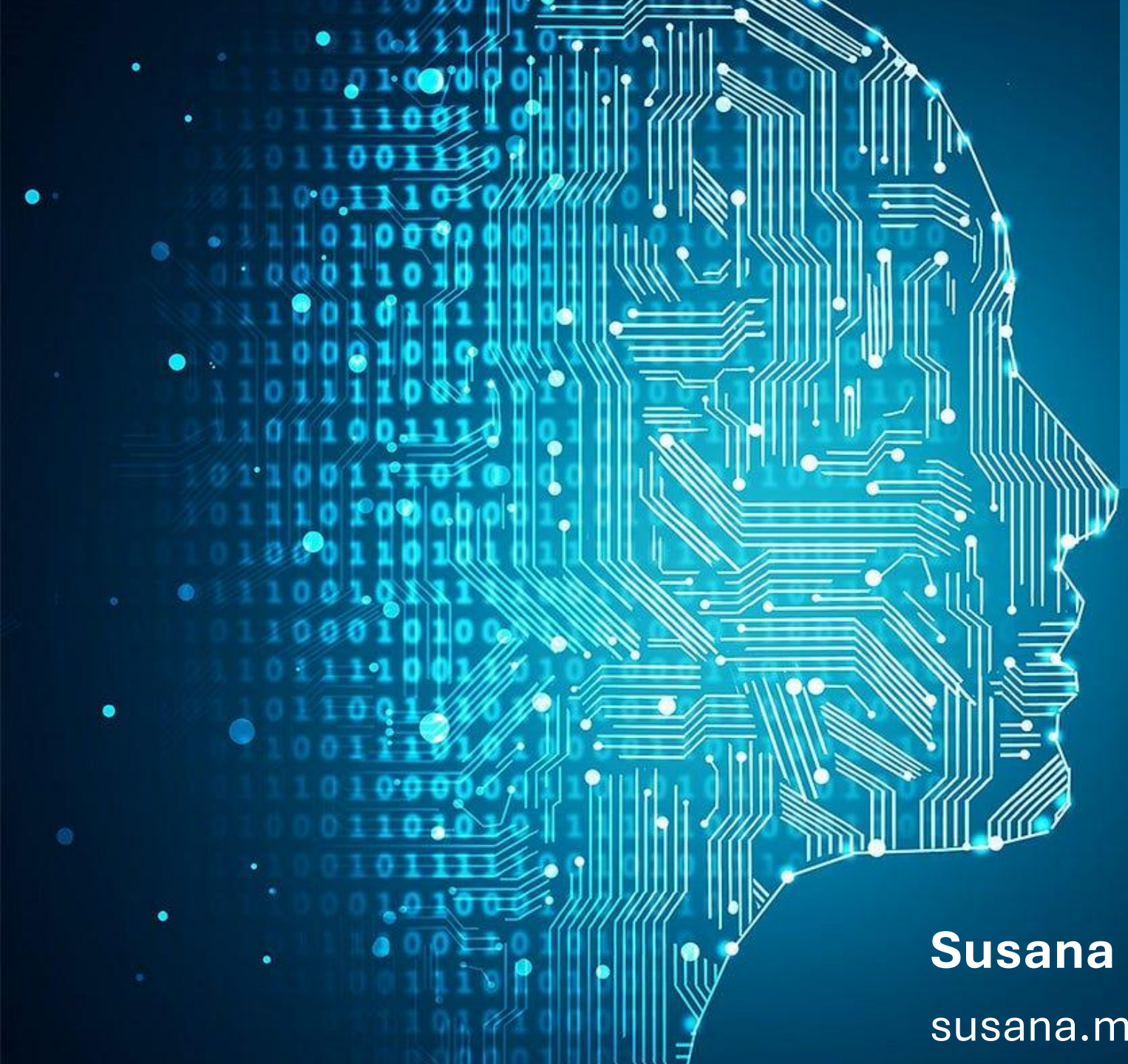
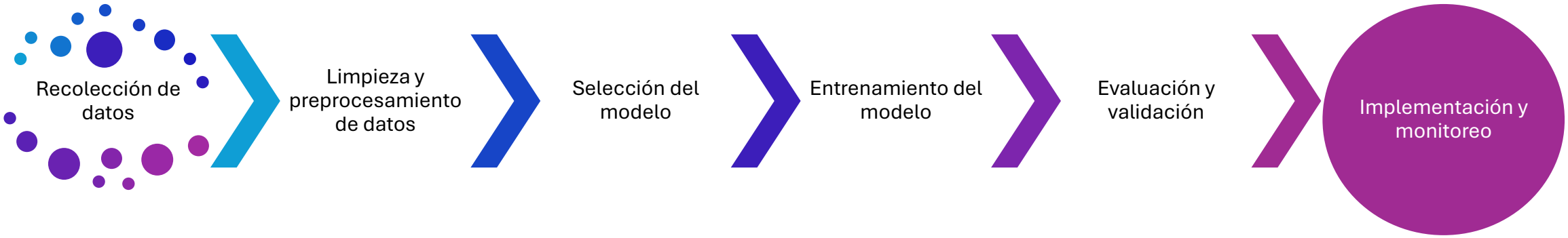


Machine Learning

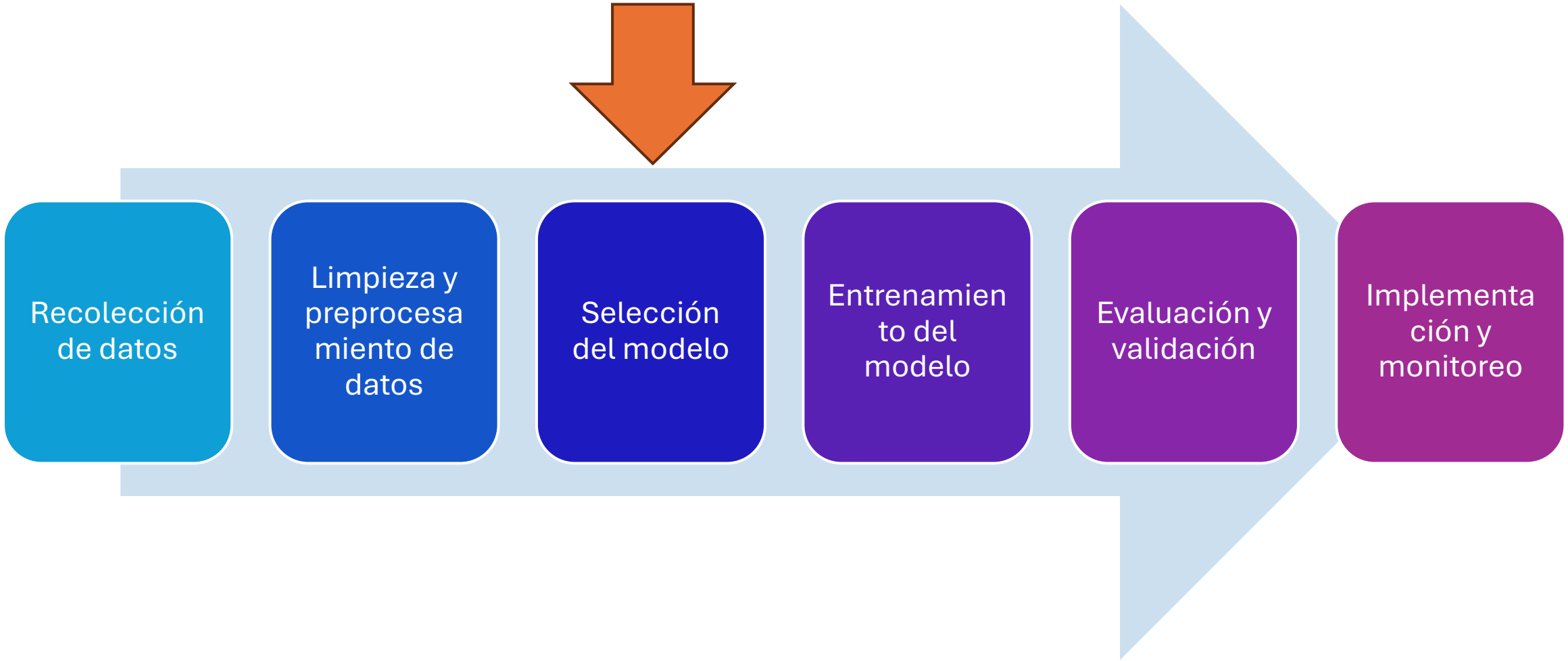
Susana Medina Gordillo

susana.medina@correounivalle.edu.co

Flujo de trabajo en Machine Learning



Flujo de trabajo en Machine Learning



Técnicas de Aprendizaje Supervisado: Clasificación

Introducción a la Clasificación

Aplicaciones comunes

Detección de spam

Diagnóstico médico

Reconocimiento de imágenes

Regresión Logística

Clasificación binaria y su importancia en el aprendizaje supervisado

Regresión Logística: definición

Clasificación binaria

- Predicción de probabilidad de pertenencia a una clase.
- Función sigmoide: transformación de valores a rango [0, 1].

Interpretación de coeficientes

- Relación entre variables predictoras y probabilidad de clase.

Evaluación

- Matriz de confusión
- AUC-ROC
- precisión
- recall
- F1-score

Regresión Logística: Clasificación binaria

La Regresión Logística se utiliza principalmente para problemas de **clasificación binaria**, donde la variable objetivo tiene dos categorías (por ejemplo, sí/no, 0/1).

También se puede extender a problemas de **clasificación multiclase** mediante técnicas como "uno contra todos" (one-vs-all) o "uno contra uno" (one-vs-one).

Regresión Logística: función sigmoide

Utiliza la **función sigmoide** (o función logística estándar) para transformar la salida **lineal** en una probabilidad entre 0 y 1. Esta función mapea cualquier valor real a un valor entre 0 y 1, lo que permite interpretar la salida como una probabilidad.

$$f(x) = \frac{1}{1 + e^{-x}}$$

Regresión Logística: función sigmoidea

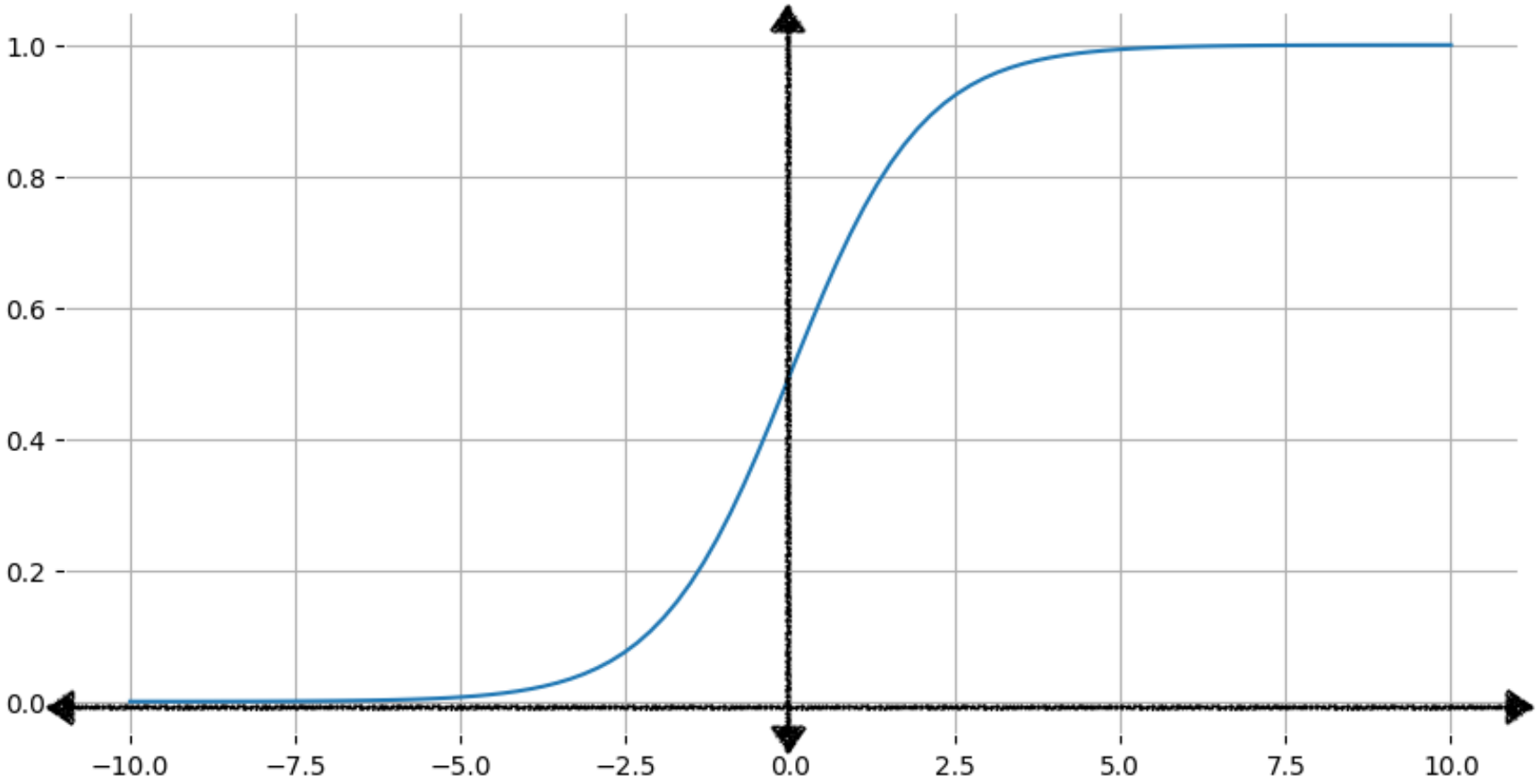
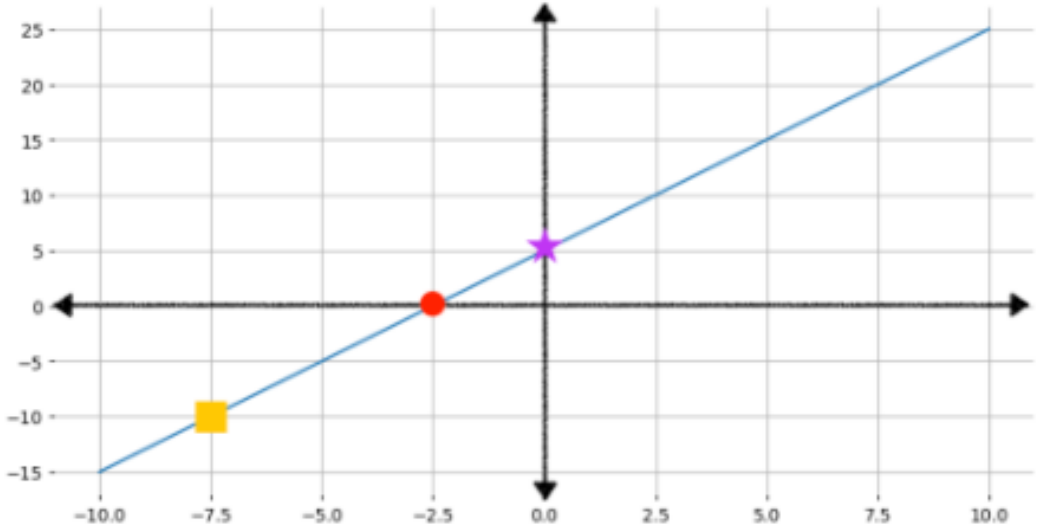


Gráfico de la función sigmoidea

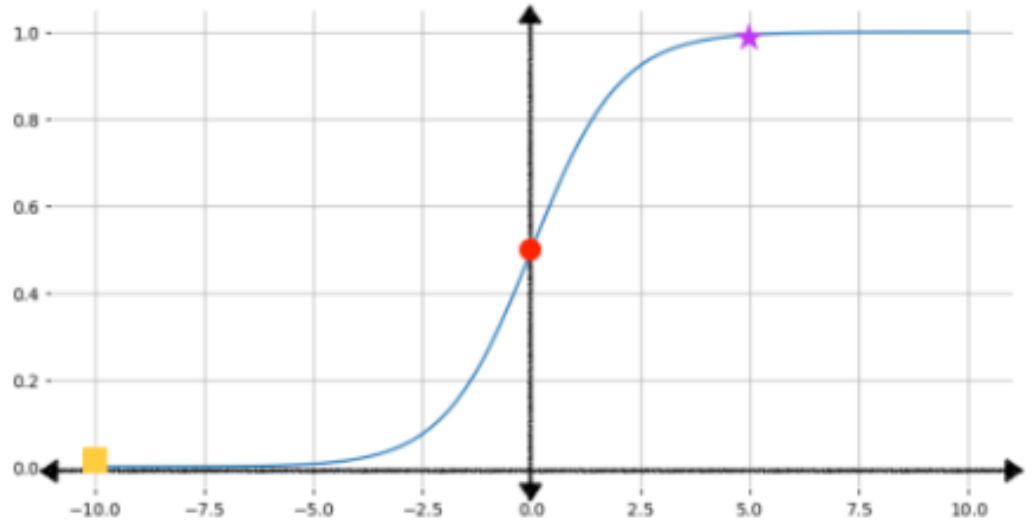
La curva se aproxima a 0 a medida que los valores de x disminuyen hasta el infinito negativo y se aproxima a 1 a medida que los valores de x aumentan hacia el infinito.

Transformación lineal a Regresión Logística

$z = 2x + 5$



$y' = 1 / (1 + e^{-z})$



Regresión Logística: pérdida y regularización

Regresión logística los modelos se entrenan con el mismo proceso regresión lineal con las siguientes dos distinciones clave:

1. Los modelos de regresión logística usan **Pérdida logística** como la función de pérdida en lugar de pérdida al cuadrado (L_2).
2. Aplicación de la regularización es fundamental para evitar sobreajuste.

La **tasa de cambio** de un modelo de regresión logística no es constante.

La curva sigmoidea tiene forma de S, en lugar de lineales. Cuando el valor de logaritmos de probabilidad (z) se acerca a 0, el valor es bajo, aumentos en z generan cambios mucho mayores en que cuando es un valor alto un número positivo o negativo.

Regresión Logística: pérdida logística

$$\text{Log Loss} = \sum_{(x,y)\in D} -y \log(y') - (1-y) \log(1-y')$$

$(x, y) \in D$ es el conjunto de datos que contiene muchos ejemplos etiquetados, que son (x, y) de pares.

y es la etiqueta en un ejemplo etiquetado. Como se trata de regresión logística, cada valor de y debe ser 0 o 1.

y' es la predicción de tu modelo (un valor entre 0 y 1), según el conjunto de atributos en x.

Regresión Logística: Interpretación de coeficientes

Los **coeficientes** del modelo representan el cambio en el logaritmo de las probabilidades (***log-odds***) de la variable objetivo por cada unidad de cambio en la variable predictora.

La interpretación de los coeficientes es diferente a la de la regresión lineal, ya que se relacionan con las probabilidades en lugar de los valores directos.

En resumen, la regresión logística es una herramienta poderosa para problemas de clasificación, especialmente cuando se trata de predecir la **probabilidad de pertenencia** a una categoría.

K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN): definición

Clasificación basada en proximidad

- Asignación de etiqueta según la mayoría de los vecinos cercanos.
- Elección del valor de k: impacto en el modelo.

Distancias

- Euclidiana
- Manhattan
- Minkowski

Ventajas y desventajas

- Simplicidad
- Sensibilidad a la escala
- Costo computacional

K-Nearest Neighbors: introducción

K-Nearest Neighbors (KNN) es un algoritmo de aprendizaje supervisado **simple** pero poderoso que se utiliza tanto para clasificación como para regresión.

Su principio fundamental radica en la idea de que **puntos de datos similares** tienden a pertenecer a la misma categoría

K-Nearest Neighbors: Aprendizaje basado en instancias

KNN es un algoritmo "*perezoso*", lo que significa que no construye un modelo explícito durante el entrenamiento.

KNN **memoriza** el conjunto de datos de entrenamiento y realiza predicciones directamente a partir de él.

Clasificación por proximidad:

- Para clasificar un nuevo punto de datos, KNN encuentra los "k" vecinos más cercanos en el conjunto de datos de entrenamiento, basándose en una **medida de distancia** (como la *distancia euclidiana*).
- La clase predicha para el nuevo punto de datos es la clase mayoritaria entre sus "k" vecinos más cercanos.

K-Nearest Neighbors: Elección de "k"

El valor de "k" es un hiperparámetro **crucial** que afecta el rendimiento del algoritmo.

- Un valor pequeño de "k" puede hacer que el modelo sea sensible al ruido y a los valores atípicos.
- Un valor grande de "k" puede suavizar demasiado las fronteras de decisión.

K-Nearest Neighbors: Elección de "k"

KNN utiliza medidas de distancia para determinar la **similitud** entre los puntos de datos.

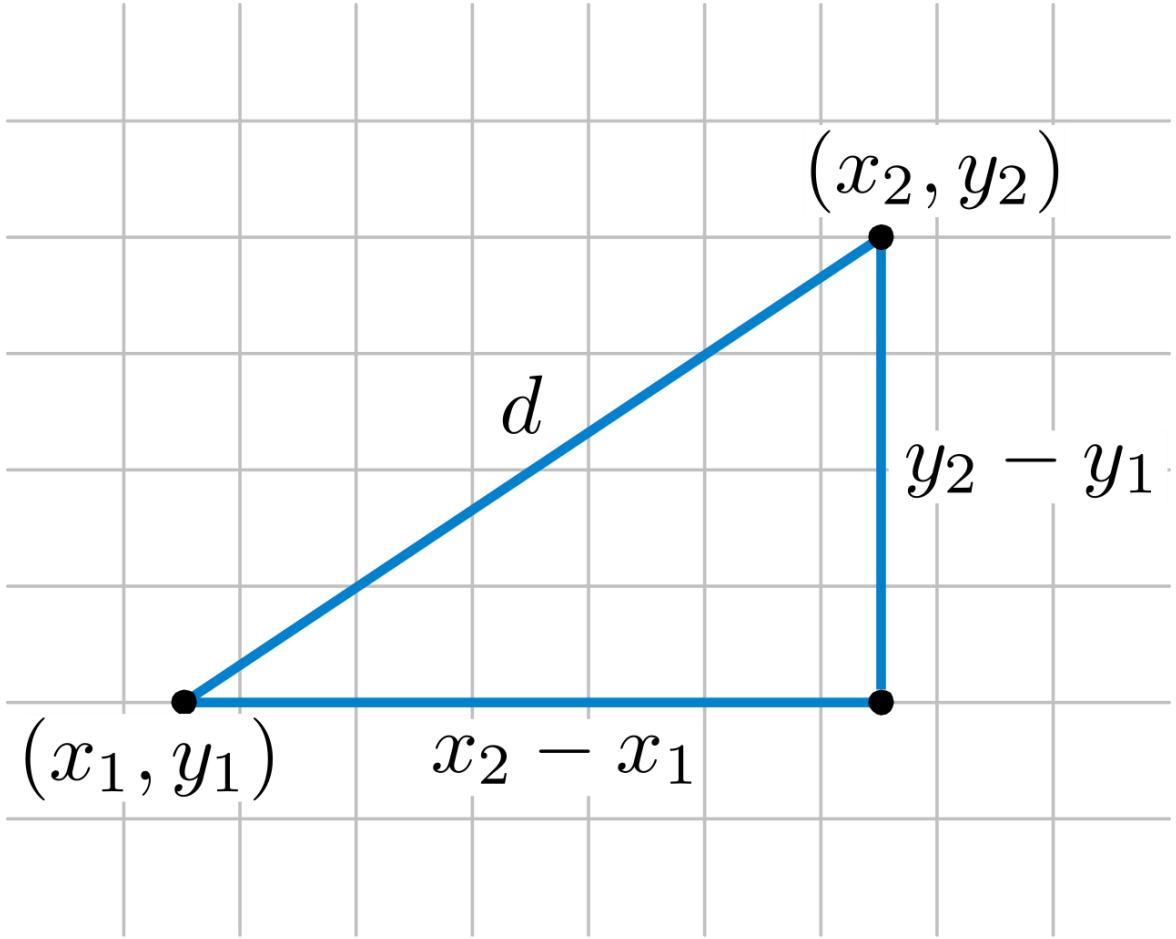
Las medidas de distancia comunes incluyen:

- Distancia euclidiana
- Distancia de Manhattan
- Distancia de Minkowski

K-Nearest Neighbors: distancias

- Distancia euclidiana

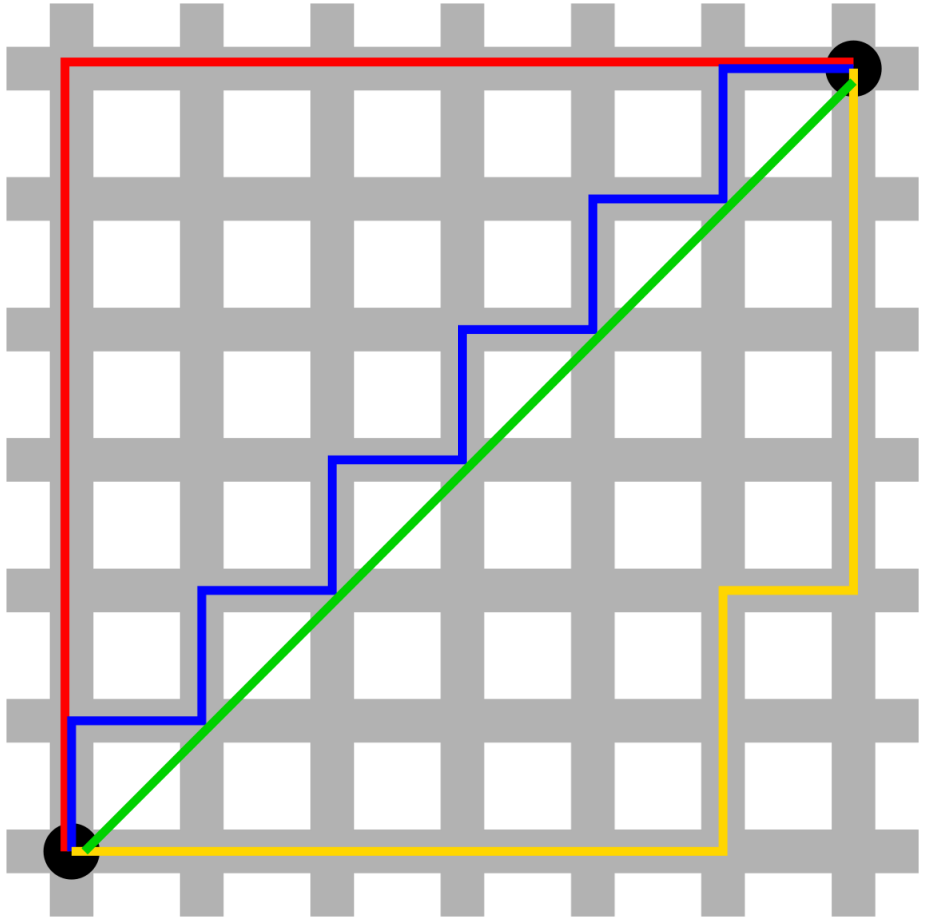
$$D_{\text{Euclidean}}(x, y) = \left(\sum_{i=1}^{n}(x_i - y_i)^2\right)^{1/2}$$



K-Nearest Neighbors: distancias

- Distancia de Manhattan

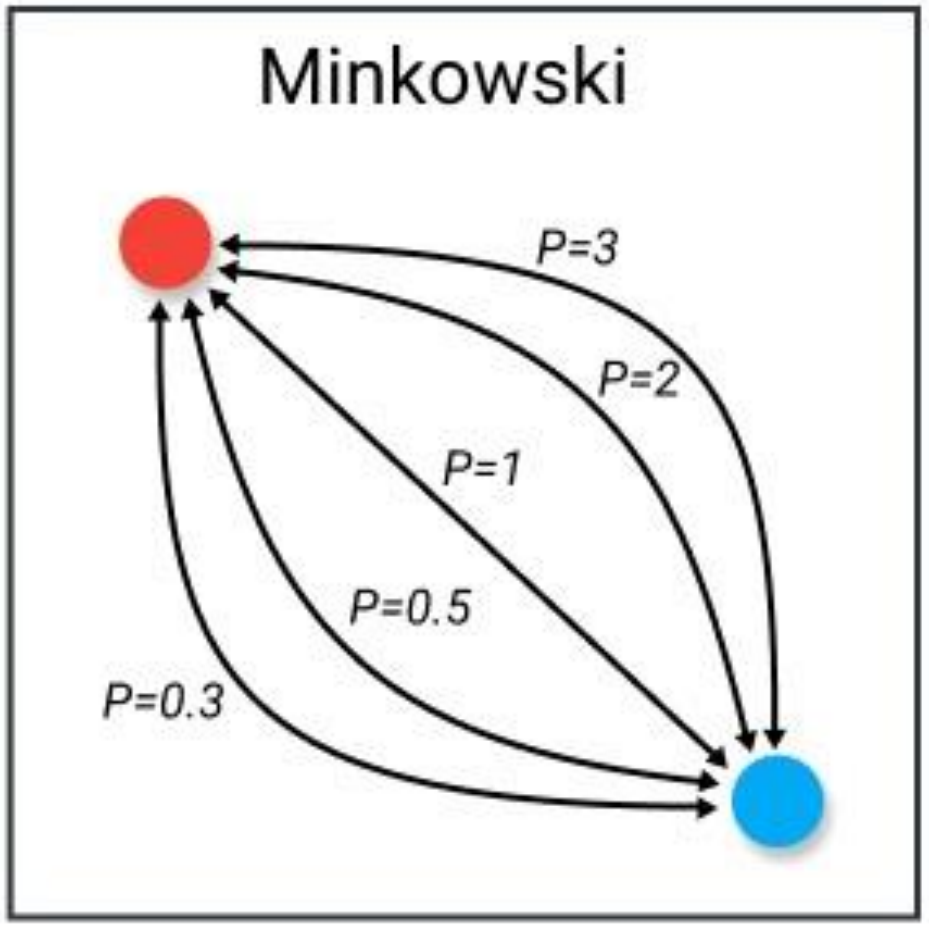
$$D_{\text{Manhattan}}(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$



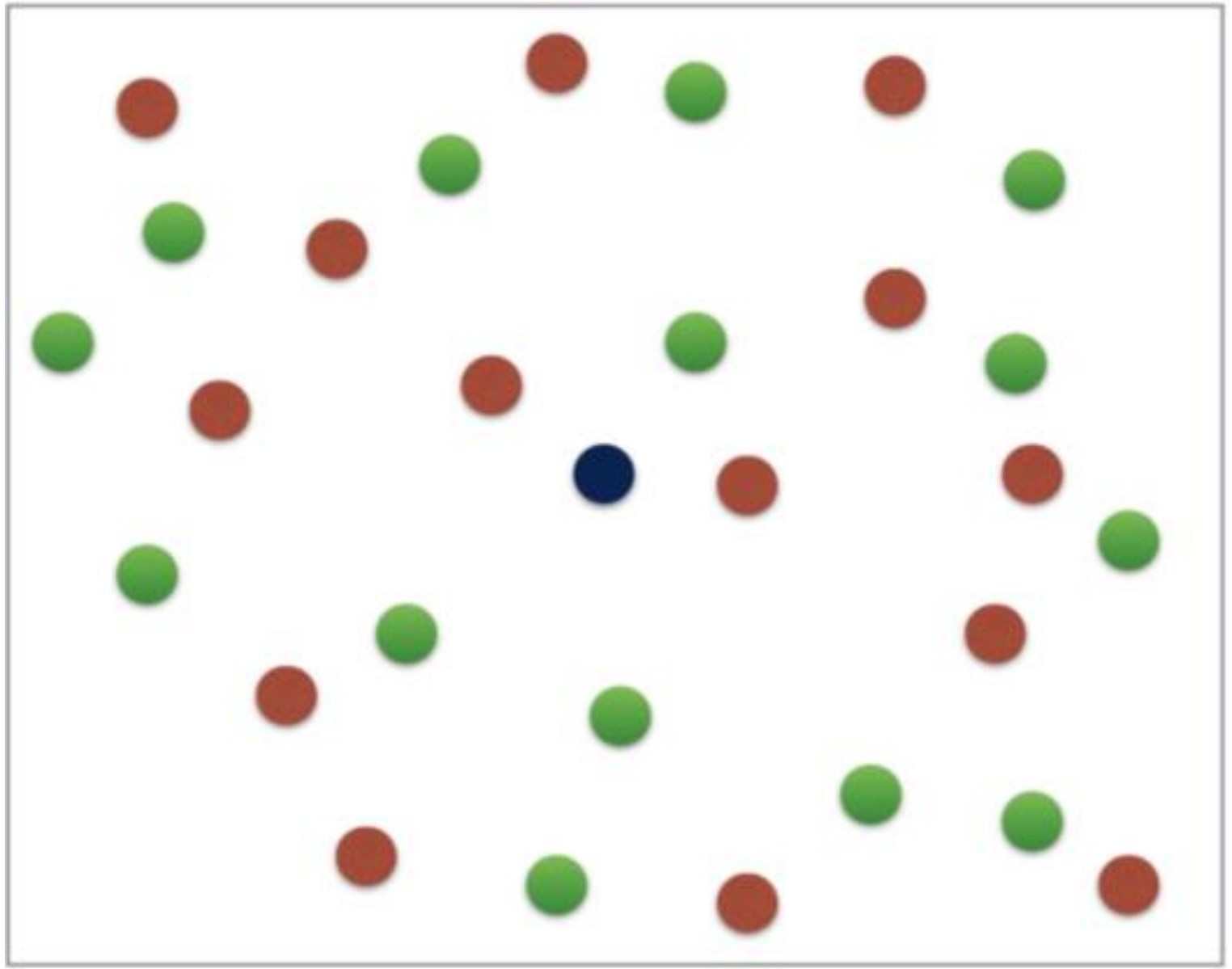
K-Nearest Neighbors: distancias

- Distancia de Minkowski

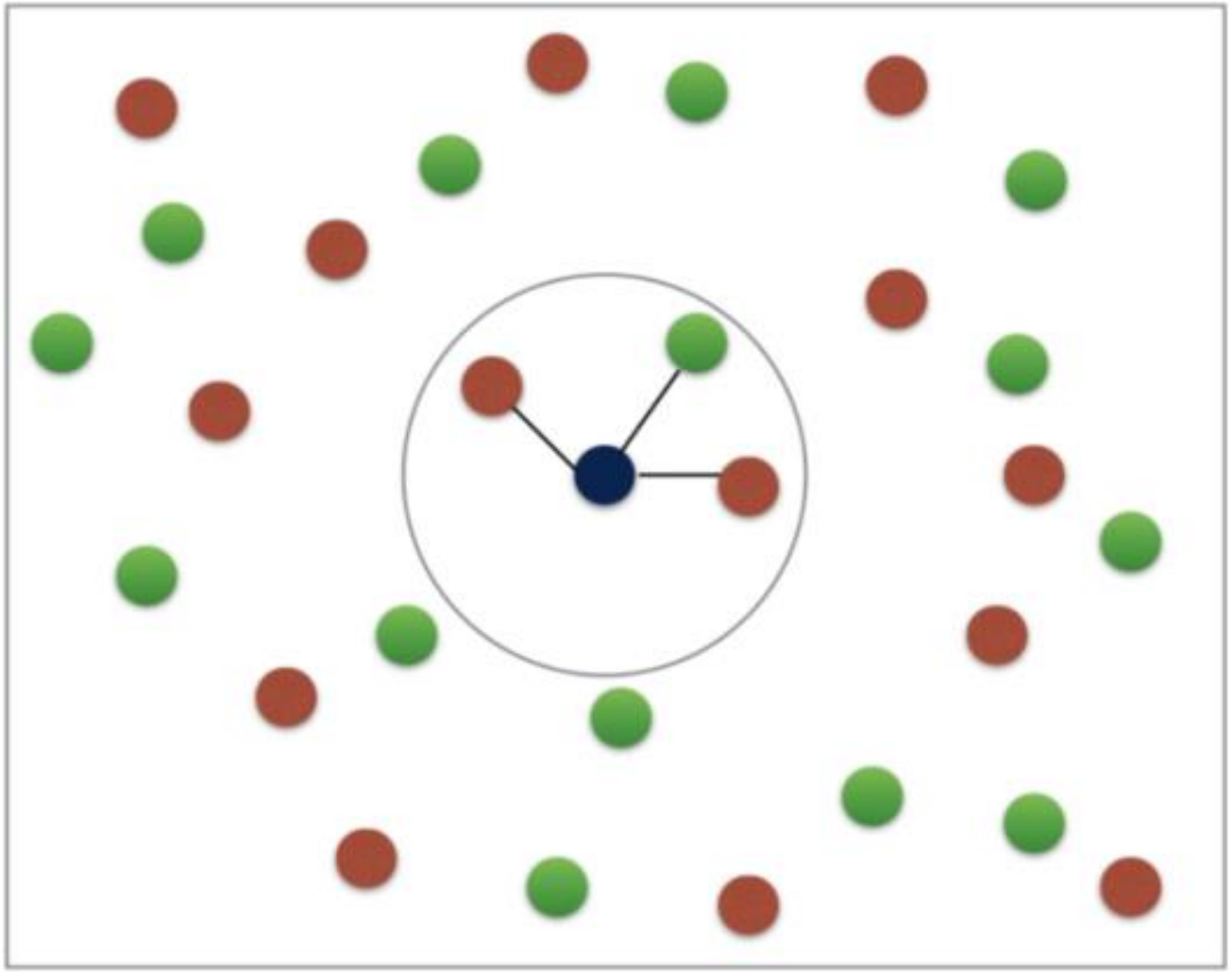
$$D(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$$



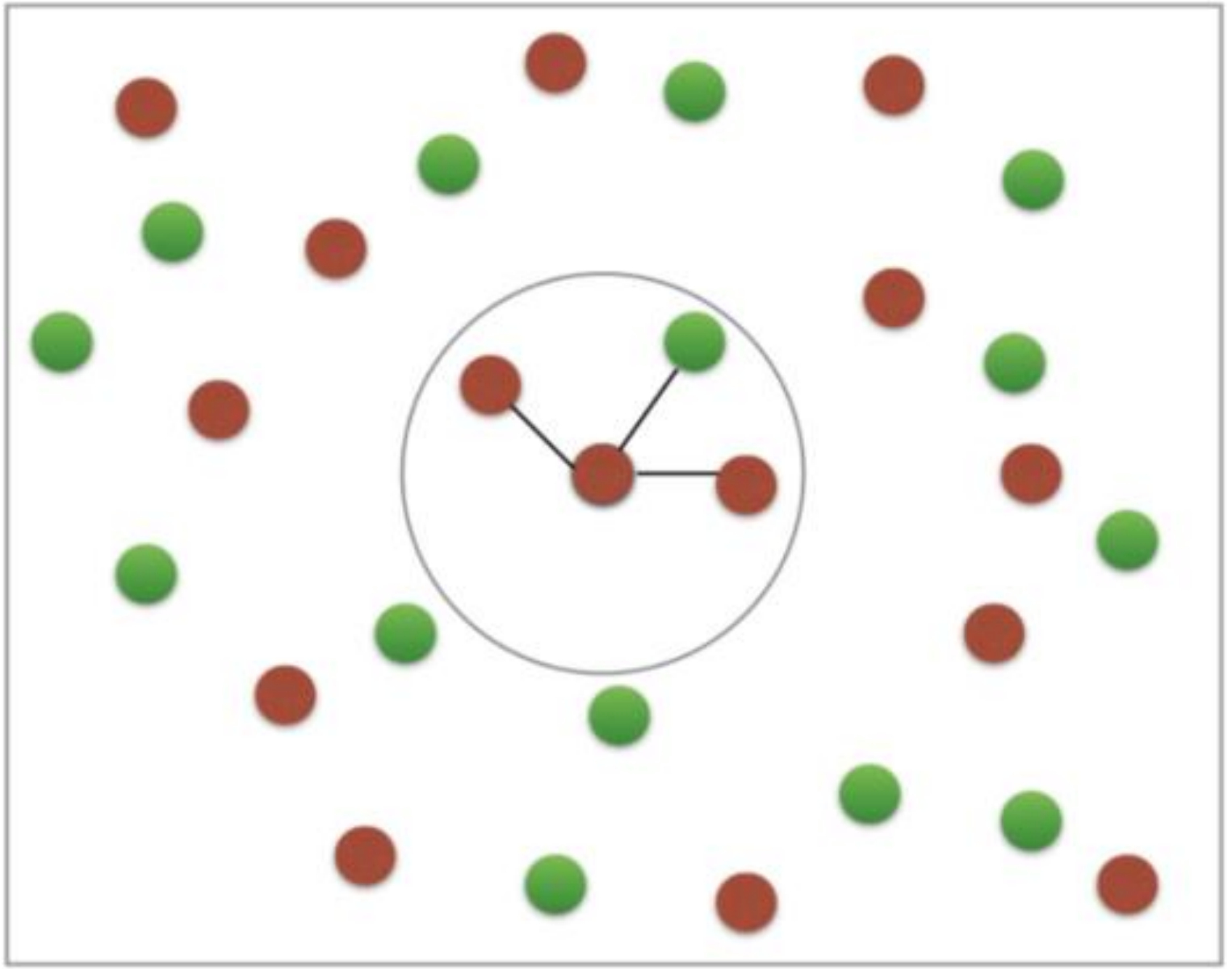
K-Nearest Neighbors: paso a paso



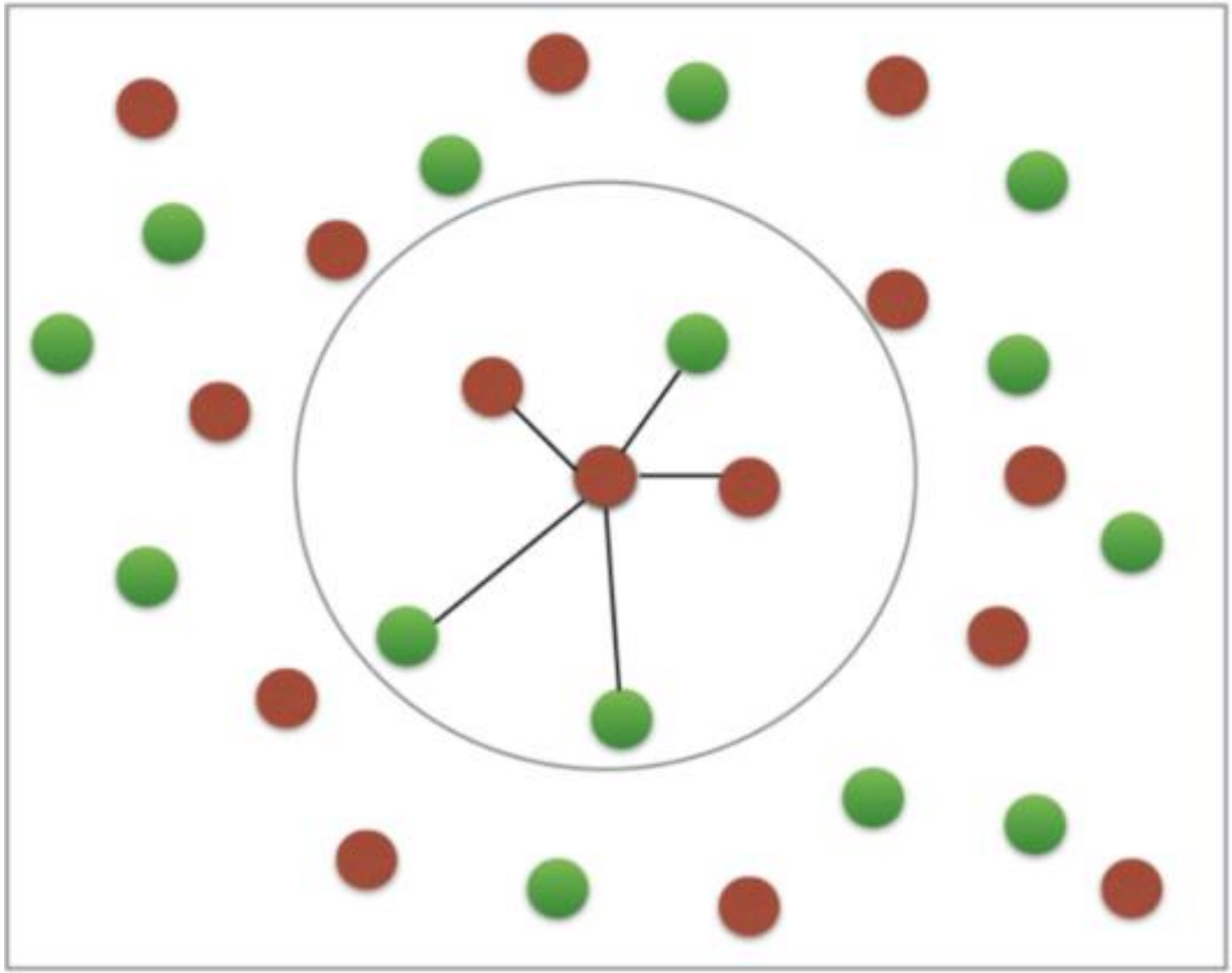
K-Nearest Neighbors: paso a paso



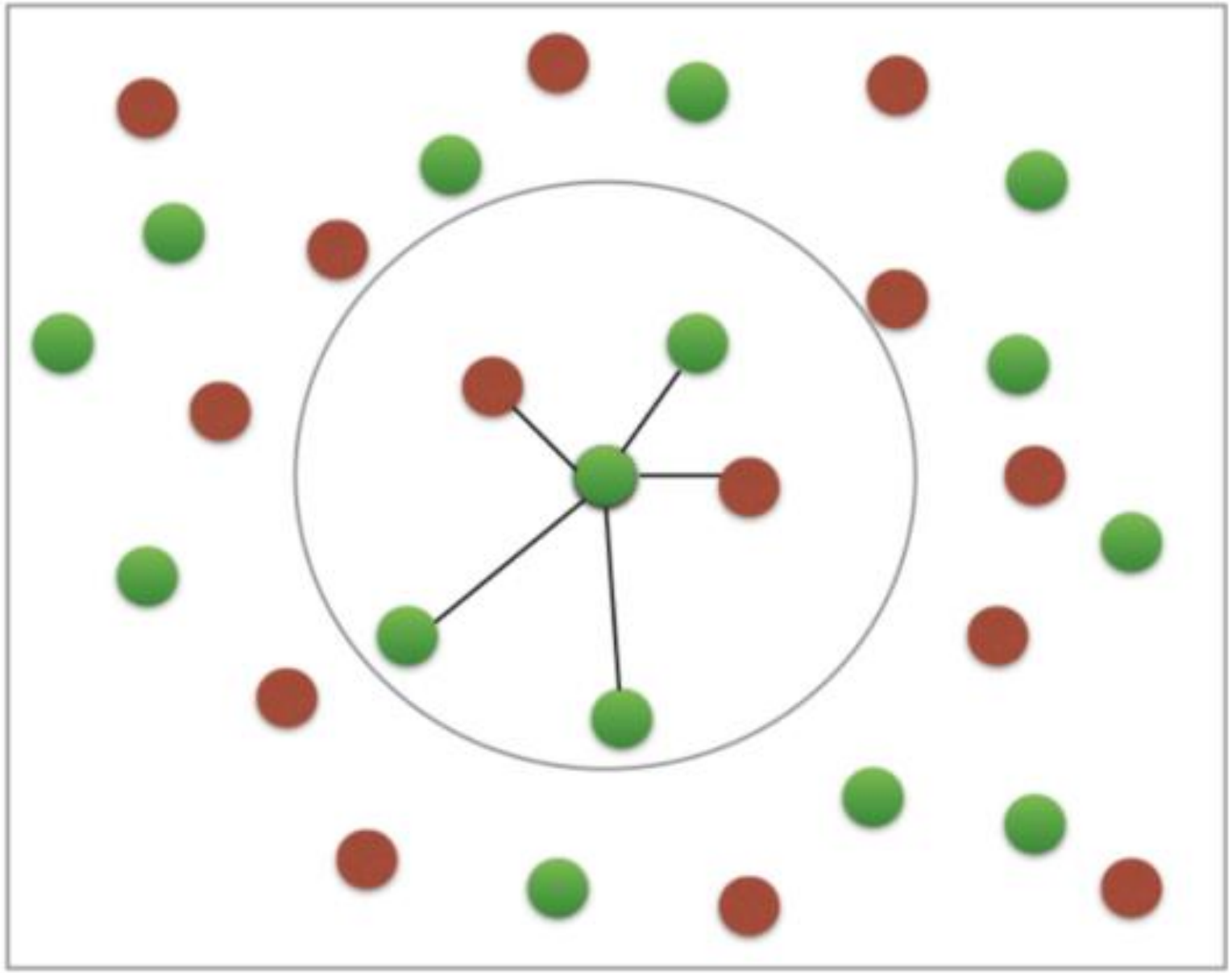
K-Nearest Neighbors: paso a paso



K-Nearest Neighbors: paso a paso



K-Nearest Neighbors: paso a paso



K-Nearest Neighbors: ventajas

Ventajas

- Simple de entender e implementar
- Funciona bien con datos no lineales.
- Versátil, se puede usar para clasificación y regresión.

Desventajas

- Computacionalmente costoso para grandes conjuntos de datos.
- Sensible a la escala de las características
- Puede sufrir la "maldición de la dimensionalidad" en espacios de alta dimensión.

Árboles de Decisión

Árboles de Decisión: definición

Clasificación jerárquica

- División de datos en nodos basados en características.
- Criterios de división: entropía, impureza de Gini.

Interpretación visual

- Facilidad de comprensión y explicación

Sobreajuste

- Importancia de la poda y ajuste de hiperparámetros

Árboles de Decisión: introducción

Los árboles de decisión son un tipo de algoritmo de aprendizaje supervisado que se utiliza tanto para **tareas** de clasificación como de regresión.

Funcionan creando una **estructura de árbol jerárquica**, donde cada **nodo** interno representa una "*decisión*" basada en una característica de los datos, cada **rama** representa un posible *resultado* de esa decisión, y cada **nodo hoja** representa la predicción final.

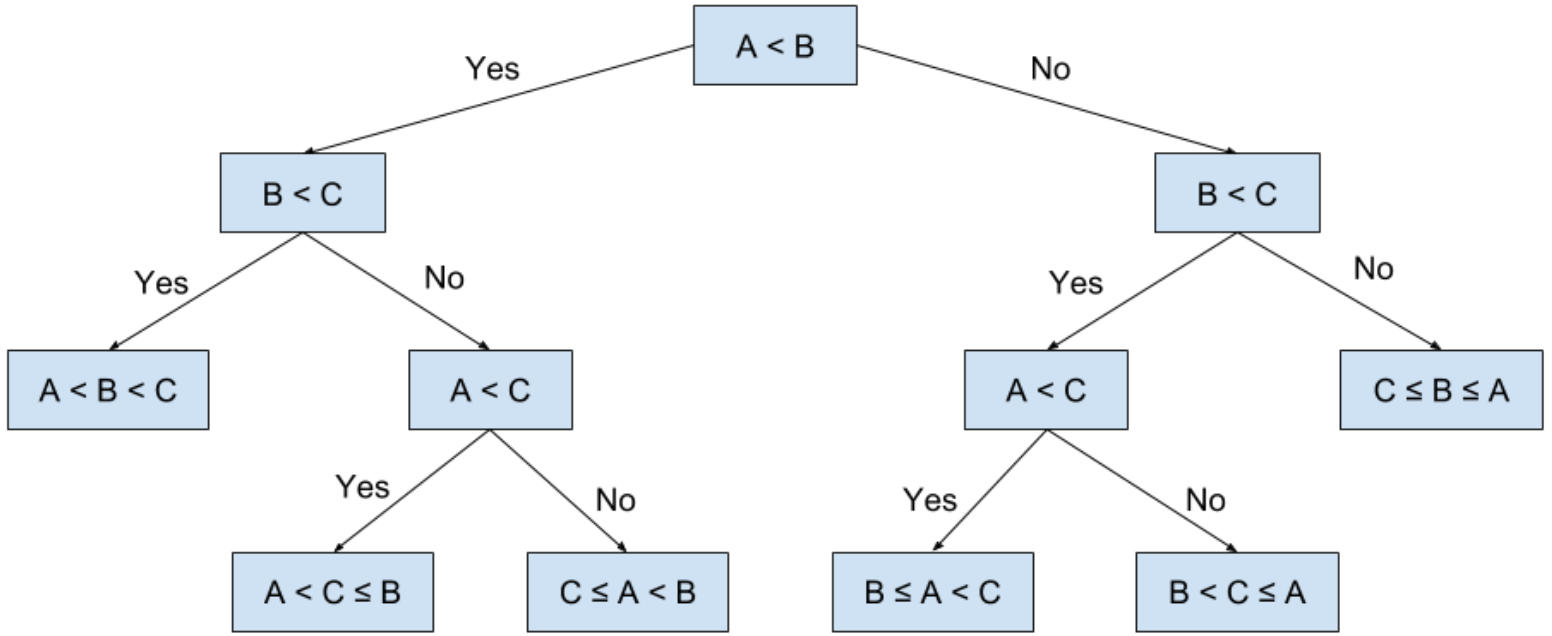
Los árboles de decisión pueden manejar tanto datos categóricos como numéricos.

Árboles de Decisión: Estructura jerárquica

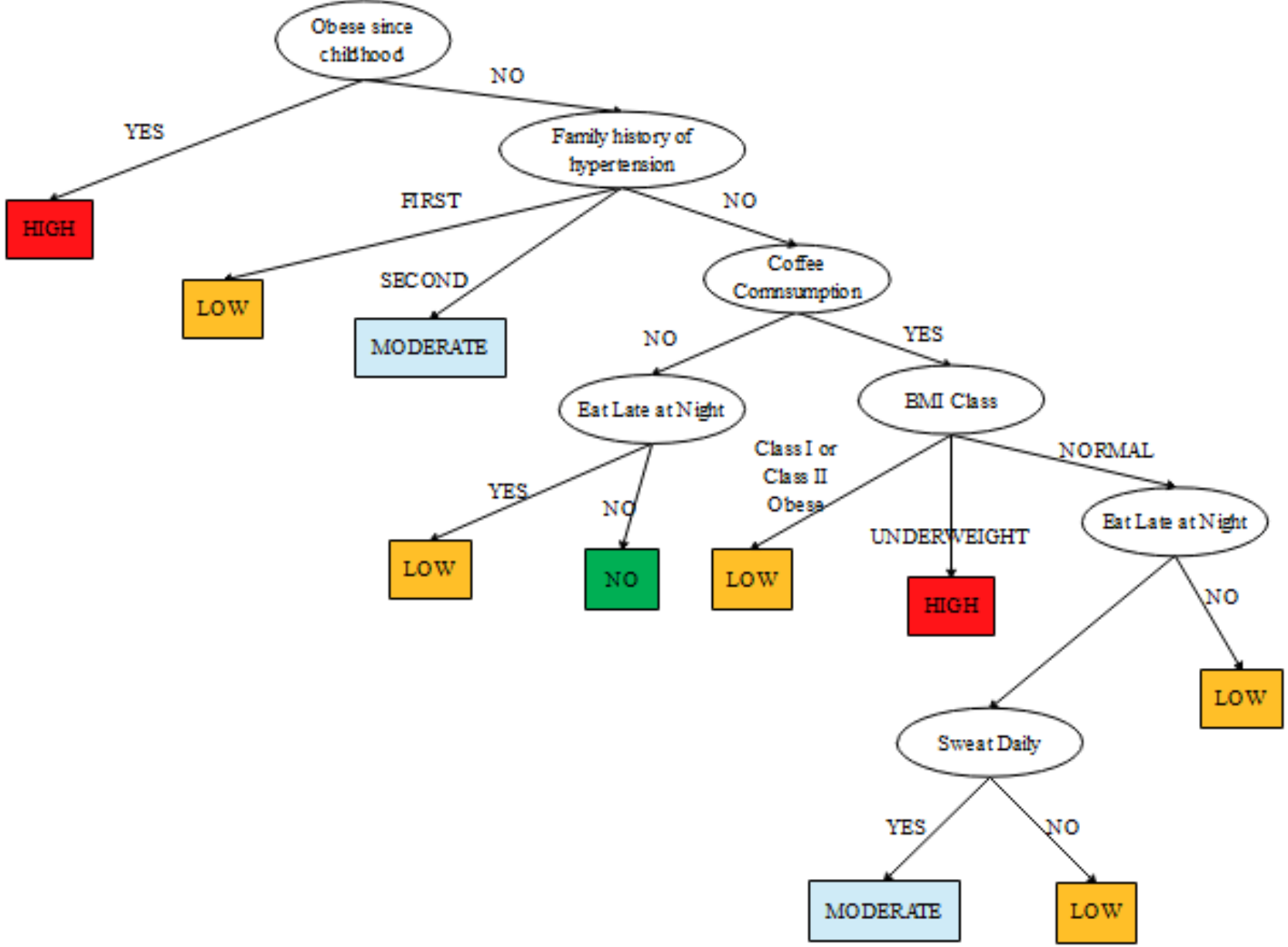
Un árbol de decisión comienza con un **nodo raíz** y se ramifica en nodos **internos** y nodos **hoja**.

- ✓Cada **nodo interno** representa una prueba en una característica, y las ramas representan los resultados de la prueba.
- ✓Los **nodos hoja** representan las predicciones finales.

Árboles de Decisión: Estructura jerárquica



Árboles de Decisión: Estructura jerárquica



Árboles de Decisión: Características

Decisiones basadas en características

En cada nodo interno, el algoritmo selecciona la característica que mejor divide los datos en **subconjuntos homogéneos**.

Los **criterios de división** comunes incluyen la entropía y la impureza de Gini.

Los árboles de decisión son **fáciles de entender** e interpretar, ya que se pueden **visualizar** como diagramas de árbol.

Árboles de Decisión: Sobreajuste (overfitting)

Los árboles de decisión pueden ser propensos al sobreajuste, especialmente si son muy **profundos**.

Técnicas como la **poda** (*pruning*) pueden ayudar a prevenir el sobreajuste.

Árboles de Decisión: matriz de confusión

		Predicted	
		Positive	Negative
Real	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

		Predicted	
		Positive	Negative
Real	Positive	35	237
	Negative	15	5113

Árboles de Decisión: matriz de confusión

NO	LOW	MODERATE	HIGH	
5	0	1	0	NO
1	17	0	2	LOW
0	2	6	3	MODERATE
0	0	0	8	HIGH

Random Forest

Random Forest : introducción

Random Forest es un algoritmo de aprendizaje automático versátil y poderoso, empleado tanto para tareas de clasificación como de regresión.

Se basa en la idea de **combinar múltiples árboles de decisión** para obtener predicciones más precisas y robustas.

Random Forest: definición

Métodos de ensamble

- Combinación de múltiples modelos para mejorar la precisión.
- Bagging: entrenamiento de árboles independientes con subconjuntos de datos.

Random Forest

- Bagging con selección aleatoria de características
- Importancia de las características

Ventajas

- Robustez
- Menor sobreajuste

Random Forest : Ensamble de árboles de decisión

- Random Forest construye múltiples árboles de decisión durante el entrenamiento.
- Cada árbol se entrena con un **subconjunto aleatorio** del conjunto de datos de entrenamiento, utilizando la técnica de "**bootstrapping**" (muestreo con reemplazo).
- Además, en cada **nodo de un árbol**, solo se considera un subconjunto aleatorio de las características para la división.

Random Forest : sobreajuste

Reducción del sobreajuste

- Al combinar las predicciones de múltiples árboles, Random Forest reduce el riesgo de sobreajuste, que es un problema común en los árboles de decisión individuales.

Importancia de las características

- Random Forest proporciona una medida de la importancia de cada característica en la predicción.
- Esto puede ser útil para la **selección de características** y la comprensión de los datos.

Random Forest : Robustez y Versatilidad

Random Forest es **robusto** ante valores atípicos y ruido en los datos.

También puede manejar datos con características faltantes.

Se puede utilizar tanto para tareas de clasificación como de regresión.

Funciona bien con datos de alta dimensión.

Máquinas de Soporte Vectorial (SVM)

Máquinas de Soporte Vectorial (SVM): definición

Clasificación basada en hiperplanos

- Búsqueda del hiperplano óptimo que maximiza el margen.
- Vectores de soporte: puntos clave para la definición del hiperplano.

Kernel trick

- Transformación de datos a espacios de alta dimensión.
- Clasificación no lineal.

Ventajas y desventajas

- Eficacia en altas dimensiones, complejidad computacional

Máquinas de Soporte Vectorial (SVM): introducción

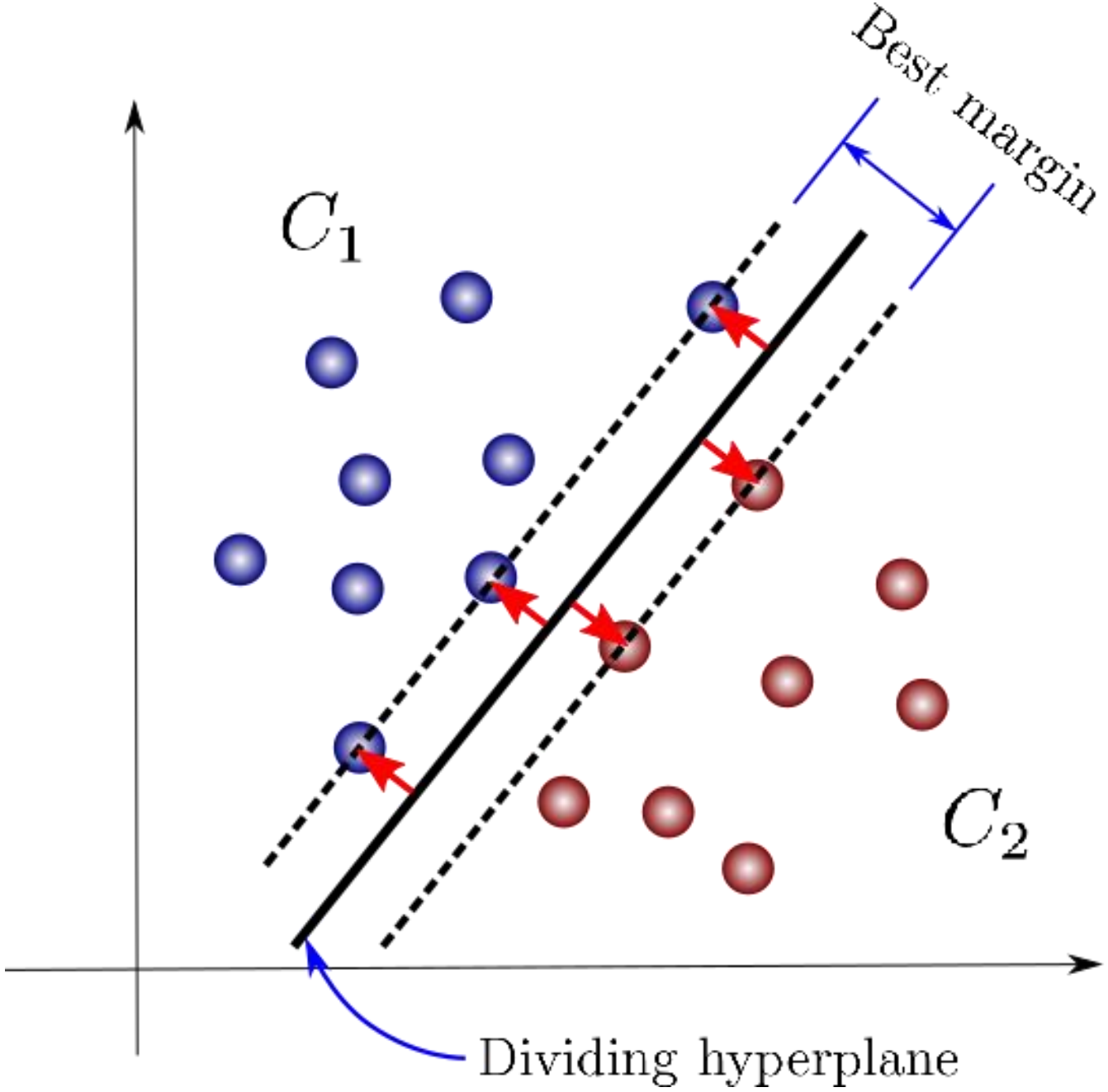
Las Máquinas de Soporte Vectorial (SVM, por sus siglas en inglés) son un algoritmo de aprendizaje supervisado potente y versátil, utilizado tanto para tareas de clasificación como de regresión.

Hiperplanos y márgenes

SVM busca encontrar el hiperplano óptimo que mejor **separe las diferentes clases** de datos.

El hiperplano se elige para **maximizar el margen**, que es la distancia entre el hiperplano y los puntos de datos más cercanos de cada clase (vectores de soporte).

Máquinas de Soporte Vectorial (SVM): introducción



Máquinas de Soporte Vectorial (SVM): Vectores de soporte

Los **vectores de soporte** son los puntos de datos que se encuentran más cerca del hiperplano.

Estos puntos son **cruciales** para definir el hiperplano y, por lo tanto, para el rendimiento del modelo.

Kernel trick

- ✓SVM puede manejar datos no lineales utilizando el "*kernel trick*".
- ✓Esta técnica *transforma los datos* a un espacio de mayor dimensión, donde es más fácil encontrar un **hiperplano lineal** que los separe.
- ✓Los *kernels* comunes incluyen el kernel lineal, el kernel polinómico y el *kernel* radial basis function (RBF).

Máquinas de Soporte Vectorial (SVM): versatilidad y robustez

- ✓SVM se puede utilizar tanto para clasificación como para regresión (SVR - **Support Vector Regression**).

- ✓Funciona bien en espacios de alta dimensión.

- ✓SVM es robusto ante valores atípicos y puede manejar datos con ruido.

Evaluación y Comparación de Modelos

Métricas de evaluación

- Precisión
- recall
- F1-score
- AUC-ROC

Técnicas de validación cruzada

- Evaluación robusta del rendimiento del modelo

Comparación de modelos

- Selección del modelo óptimo según el problema

Métricas de evaluación

- ✓La **precisión** de los algoritmos de clasificación se expresa en un porcentaje.
- ✓Mientras más cercano ese porcentaje a 100% más preciso es el modelo generado.
- ✓Sin embargo, un porcentaje 100% indica que el modelo podría estar demasiado ajustado a los datos (***overfitting***).

Ejercicio práctico #1

Regresión logística:
Cómo calcular una probabilidad con la función sigmoidea

developers.google.com

Ejercicio práctico #2

Regresión logística: pérdida y regularización

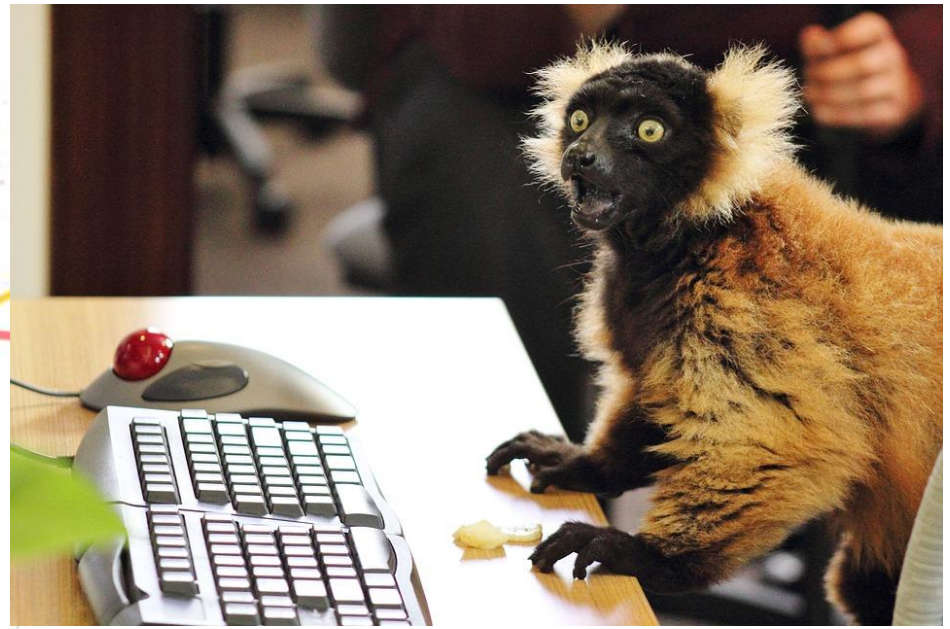
Tu tarea: Leer el ejemplo de regresión logística.

developers.google.com

Ejercicio práctico #3

Regresión logística: developers.google.com

Pon a prueba tus conocimientos



scikit-learn

Machine Learning in Python

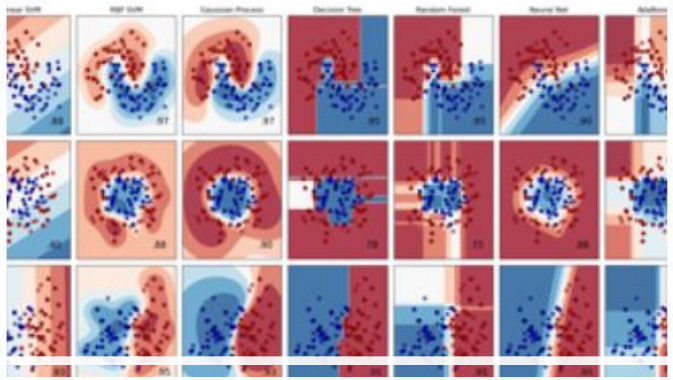
Getting Started | Release Highlights for 1.6

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.
Algorithms: Gradient boosting, nearest neighbors, random forest, logistic regression, and more...

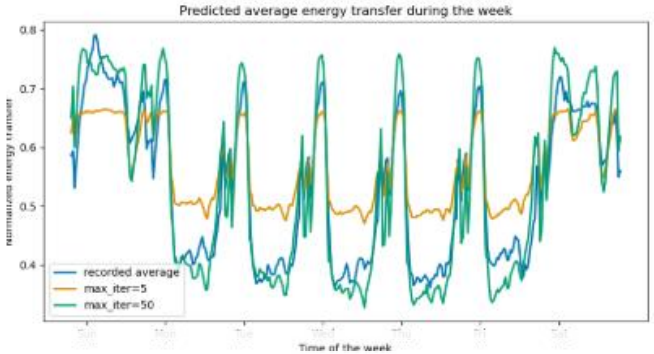


Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, stock prices.
Algorithms: Gradient boosting, nearest neighbors, random forest, ridge, and more...

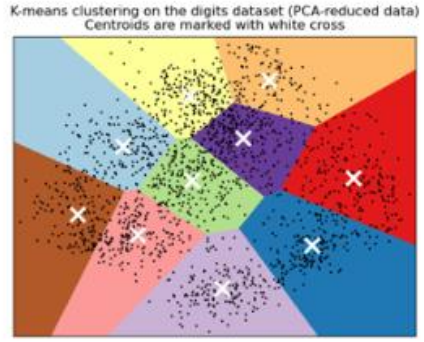


Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, grouping experiment outcomes.
Algorithms: k-Means, HDBSCAN, hierarchical clustering, and more...



Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, increased efficiency.

Model selection

Comparing, validating and choosing parameters and models.

Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text for

Conclusiones...

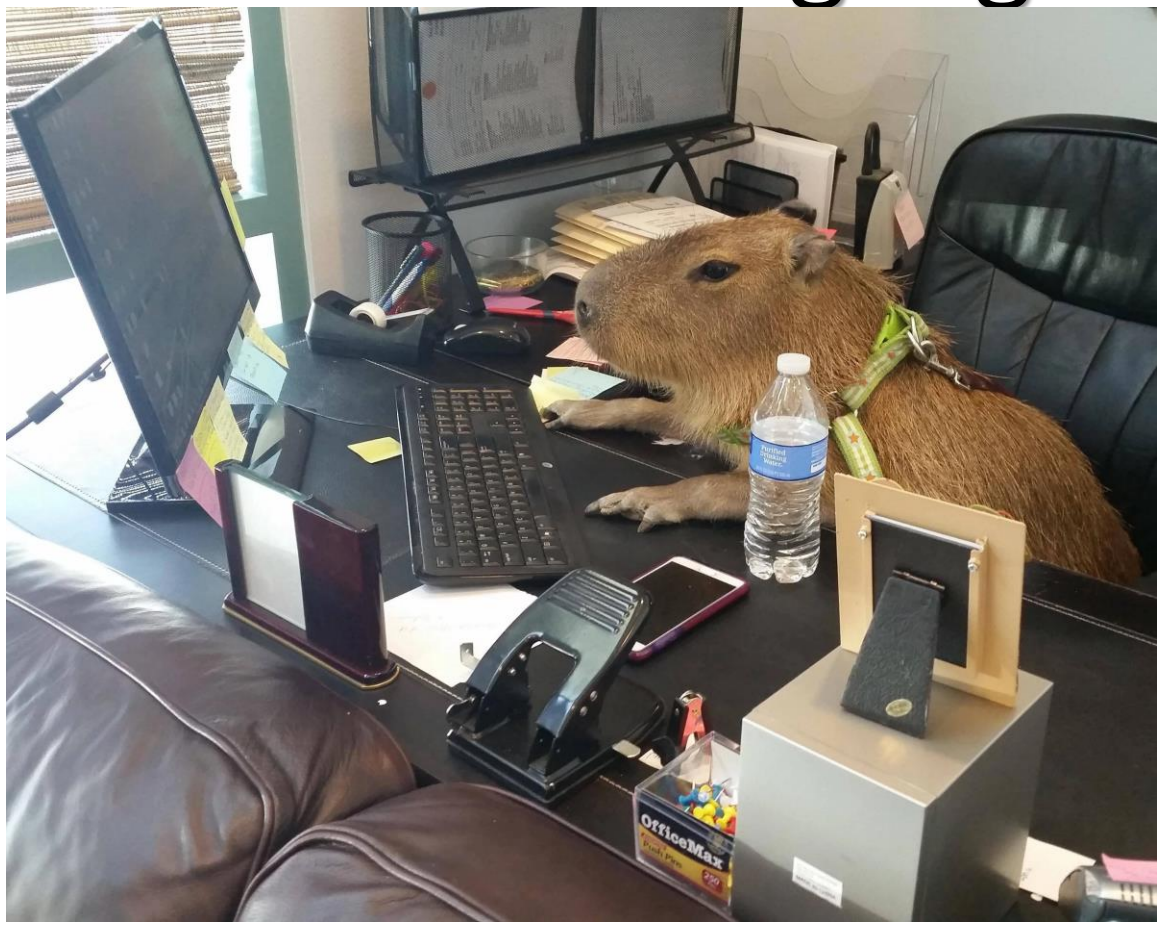
Cada algoritmo de clasificación (Regresión Logística, KNN, Árboles de Decisión, Random Forest, SVM) tiene sus propias fortalezas y debilidades, lo que los hace adecuados para diferentes tipos de problemas y conjuntos de datos.

La Regresión Logística es excelente para problemas de clasificación binaria, KNN para datos no lineales pero con alta demanda computacional, los Árboles de Decisión para la interpretabilidad, Random Forest para la robustez y SVM para datos de alta dimensión.

Es crucial comprender las características de cada algoritmo para seleccionar el más apropiado para cada tarea específica.

Ejercicio práctico

colab.research.google.com

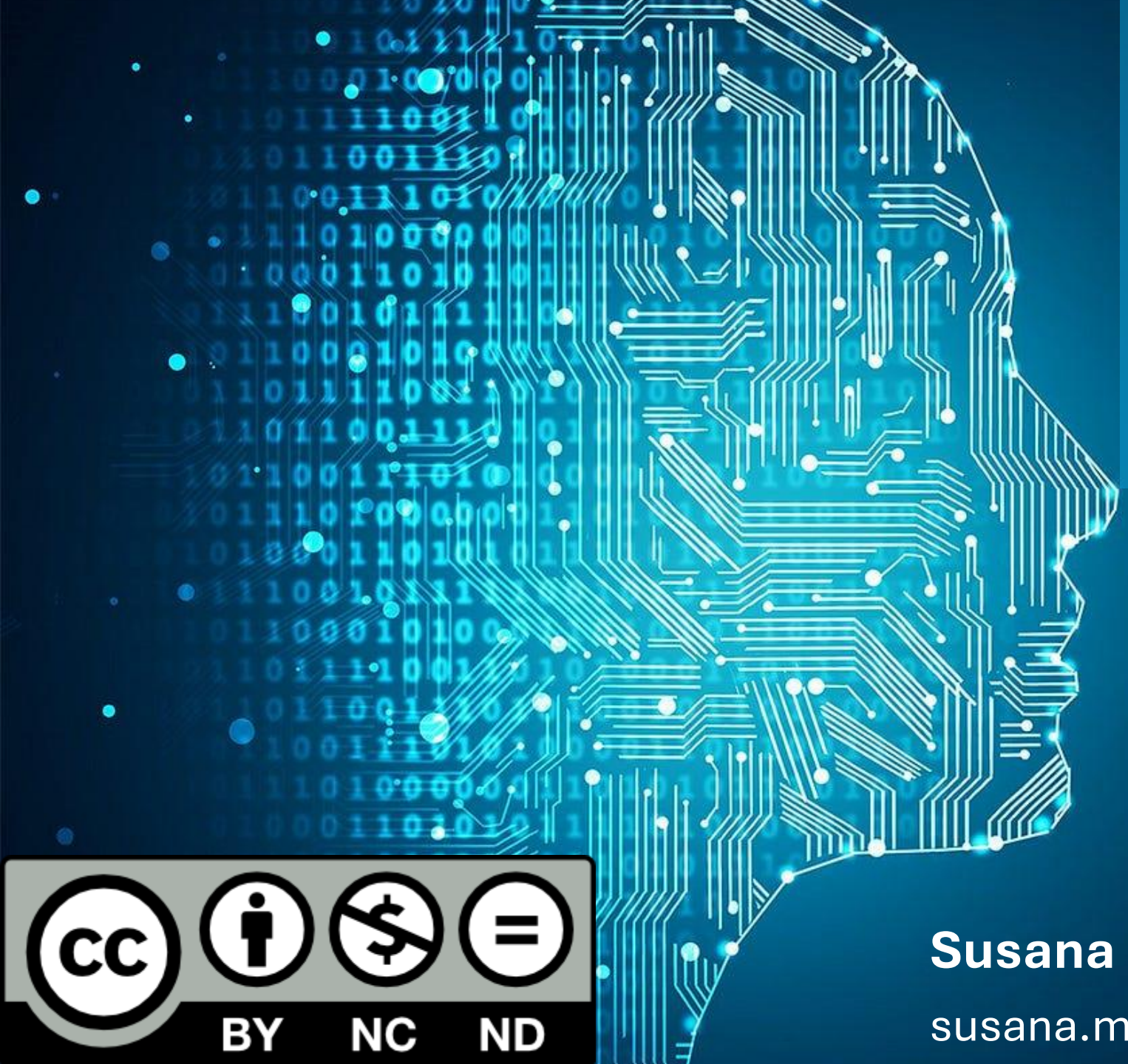


Ejercicio práctico: Google Colaboratory (*Colabs*)

- Página oficial: https://colab.google/
- Abrir Colab (incluye tutorial): https://colab.research.google.com/
- Guía para EDA: https://colab.research.google.com/github/Tanu-N-Prabhu/Python/blob/master/Exploratory_data_Analysis.ipynb
- Guía / tutorial para Selección de características con **scikit-learn**: https://www.datacamp.com/tutorial/feature-selection-python

Referencias

- “Regresión logística| Machine Learning | Google for Developers”. Consultado: el 26 de marzo de 2025. [En línea]. Disponible en: https://developers.google.com/machine-learning/crash-course/linear-regression?hl=es-419
- “scikit-learn: Machine Learning in Python”. Consultado: el 20 de febrero de 2025. [En línea]. Disponible en: https://scikit-learn.org/stable/
- Información e ideas presentadas basadas en el conocimiento general de modelos de lenguaje de IA. Gemini 2.9 Flash. Consultado: el 3 de abril de 2025. [En línea].



Machine Learning

Susana Medina Gordillo

susana.medina@correounivalle.edu.co