# ## KNN

PASOS IMPORTANTES

librerias

# Importar les llibreries necessàries from sklearn.model\_selection import train\_test\_split from sklearn.neighbors import KNeighborsClassifier from sklearn.preprocessing import StandardScaler from sklearn.metrics import precision\_score, recall\_score, f1\_score, accuracy\_score, confusion\_matrix

## **#### Definir variables objetivo**

X = artists\_cartelera.drop(['top'], axis=1)

y = artists\_cartelera['top']

## DIVIDIR DATOS TRAIN Y TEST

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

## Escalar los datos

**# Inicializar el escalador (StandardScaler)**

**scaler = StandardScaler()**

**# Ajustar y transformar los datos de entrenamiento**

**X\_train\_scaled = scaler.fit\_transform(X\_train)**

**# Transformar los datos de prueba (usando los parámetros aprendidos del conjunto de entrenamiento)**

**X\_test\_scaled = scaler.transform(X\_test)**

## CREAR MODELO:

knn = KNeighborsClassifier(n\_neighbors=3)

## TRAIN MODELO

knn.fit(X\_train, y\_train)

## PREDICCIONES

y\_pred = knn.predict(X\_test)

## 

## METRICAS

# Cálculo de las métricas

precision = precision\_score(y\_test, y\_pred)

recall = recall\_score(y\_test, y\_pred)

f1 = f1\_score(y\_test, y\_pred)

accuracy = accuracy\_score(y\_test, y\_pred)

# Imprimir los resultados

print("Precisión:", precision)

print("Recall:", recall)

print("F1-Score:", f1)

print("Accuracy:", accuracy)

matriu\_confusion = confusion\_matrix(y\_test, y\_pred)

print("Matriz de Confusión:\n", matriu\_confusion)

# REGRESSOR:

# Importar les llibreries necessàries from sklearn.model\_selection import train\_test\_split from sklearn.neighbors import KNeighborsRegressor from sklearn.preprocessing import # Importar les llibreries necessàries

from sklearn.model\_selection import train\_test\_split

from sklearn.neighbors import KNeighborsRegressor

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import mean\_squared\_error, r2\_score

# Definir les variables X (features) i y (target)

X = artists\_cartelera.drop(['top'], axis=1) # Assumint que 'top' és la columna target (valors continus)

y = artists\_cartelera['top']

# Dividir les dades en conjunt d'entrenament i conjunt de test (70% entrenament, 30% test)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Escalar les dades

scaler = StandardScaler()

# Ajustar i transformar les dades de l'entrenament

X\_train\_scaled = scaler.fit\_transform(X\_train)

# Transformar les dades de test amb l'escalador ajustat a l'entrenament

X\_test\_scaled = scaler.transform(X\_test)

# Crear el model KNN per a regressió

knn = KNeighborsRegressor(n\_neighbors=3)

# Entrenar el model

knn.fit(X\_train\_scaled, y\_train) # Utilitzem X\_train\_scaled

# Fer prediccions amb el conjunt de test

y\_pred = knn.predict(X\_test\_scaled) # Utilitzem X\_test\_scaled

# Càlcul de les mètriques per a regressió

mse = mean\_squared\_error(y\_test, y\_pred) # Mean Squared Error (MSE)

r2 = r2\_score(y\_test, y\_pred) # R² Score

# Imprimir els resultats

print("Mean Squared Error:", mse)

print("R² Score:", r2)

# Opcional: Imprimir les primeres prediccions i els valors reals per comparar

print("Primeres prediccions:", y\_pred[:10])

print("Primeres veritats reals:", y\_test[:10])