**EDA**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import LabelEncoder, StandardScaler, MinMaxScaler

from imblearn.over\_sampling import SMOTE

from imblearn.under\_sampling import TomekLinks

from imblearn.combine import SMOTETomek

**Classification**

**Libraries:**

import pandas as pd

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

from sklearn.naive\_bayes import GaussianNB

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, confusion\_matrix, ConfusionMatrixDisplay

import joblib

**Evaluation Metrics:**

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

print(f"Accuracy: {accuracy}")

precision = precision\_score(y\_test, y\_pred, average='binary')

print(f"Precision: {precision}")

recall = recall\_score(y\_test, y\_pred, average='binary')

print(f"Recall: {recall}")

f1 = f1\_score(y\_test, y\_pred, average='binary')

print(f"F1 Score: {f1}")

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

ConfusionMatrixDisplay(conf\_matrix).plot()

**Regression**

**Libraries:**

import pandas as pd

import numpy as np

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

from sklearn.neighbors import KNeighborsRegressor

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

import joblib

**Evaluation Metrics:**

mae = mean\_absolute\_error(y\_test, y\_pred)

print(f"Mean Absolute Error (MAE): {mae}")

mse = mean\_squared\_error(y\_test, y\_pred)

print(f"Mean Squared Error (MSE): {mse}")

rmse = np.sqrt(mse)

print(f"Root Mean Squared Error (RMSE): {rmse}")

r2 = r2\_score(y\_test, y\_pred)

print(f"R-squared (R²): {r2}")

mape = mean\_absolute\_percentage\_error(y\_test, y\_pred)

print(f"Mean Absolute Percentage Error (MAPE): {mape}")

**Clustering**

**Libraries:**

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

from sklearn.cluster import KMeans

from sklearn.metrics import silhouette\_score, davies\_bouldin\_score, calinski\_harabasz\_score

**Evaluation Metrics:**

sil\_score = silhouette\_score(df, clusters)

print(f"Silhouette Score: {sil\_score:.3f}")

db\_index = davies\_bouldin\_score(df, clusters)

print(f"Davies-Bouldin Index: {db\_index:.3f}")

ch\_index = calinski\_harabasz\_score(df, clusters)

print(f"Calinski-Harabasz Index: {ch\_index:.3f}")

inertia = kmeans.inertia\_

print(f"Inertia (within-cluster sum of squares): {inertia:.2f}")