#Linear

import numpy as np

import pandas as pd

from sklearn import datasets

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

from sklearn.linear\_model import LinearRegression

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

import matplotlib.pyplot as plt

import seaborn as sns

diabetes = datasets.load\_diabetes()

diabetes\_df = pd.DataFrame(data=diabetes.data, columns=diabetes.feature\_names)

diabetes\_df['target'] = diabetes.target

#EDA = shape, columns, null values , summary statistics

print("\nDataset Shape:", diabetes\_df.shape)

print("\nColumns:", diabetes\_df.columns)

print("\nInfo:")

print(diabetes\_df.info())

print("\nNull Values:")

print(diabetes\_df.isnull().sum())

print("\nSummary Staistics ",diabetes\_df.describe())

plt.figure(figsize=(8, 6))

sns.histplot(diabetes\_df['target'], bins=30, kde=True, color='blue')

plt.title('Distribution of Diabetes Progression')

plt.xlabel('Diabetes Progression')

plt.ylabel('Frequency')

plt.show()

#pairplot

plt.figure(figsize=(12, 10))

sns.pairplot(diabetes\_df, diag\_kind='kde')

plt.suptitle("Pairplot of Diabetes Dataset", y=1.02)

plt.show()

#Linear Regression

# Split the dataset into features (X) and target (y)

X = diabetes\_df.drop('target', axis=1) # Features

y = diabetes\_df['target'] # Target (continuous variable)

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Initialize the Linear Regression model

model = LinearRegression()

# Train the model on the training data

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

# Make predictions on the test data

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

# Evaluate the model's performance

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

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

print("\nMean Squared Error (MSE):", mse)

print("R-squared Score:", r2)

# Plot predicted vs actual values

plt.figure(figsize=(8, 6))

plt.scatter(y\_test, y\_pred, color='blue')

plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], linestyle='--', color='red')

plt.xlabel('Actual Diabetes Progression')

plt.ylabel('Predicted Diabetes Progression')

plt.title('Actual vs Predicted (Diabetes Dataset)')

plt.show()