import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# Load the dataset

dataset = pd.read\_csv(r'D:\Samsom - All Data\Naresh IT Institute\New folder\Salary\_Data.csv')

# Check the shape of the dataset

print("Dataset Shape:", dataset.shape) # (30, 2)

# Feature selection (independent variable x and dependent variable)

x = dataset.iloc[:, :-1] # Years of experience (Independent Variable)

y = dataset.iloc[:, -1] # Salary (Dependent variable)

# Split the dataset into training and testing sets (80% training)

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

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=0)

# Reshape x\_train and x\_test into 2D arrays if they are single

x\_train = x\_train.values.reshape(-1, 1)

x\_test = x\_test.values.reshape(-1, 1)

# Predicting the results for the test set

from sklearn.linear\_model import LinearRegression

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

regressor = LinearRegression()

regressor.fit(x\_train, y\_train)

y\_pred = regressor.predict(x\_test)

# Compare predicted and actual salaries from the test set

comparison = pd.DataFrame({'Actual': y\_test, 'Predicted': y\_pred})

print(comparison)

# Visualizing the Training set results

plt.scatter(x\_test, y\_test, color = 'red') # Real salary

plt.plot(x\_train, regressor.predict(x\_train), color = 'blue')

plt.title('Salary vs Experience (Training set)')

plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.show()

m\_slope = regressor.coef\_

print(m\_slope)

c\_intercept = regressor.intercept\_

print(c\_intercept)

y\_12 = m\_slope\*12+c\_intercept

print(y\_12)

bias = regressor.score(x\_train,y\_train)

print(bias)

variance = regressor.score(x\_test,y\_test)

print(variance)

# statistic concept need to add on the code

dataset.mean() # this will give mean of entire dataframe

dataset['Salary'].mean() # this will give us mean of that particular column

# Median

dataset.median() # this will give median of entire dataframe

dataset['Salary'].median() # this will give us median of that particular column

# Mode

dataset['Salary'].mode() # this will give us mode of that particular column

# Variance

dataset.var() # this will give variance of entire dataframe

dataset['Salary'].var() # this give us variance of that particular column

# Standard deviation

dataset.std() # this will give standard deviation of entire dataframe

dataset['Salary'].std() # this will give us standard deviation of that particular column

# Coefficient of variation(cv)

# for calculating cv we have to import a library first

from scipy.stats import variation

variation(dataset.values) # this will give cv of entire dataframe

variation(dataset['Salary']) # this will give us cv of that particular column

# Correlation

dataset.corr() # this will give correlation of entire dataframe

dataset['Salary'].corr(dataset['YearsExperience']) # this will give us correlation between these

# Skewness

dataset.skew() # this will give skewness of entire dataframe

dataset['Salary'].skew() # this will give us skewness of that particular column

# Standard Error

dataset.sem() # this will give standard error of entire dataframe

dataset['Salary'].sem() # this will give us standard error of that particular column

# Z-score

# for calculating Z-score we have to import a library first

import scipy.stats as stats

dataset.apply(stats.zscore) # this will give Z-score of entire dataframe

stats.zscore(dataset['Salary']) # this will give us Z-score of that particular column

# Degree of Freedom

a = dataset.shape[0] # this will gives us no.of rows

b = dataset.shape[1] # this will give us no.of columns

degree\_of\_freedom = a-b

print(degree\_of\_freedom) # this will give us degree of freedom for entire dataset

# sum of squer regresso (SSR)

#First we have to separate dependent and independent variables

y\_mean = np.mean(y)

SSR = np.sum((y\_pred-y\_mean)\*\*2)

print(SSR)

# SSE

y = y[0:6]

SSE = np.sum((y-y\_pred)\*\*2)

print(SSE)

# SST

mean\_total = np.mean(dataset.values)# here df.to\_nump()will

SST = np.sum((dataset.values-mean\_total)\*\*2)

print(SST)

# R2 SQUER

r\_square = 1 - (SSR/SST)

r\_square

# Save the trained model to disk

import pickle

filename = 'linear\_regression\_model.pk1'

with open(filename, 'wb') as file:

pickle.dump(regressor, file)

print("Model has been pickled and saved as linear\_regression\_model.pk1")

import os

print(os.getcwd())