#Rohitkumar pandey 211P002

import matplotlib.pyplot as plt

from matplotlib.figure import Figure

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

import pandas as pd

import numpy as np

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

from sklearn.linear\_model import LinearRegression

import tkinter as tk

df = pd.read\_csv("placement.csv")

root = tk.Tk()

root.title("CGPA vs Package Prediction")

root.geometry("800x600")

fig = Figure(figsize=(8, 6))

scatter\_plot = fig.add\_subplot(111)

x = df.iloc[:, 0].values.reshape(-1, 1)

y = df.iloc[:, -1].values

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=2)

model = LinearRegression()

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

model.score(x\_test, y\_test)

def predict\_package():

try:

cgpa = float(entry\_cgpa.get())

predicted\_package = model.predict([[cgpa]])

label\_prediction.config(text=f"Predicted Package: {predicted\_package[0]:.2f} LPA")

except ValueError:

label\_prediction.config(text="Please enter a valid CGPA")

label\_cgpa = tk.Label(root, text="Enter CGPA:")

label\_cgpa.pack(pady=10)

entry\_cgpa = tk.Entry(root, width=10)

entry\_cgpa.pack()

button\_predict = tk.Button(root, text="Predict Package", command=predict\_package)

button\_predict.pack(pady=10)

label\_prediction = tk.Label(root, text="")

label\_prediction.pack(pady=10)

scatter\_plot.scatter(df['cgpa'], df['package'])

scatter\_plot.plot(x\_train, model.predict(x\_train), color='red') # Use x\_train and predict on it

scatter\_plot.set\_xlabel('CGPA')

scatter\_plot.set\_ylabel('Package')

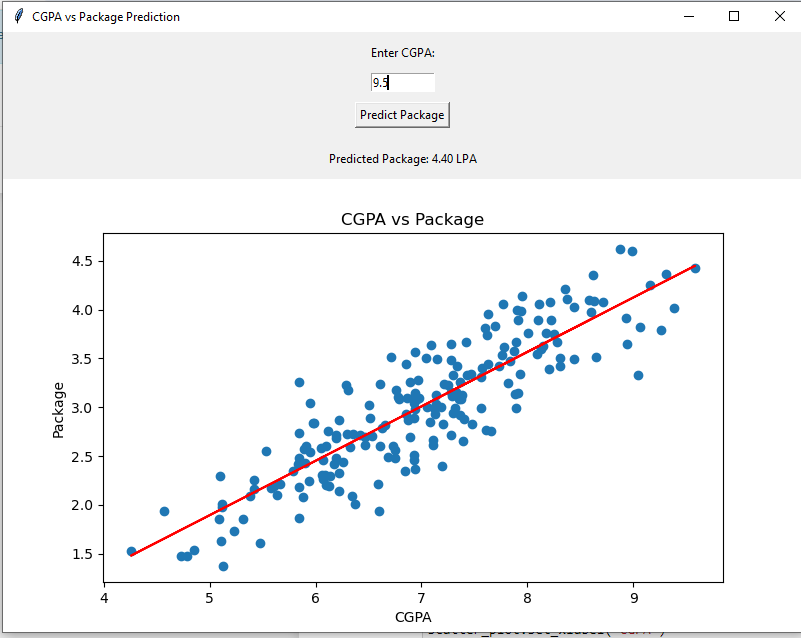
scatter\_plot.set\_title('CGPA vs Package')

canvas = FigureCanvasTkAgg(fig, master=root)

canvas.draw()

canvas.get\_tk\_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=1)

tk.mainloop()



POSTLAB

1.

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np

df = pd.read\_csv('./placement.csv')

df.head()

# plotting the scatter plot

plt.scatter(df['cgpa'],df['package'])

plt.xlabel('CGPA') # setting x label

plt.ylabel('Package(in lpa)') # setting y label

plt.title('By Rohitkumar Pandey')

plt.show()

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

# Getting cgpa values as X and pacakge as Y

X, Y= df.iloc[:,0], df.iloc[:,1]

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(X,Y,test\_size=0.2,random\_state=0)

lr\_model = LinearRegression()

lr\_model.fit(np.array(X\_train).reshape(-1,1),np.array(Y\_train).reshape(-1,1))

Y\_pred = lr\_model.predict(np.array(7).reshape(-1, 1))

print(Y\_pred)

class LinearRegression:

def \_\_init\_\_(self):

self.m = None

self.c = None

def fit(self,x\_train,y\_train):

num, den = 0,0

x\_mean, y\_mean = x\_train.mean(), y\_train.mean()

for i in range(len(x\_train)):

num = num + ((x\_train[i]-x\_mean)\*(y\_train[i]-y\_mean))

den = den + ((x\_train[i]-x\_mean)\*\*2)

m = num/den

c = y\_mean - (m\*x\_mean)

self.m = m

self.c = c

def predict(self,x\_test):

if self.m is None or self.c is None:

print('Model is not trained yet, Call the fit method.')

return None

return self.m\*x\_test + self.c

import tkinter as tk

from matplotlib.figure import Figure

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

root = tk.Tk()

root.title("Package (in LPA) Prediction")

root.geometry('800x400')

fig = Figure(figsize=(8, 6))

scatter\_plot = fig.add\_subplot(111)

def predict\_package():

try:

cgpa = float(entry\_cgpa.get())

predicted\_package = lr\_model.predict([[cgpa]])

label\_prediction.config(text=f"Predicted Package: {predicted\_package[0][0]:.2f} LPA")

except ValueError:

label\_prediction.config(text="Please enter a valid CGPA")

label\_cgpa = tk.Label(root, text="Enter CGPA:")

label\_cgpa.pack(pady=10)

entry\_cgpa = tk.Entry(root, width=10)

entry\_cgpa.pack()

button\_predict = tk.Button(root, text="Predict Package", command=predict\_package)

button\_predict.pack(pady=10)

label\_prediction = tk.Label(root, text="")

label\_prediction.pack(pady=10)

scatter\_plot.scatter(df['cgpa'], df['package'])

scatter\_plot.plot(X\_train, lr\_model.predict(np.array(X\_train).reshape(-1,1)), color='orange')

scatter\_plot.set\_xlabel('CGPA')

scatter\_plot.set\_ylabel('Package')

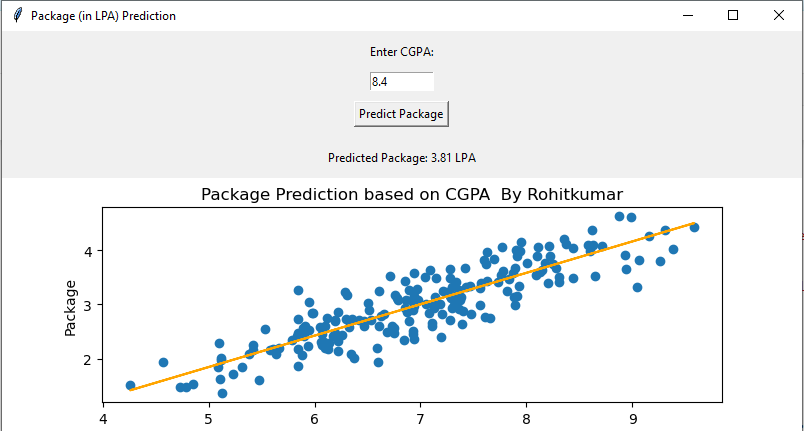
scatter\_plot.set\_title('Package Prediction based on CGPA By Rohitkumar Pandey')

canvas = FigureCanvasTkAgg(fig, master=root)

canvas.draw()

canvas.get\_tk\_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=1)

tk.mainloop()



Postlab 2\\\\;

# Postlab 3

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np

df = pd.read\_csv('./placement.csv')

df.head()

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

# Getting cgpa values as X and pacakge as Y

X, Y= df.iloc[:,0], df.iloc[:,1]

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(X,Y,test\_size=0.2,random\_state=0)

from sklearn.linear\_model import LinearRegression

lr\_model = LinearRegression() # Creating the model

lr\_model.fit(np.array(X\_train).reshape(-1,1),Y\_train) # Fitting the model

# To show the errors in prediction MAE, MSE etc

import tkinter as tk

from matplotlib.figure import Figure

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

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

root = tk.Tk()

root.title("Package (in LPA) Prediction")

root.geometry('800x600')

fig = Figure(figsize=(8, 6))

scatter\_plot = fig.add\_subplot(111)

def predict\_package():

try:

cgpa = float(entry\_cgpa.get())

predicted\_package = lr\_model.predict([[cgpa]])

label\_prediction.config(text=f"Predicted Package: {predicted\_package[0]:.2f} LPA")

show\_errors()

except ValueError:

label\_prediction.config(text="Please enter a valid CGPA")

def show\_errors():

y\_pred = lr\_model.predict(np.array(X\_test).reshape(-1, 1))

# Calculate metrics

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

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

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

rmse = np.sqrt(mse)

error\_text = f"Regression Metrics:- \n"\

f"Mean Squared Error (MSE): {mse:.2f}\n" \

f"Mean Absolute Error (MAE): {mae:.2f}\n" \

f"R-squared (R2): {r2:.2f}\n" \

f"Root Mean Squared Error (RMSE): {rmse:.2f}"

label\_errors.config(text=error\_text)

label\_cgpa = tk.Label(root, text="Enter CGPA:")

label\_cgpa.pack(pady=10)

entry\_cgpa = tk.Entry(root, width=10)

entry\_cgpa.pack()

button\_predict = tk.Button(root, text="Predict Package", command=predict\_package)

button\_predict.pack(pady=10)

label\_prediction = tk.Label(root, text="")

label\_prediction.pack(pady=10)

label\_errors = tk.Label(root, text="")

label\_errors.pack(pady=10)

scatter\_plot.scatter(df['cgpa'], df['package'])

scatter\_plot.plot(X\_train, lr\_model.predict(np.array(X\_train).reshape(-1,1)), color='orange')

scatter\_plot.set\_xlabel('CGPA')

scatter\_plot.set\_ylabel('Package')

scatter\_plot.set\_title('Package Prediction based on CGPA By Rohitkumar Pandey')

canvas = FigureCanvasTkAgg(fig, master=root)

canvas.draw()

canvas.get\_tk\_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=1)

tk.mainloop()

