# -\*- coding: utf-8 -\*-

"""

Created on Sun Jan 19 18:53:10 2020

@author: Sheikh

"""

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np

dataset = pd.read\_csv('monthlyexp vs incom.csv')

X = dataset.iloc[:, :1].values

Y = dataset.iloc[:, 1:2].values

# Splitting the dataset into the Training set and Test set

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

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size = 1/3, random\_state = 0)

# Fitting Simple Linear Regression to the Training set

from sklearn.linear\_model import LinearRegression

regressor = LinearRegression()

regressor.fit(X\_train, Y\_train)

# Predicting the Test set results

Y\_pred = regressor.predict(X\_test)

# Fitting Polynomial Regression to the dataset

from sklearn.preprocessing import PolynomialFeatures

poly\_reg = PolynomialFeatures(degree = 8)

X\_poly = poly\_reg.fit\_transform(X)

poly\_reg.fit(X\_poly, Y)

lin\_reg\_2 = LinearRegression()

lin\_reg\_2.fit(X\_poly, Y)

# Visualising the Training set results

plt.scatter(X\_train, Y\_train, color = 'black')

plt.plot(X\_train, regressor.predict(X\_train), color = 'brown')

plt.title('Mothly experience vs Income')

plt.show()

# Visualising the Test set results

plt.scatter(X\_test, Y\_test, color = 'black')

plt.plot(X\_train, regressor.predict(X\_train), color = 'brown')

plt.title('Mothly experience vs Income')

plt.xlabel('Monthly Experience')

plt.ylabel('Income')

plt.show()

# Visualising the Polynomial Regression results

plt.scatter(X, Y, color = 'black')

plt.plot(X, lin\_reg\_2.predict(poly\_reg.fit\_transform(X)), color = 'brown')

plt.title('Mothly experience vs Income (Polynomial Regression)')

plt.xlabel('Monthly Experience')

plt.ylabel('Income')

plt.show()

################# checking accuracy

from sklearn import metrics

print('Mean Absolute Error:', metrics.mean\_absolute\_error(Y\_test, Y\_pred))

print('Mean Squared Error:', metrics.mean\_squared\_error(Y\_test, Y\_pred))

print('Root Mean Squared Error:', np.sqrt(metrics.mean\_squared\_error(Y\_test, Y\_pred)))