INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

MASTER'S OF COMPUTER SCINENCE

RESEARCH TOPIC: **DESIGN AND IMPLEMENTATION OF PYTHON PROGRAM FOR LEAST-SQUARES REGRESSION ANALYSIS**

MAJOR: **PROGRAMMING WITH PYTHON - DLMDSPWP01**

NAME: **REVAN SAI ANDE**

MATRICULATION: **92206451**

UNIVERSITY: **INTERNATIONAL UNIVERSITY OF APPLIED SCIENCE**

Thesis Advisor: **DR. COSMINA CROITORU**

c.croitoru@iubh-fernstudium.de

Due Date : **APRIL 2023**

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

TABLE OF CONTENT

INTRODUCTION

It is an assignment to program using Python. It generates the best-fitting results for each x and y using the training data set and a few ideal functions.

AIM

The goal of the assignment is to create Python software that selects four out of fifty specified ideal functions as the best match based on training data. Once finished, the programmer may be able to detect each pair of x and y coordinates and determine if they fall into one of four ideal functions. All the data might be shown visually.

OBJECTIVE

The programmer must be capable of creating a SQL database and loading it into a single, five-column spreadsheet on their own. Each of the 50 ideal functions from the CSV file must be loaded into a separate table. After loading the perfect functions into the database, the test data should be supplied line by line from an additional CSV file, and if it compiles, it should be harmonized to one of the four functions specified under I. (subcategory above). The outcomes must then be saved in a separate four-column SQLite database table. After everything has been completed, all of the aforementioned components, y, the training data, the test data, the selected ideal purposes, and the matching / assigned datasets should be clearly depicted.

Literature Review: The existing research on least-squares regression analysis and associated programming ideas will be thoroughly reviewed. Researching the popular Python libraries and tools for regression analysis and data visualization will be part of this.

Program Design: The Python program that will be used to carry out the least-squares regression analysis will have a design made for it. The design will contain an object-oriented program structure, regression algorithm implementation, data loading into SQLite tables, and result visualization.

Program Implementation: Python and necessary libraries such as pandas, SQL alchemy, and Bokeh will be used to implement the program concept. The program will be thoroughly tested to ensure that it performs as intended and achieves the research objectives.

Program Evaluation: The programmer's capacity to do least squares regression analysis, identify ideal functions that minimize the sum of all y-deviations squared, and save the results in a SQLite database will be evaluated. The programmer will also be evaluated based on its adherence to the design and implementation standards, the inclusion of exception handling and unit tests, and the documentation quality.

METHODOLOGY

Python packages utilized in the work include NumPy, Pandas, Matplotlib, Seaborn, SQL Alchemy, and Scikit-learn. Regression analysis on a dataset with two variables (x and y1) is the primary objective of the code. The code also reads and writes information to and from a SQLite database.

The code begins by importing all the required libraries and modules. The information is then read from the train.csv file and placed in a Pandas data frame. The ideal and test.csv. Pandas is also used to read csv files.

The method then displays the train and test datasets using Matplotlib and Seaborn. To display the relationship between x and y1 in the training dataset, a line plot is made. Additionally, a new line plot is made for the test dataset.

The code then generates an SQLite database engine and stores the train data in a brand-new table called "train table." Using the SQLite database, big datasets may be efficiently stored and managed.

Then, the code applies Scikit-Learn's linear regression model to regression analysis on the train data. To train the model, the x and y1 values from the training dataset are transformed into NumPy arrays. A scatter plot of the training data is presented alongside the regression line after the regression coefficients (slope) and intercept have been generated. The code predicts the result for an input test value of x = 8 using the trained regression model.

According to its definition, the regression function uses the trained model to predict the outcome from the test data and then returns it.

The programmer reads and displays the train and test datasets, builds a SQLite database engine to store the data, uses Scikit-learn to perform regression analysis on the training data, plots the training data and regression line, and uses the trained model to predict the results for a test input value

DATASET DESCRIPTION:

In the case of an ideal dataset, it is a cluster of data that has been found from any studies performed professionally. They have several characteristics and behaviors such as size, relevance, quality, representativeness, and accessibility. This is generally used for the decision-making process for a trained model. In this task, four training data sets are given which are mentioned as y1, y2, y3, and y4 as a training function as well as one test dataset which itself consists of two test functions X and Y are provided. Moreover, we have an ideal dataset that has 50 ideal functions to work with.

Data Collection

Data Understanding

Three datasets in csv format are provided for the investigative project. Visual validation of the dataset is performed to check for irregularities and impurities of data, and it is validated that the dataset has no value inconsistencies**.**

Training Dataset (four distinct datasets).

Analyze the provided dataset for behavioral and distinctive patterns.

The dataset comprises an independent variable (X) and four dependent variables (Y1, Y2, Y3, Y4) based on visual inspection.

Train

Dataset collection of 50 (ideal function).

Estimate the line of best fit to the train dataset.

The dataset comprises an independent variable (X) and fifty dependent variables (Y1, Y2, Y3...Y5o) based on visual inspection**.**

Ideal

One data set for testing.

Task: Mapping of test dataset on ideal function within maximum deviation.

The dataset has an independent variable (X) and one dependent variable (Y) based on visual inspection.

Test

Storage of data

The project's datasets are in plain text (CSV format), and the data must be kept in a database.

SQLite is the preferred option due to the following attributes and appropriateness.

a) SQLite is a free and open-source relational database management system.

b) Unlike PostgreSQL, SQLite does not require installation.

b) High performance with only one read/write operation performed at a time

Data access

The SQL Alchemy Python module, a python SQL toolkit, and Object-relational Mapper are used to save and retrieve the dataset from SQLite.

The following are SQL Alchemy properties that are appropriate for project requirements.

a. Stable and high-performance b. SQLite database support

c. SQL queries in Python can be written

d. Python SQL queries for several SQL databases

4.Result of data stored in database Below are the tables to be created in the SQLite database

IDEAL Data Frame

```
In [5]: ideal_df
```

Out[5]:

	x	y1	y2	y3	y4	y5	y6	y7	y8	y9	...	y41	y42	y43	y44	
0	-20.0	-0.912945	0.408082	9.087055	5.408082	-9.087055	0.912945	-0.839071	-0.850919	0.816164	...	-40.456474	40.204040	2.995732	-0.008333	12.
1	-19.9	-0.867644	0.497186	9.132356	5.497186	-9.132356	0.867644	-0.865213	0.168518	0.994372	...	-40.233820	40.048590	2.990720	-0.008340	12.
2	-19.8	-0.813674	0.581322	9.186326	5.581322	-9.186326	0.813674	-0.889191	0.612391	1.162644	...	-40.006836	39.890660	2.985682	-0.008347	12.
3	-19.7	-0.751573	0.659649	9.248426	5.659649	-9.248426	0.751573	-0.910947	-0.994669	1.319299	...	-39.775787	39.729824	2.980619	-0.008354	12.
4	-19.6	-0.681964	0.731386	9.318036	5.731386	-9.318036	0.681964	-0.930426	0.774356	1.462772	...	-39.540980	39.565693	2.975530	-0.008361	12.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
395	19.5	0.605540	0.795815	10.605540	5.795815	-10.605540	-0.605540	-0.947580	-0.117020	1.591630	...	39.302770	-38.602093	2.970414	-0.012422	12.
396	19.6	0.681964	0.731386	10.681964	5.731386	-10.681964	-0.681964	-0.930426	0.774356	1.462772	...	39.540980	-38.834310	2.975530	-0.012438	12.
397	19.7	0.751573	0.659649	10.751574	5.659649	-10.751574	-0.751573	-0.910947	-0.994669	1.319299	...	39.775787	-39.070175	2.980619	-0.012453	12.
398	19.8	0.813674	0.581322	10.813674	5.581322	-10.813674	-0.813674	-0.889191	0.612391	1.162644	...	40.006836	-39.309338	2.985682	-0.012469	12.
399	19.9	0.867644	0.497186	10.867644	5.497186	-10.867644	-0.867644	-0.865213	0.168518	0.994372	...	40.233820	-39.551407	2.990720	-0.012484	12.

400 rows × 51 columns

TRAIN Data Frame

```
[3]: train_df
```

[3]:

	x	y1	y2	y3	y4
0	-20.0	-1.290358	0.971772	-8020.1840	-57.798700
1	-19.9	-0.856480	0.760779	-7900.3633	-57.248300
2	-19.8	-0.476500	1.072470	-7782.3230	-57.198140
3	-19.7	-1.240305	0.400996	-7665.4790	-57.041080
4	-19.6	-0.864219	0.624187	-7549.5490	-57.004307
...	...	...	...	...	...
395	19.5	0.976830	-0.942696	7434.5490	60.119160
396	19.6	0.462573	-0.947579	7548.7197	60.744850
397	19.7	0.915682	-1.033946	7664.7600	61.520294
398	19.8	1.196980	-0.955442	7781.9290	60.991844
399	19.9	0.818385	-0.488116	7900.7310	61.907253

400 rows × 5 columns

TEST Data Frame

```
[4]: test_df
```

t[4]:

	x	y
0	-4.0	1.044171
1	19.6	5240.178700
2	-7.2	2.063293
3	-14.6	-43.106170
4	-17.4	-1.217196
...	...	...
95	10.9	-0.647010
96	-1.9	-2.322036
97	19.5	-0.028896
98	14.8	-1435.503500
99	12.0	36.656208

100 rows × 2 columns

LEAST-SQUARES REGRESSION

The least squares approach can be used to find the best fitting line or curve for a set of data points. It is widely used in machine learning, particularly linear regression, time series analysis, and signal processing.

The goal of this strategy is to determine the line or curve factors that minimize the sum of squared errors between anticipated and actual values of data points. One significant feature of the least squares method is that it gives a straightforward and efficient method for estimating model parameters.

However, it may be insufficiently accurate for more complicated data that is nonlinear or contains outliers. Many expansions and changes to the least squares method have been devised to overcome these constraints, such as robust regression and nonlinear regression, which allow for more flexible modelling and potentially give superior results.

Remember that, while the least squares method is useful in machine learning, it is critical to be aware of its limitations and to apply more advanced techniques when working with more complex data.

Linear Regression R-Squared and Correlation Coefficient Regression Below is the correlation coefficient formula that statistically measures the robustness of the relationship between two variables, ranging from -1 to 1. The correlation between the two variables is negative when r is close to -1. There is no relationship between the two variables when r is close to o. The correlation between the two variables is positive when r is close to 1.

To forecast the behavior of dependent variables, least squares regression is performed.

R-squared linear regression and the correlation coefficient r Regression

The formula for the correlation coefficient, which measures the strength of the association between two variables statistically and runs from -1 to 1, is shown below.

$$\rho(x, y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^{n} (y_i - \bar{y})^2}}$$

The correlation between the two variables is negative when r is near to -1.

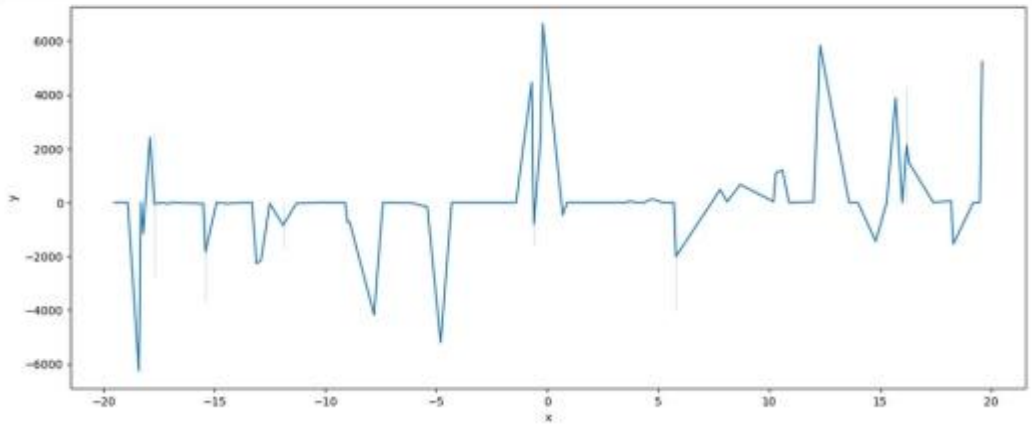
There is no connection between the two variables when r is close to o.

The correlation between two variables is positive when the value of r is close to 1.

Test Line Plot X and Y

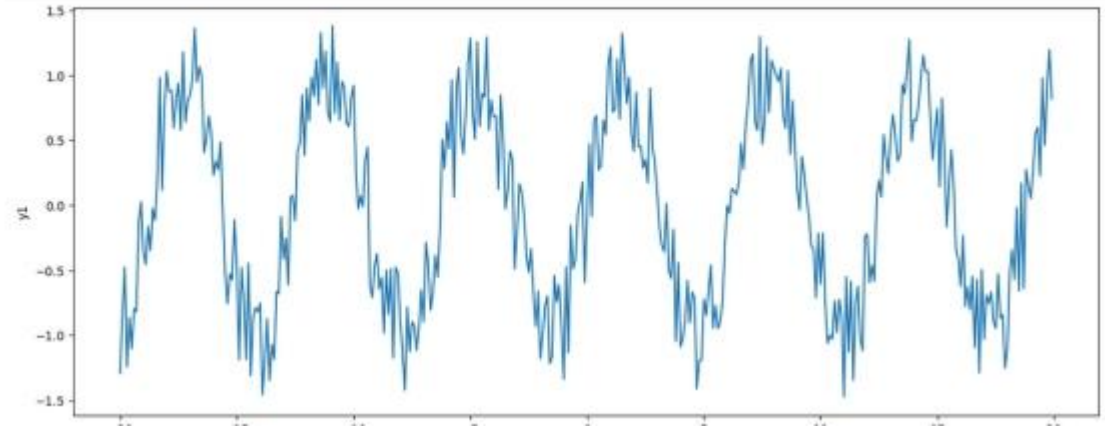
```
In [7]: plt.figure(figsize=(15,6))
        sns.lineplot(x=test_df['x'],y=test_df['y'],data=test_df)

        plt.show()
```



Train Line Plot X and Y

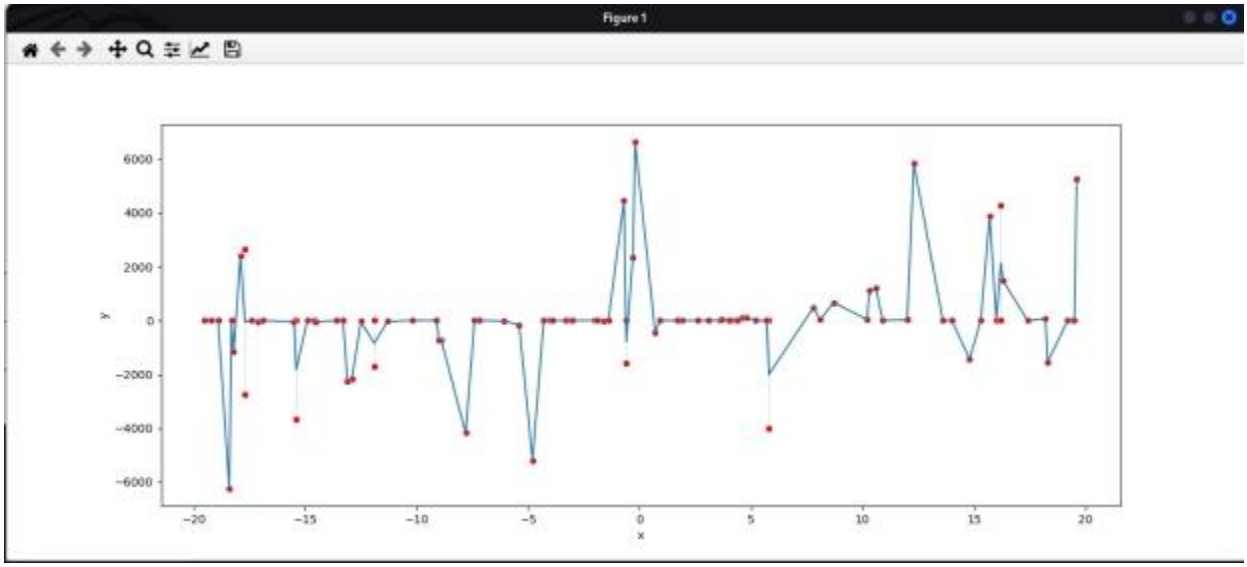
```
In [6]: plt.figure(figsize=(15,6))
        sns.lineplot(x=train_df['x'],y=train_df['y1'],data=train_df)
        plt.show()
```



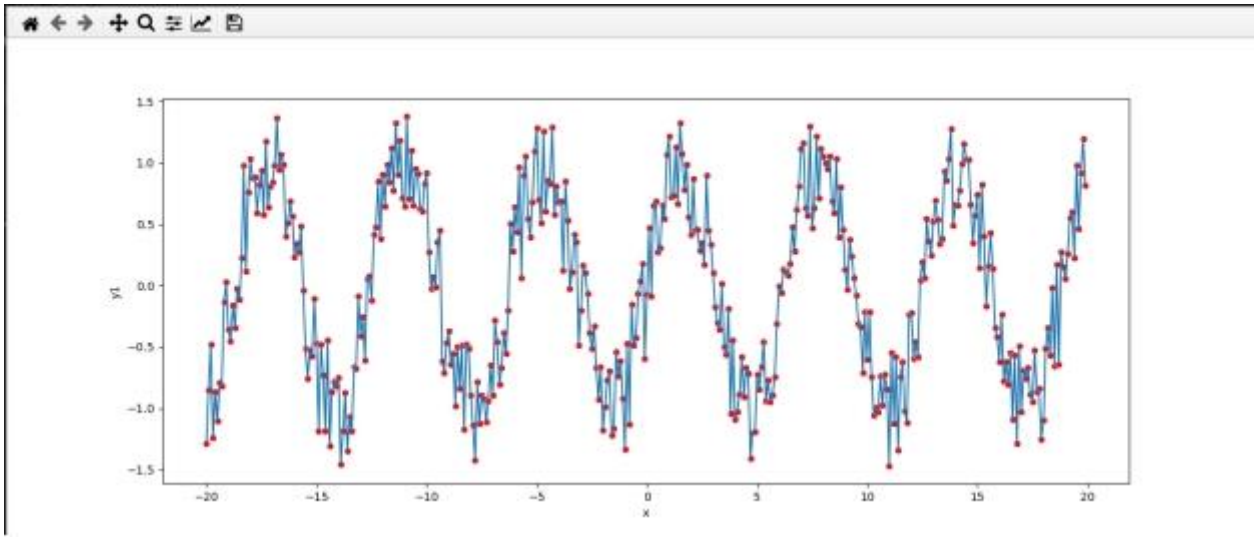
RESULT AND EVALUATION

The task mentioned is performed with the help of programming in python where a two-line plot has been shown for both the test and train datasets as mentioned below.

Test Data frame

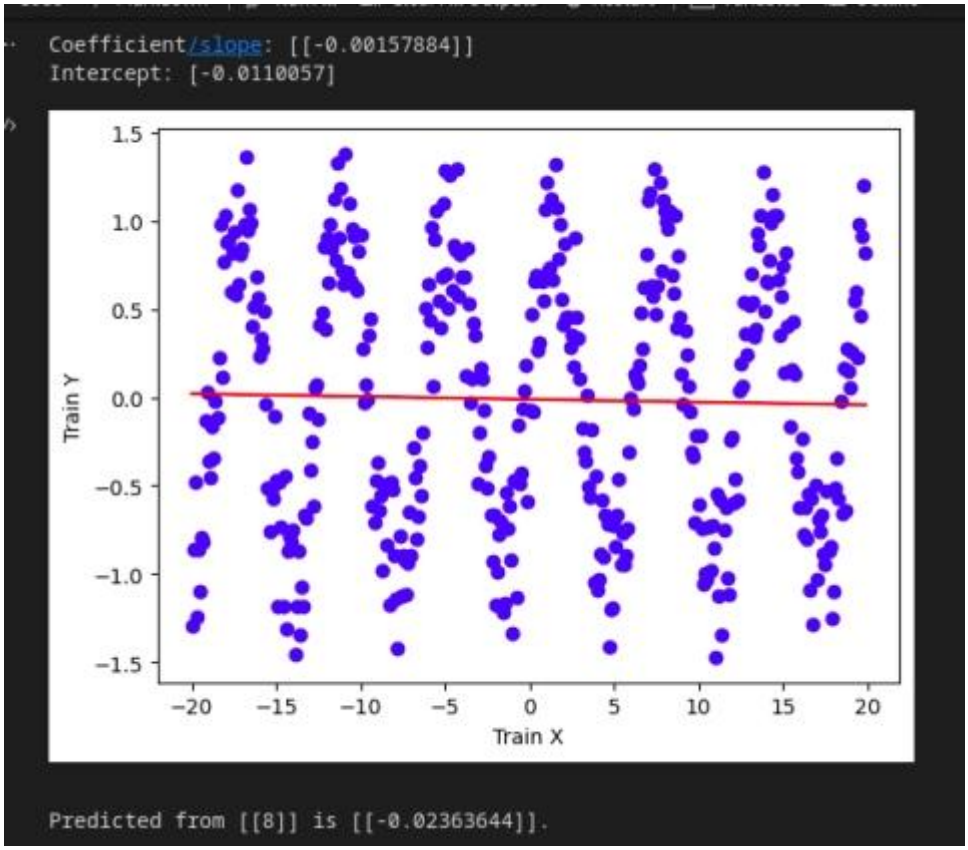


Train Data Frame



After the data from the CSV file is stored in the data frame, linear regression is done. The value of the slope and the intercept is found out that is -0. 00157884 and -0.0110057 respectively. Then the prediction is calculated which comes out to be –0.02363644.

Train x and Train y Data frame with Points



Line plot graph

CONCLUSION

In conclusion, we have used python software to perform the task using training data and testing data with fifty specified ideal functions. The act is accomplished on four variables characterized through the test functions X and Y that are provided. As we perform a linear regression model to analyze the training data set as well as the scikit-learn plot to visualize the relation between train and test datasets.

REFERENCE

Flatt, C., & Jacobs, R. L. (2019). Principle assumptions of regression analysis: Testing, techniques, and statistical reporting of imperfect data sets. Advances in Developing Human Resources, 21(4), 484-5o2.

Link: https://journals.sagepub.com/doi/pdf/1o.1177/1523422319869915

Tran, M. K., Panchal, S., Chauhan, V., Brahmbhatt, N., Meva Walla, A., Fraser, R., & Fowler, M. (2022). Python-based scikit-learn machine learning models for thermal and electrical performance prediction of high-capacity lithium-ion battery. International Journal of Energy Research, 46(2), 786-794.

Link: https://onlinelibrary.wiley.com/doi/abs/1o.1oo2/er.72o2

GITHUB

Rande90/DLMDSPWP01: Design and Implementation of Python Program for Least-Squares Regression Analysis (github.com)

DLMDSPWP01-main.zip

CODING:

Reading data present the CSV file and visualization it has graph using Matplot library.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

train_df=pd.read_csv('train.csv')
test_df=pd.read_csv('test.csv')
ideal_df=pd.read_csv('ideal.csv')

train_df

test_df

ideal_df

plt.figure(figsize=(15,6))
sns.lineplot(x=train_df['x'],y=train_df['y1'],data=train_df,palet
te='hls')
sns.scatterplot(x=train_df['x'],y=train_df['y1'],data=train_df,pa
lette='hls',color="red")
plt.show()

plt.figure(figsize=(15,6))
sns.lineplot(x=test_df['x'],y=test_df['y'],data=test_df,palette='
hls')
sns.scatterplot(x=test_df['x'],y=test_df['y'],data=test_df,palett
e='hls',color="red")
plt.show()
```

Creating Database Engine and pushing the data present in the CSV file into the database and visualization the data by plotting the data points in the graph.

```
import pandas as pd
import sqlalchemy as db
from sqlalchemy import create_engine
import sqlite3 as sql
from sklearn import linear_model
import os
import matplotlib.pyplot as plt
import numpy as np

def regression(test_data, reg):
    print("Predicted from {} is {}.".format(test_data,
reg.predict(test_data)))

def main():
    #Error handling when creating the engine to SQLite
    try:
        engine = create_engine('sqlite:///test.db', echo=False)
    except:
        print("Failed to create engine.")

    #Read data from csv file and store it to dataframe
    train_df = pd.read_csv(os.path.join(os.getcwd(), "train.csv"))
    #Create new table in SQLite based on dataframe
    train_df.to_sql('train_table',con=engine, index=False,
if_exists='replace')

    #regress to train the data
    train_x = np.asanyarray(train_df[['x']])
    train_y = np.asanyarray(train_df[['y1']])
    global reg
```

```
    reg = linear_model.LinearRegression()
    reg.fit(train_x, train_y)

    coef = reg.coef_
    intercept = reg.intercept_

    print('Coefficient/slope: {}'.format(coef))
    print('Intercept: {}'.format(intercept))

    #plotting the training data with regression result
    #plt.scatter(train_df.iloc[:,0], train_df.iloc[:,1],
color='blue')
    #plt.plot(train_x, coef * train_x + intercept, color='red')
    plt.plot(train_df.iloc[:,0], train_df.iloc[:,1], 'o',
color='blue', markersize=8, label='Data points')
    plt.plot(train_x, coef * train_x + intercept, '-',
color='red', label='Regression line')
    plt.xlabel('Train X')
    plt.ylabel('Train Y')
    plt.show()

    #regression to predict - best candidate for the function
    test_data = [[8]]
    regression(test_data,reg)

if __name__ == "__main__":
    main()
```