CHAPTER 5

DESIGN

5.1 DESIGN GOALS

- To predict the MSE of tesla stock using linear regression algorithm.
- we will use number of python and machine learning libraries in this project.
- This method gives a faster and more accurate output when compared to the other techniques.
- For our experiment we have used the python libraries like matplotlib and SKlearn for implementing and training the machine learning model.
- We will get a final prediction score that tells us the accuracy.
- Linear regression machine learning algorithm is used for predictions and gives accurate result.
- Matplotlib is also used for plotting graphs.

5.2 ALOGORITHM

1. Import all the required libraries.
2. Chose an csv file which has a dataset.
3. Load the tesla stock price dataset.

4. Read the csv file.

5. Remove all the unwanted columns.

6. Print the number of rows.

7. Splitting the info into training and test sets.

8. The ratio for training and test sets is taken as 75 and 25.

9. 75 for training set.

10. 25 for test set.

11. Creating the regressor from sci-kit's rectilinear regressor module.

12. Make prediction and evaluate the results.

13. Plot the predicted and actual values.

CHAPTER 6

IMPLEMENTATION

6.1 Importing the libraries

We load here the necessary Python and machine learning libraries.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
import tkinter as tk
from tkinter.filedialog import askopenfilename
from tkinter import *
from PIL import ImageTk, Image
```

Fig 6.1: - Importing the libraries

6.2 Reading and loading the csv file

Now, we will read and load the csv file.

```
data = pd.read_csv("/Users/shreyas/Desktop/TSLA.csv")
print('Raw data from tesla dataset : ')
print(pd.options.display.max_rows)


# print(data.head())
```

Fig 6.2: - Reading and loading the csv file

6.3 Selecting the datasets and displaying it

```
def import_csv_data():
    global v
    csv_file_path = askopenfilename()
    print(csv_file_path)
    v.set(csv_file_path)
    df = pd.read_csv(csv_file_path)
    print(df.head())
    # data = pd.read_csv("/Users/shreyas/Desktop/TSLA.csv")


root = tk.Tk()
tk.Label(root, text='File Path').grid(row=0, column=0)
v = tk.StringVar()
entry = tk.Entry(root, textvariable=v).grid(row=0, column=1)
tk.Button(root, text='Browse Data Set', command=import_csv_data).grid(row=1, column=0)
tk.Button(root, text='Close', command=root.destroy).grid(row=1, column=1)
root.mainloop()
```

Fig 6.3: - Selecting and displaying the dataset

6.4 displaying the selected dataset.

```
data = pd.read_csv("/Users/shreyas/Desktop/TSLA.csv")
print('Raw data from tesla dataset : ')
print(pd.options.display.max_rows)


# print(data.head())
```

Fig 6.4: - displaying the selected dataset

6.5 printing the number of rows

```
data = pd.read_csv("/Users/shreyas/Desktop/TSLA.csv")
print('Raw data from tesla dataset : ')
print(pd.options.display.max_rows)


# print(data.head())
```

Fig 6.5: - printing the number of rows

6.6 Removing the unwanted columns

```
root.mainloop()

data = data.drop('Date', axis=1)
data = data.drop('Adj Close', axis=1)
print('\n\nData after removing Date and Adj Close : ')
print(data.head())
```

Fig 6.6: - Removing the unwanted columns

6.7 Shuffle and Split Data

we will presently divide the information into preparing and test sets. 75% of the data will be used for training and 25% for testing.

```
print(data.head())

data_X = data.loc[:, data.columns != 'Close']
data_Y = data['Close']
train_X, test_X, train_y, test_y = train_test_split(data_X, data_Y, test_size=0.25)
print('\n\nTraining Set')
print(train_X.head())
print(train_y.head())
```

Fig 6.7: - Shuffle and Split Data

6.8 Creating the regressor from sci-kit's rectilinear regressor module

```
# calling a variable regressor
regressor = LinearRegression()
regressor.fit(train_X, train_y)

predict_y = regressor.predict(test_X)
score = regressor.score(test_X, test_y)
print('Prediction Score : ', score)
mse = mean_squared_error(test_y, predict_y)
print('Mean Squared Error : ', mse)
```

Fig 6.8: - Creating the regressor from sci-kit's rectilinear regressor module

6.9 Make predictions and evaluate the results

To appropriately assess the exhibition of each model you've picked, we genuinely should make a preparation and foreseeing pipeline that permits us to rapidly and successfully train models utilizing different sizes of preparing information and perform expectations on the testing information.

```
predict_y = regressor.predict(test_X)
score = regressor.score(test_X, test_y)
print('Prediction Score : ', score)
mse = mean_squared_error(test_y, predict_y)
print('Mean Squared Error : ', mse)

s2 = "Prediction Score:" + str(score)
m2 = "MSE:" + str(mse)

root2 = Tk()
```

Fig 6.9: - make predictions and evaluate the results

6.10 Plotting the graphs

```
def graph():
    fig = plt.figure()
    ax = plt.axes()
    ax.grid()
    ax.set(xlabel='Close ($)', ylabel='Open ($)', title='Tesla Stock Prediction using Linear Regression')
    ax.plot(test_X['Open'], test_y)
    ax.plot(test_X['Open'], predict_y)
    fig.savefig('LRPlot.png')
    plt.show()


my_button = Button(root, text="PRESS", padx=30, command=graph, fg="red")
root.configure(bg='blue')
my_button.pack()

root.mainloop()
```

Fig 6.11: - Plotting the graphs

6.12 Buttons to plot the graph using tkinter

```
def click():
    myllabel = Label(root2, text=s2, fg="blue", bg="yellow")
    myllabel.pack()
    myllabel2 = Label(root2, text=m2, fg="red", bg="yellow")
    myllabel2.pack()


myButton = Button(root2, text="values", command=click)
myButton.pack()
root2.mainloop()

root = Tk()
root.title('tesla stock prediction')
root.geometry("400x200")
```

Fig 6.13: - Buttons to plot graph

CHAPTER 7

RESULTS

SCREENSHOT 7.1

Select the dataset with which you want to go ahead with

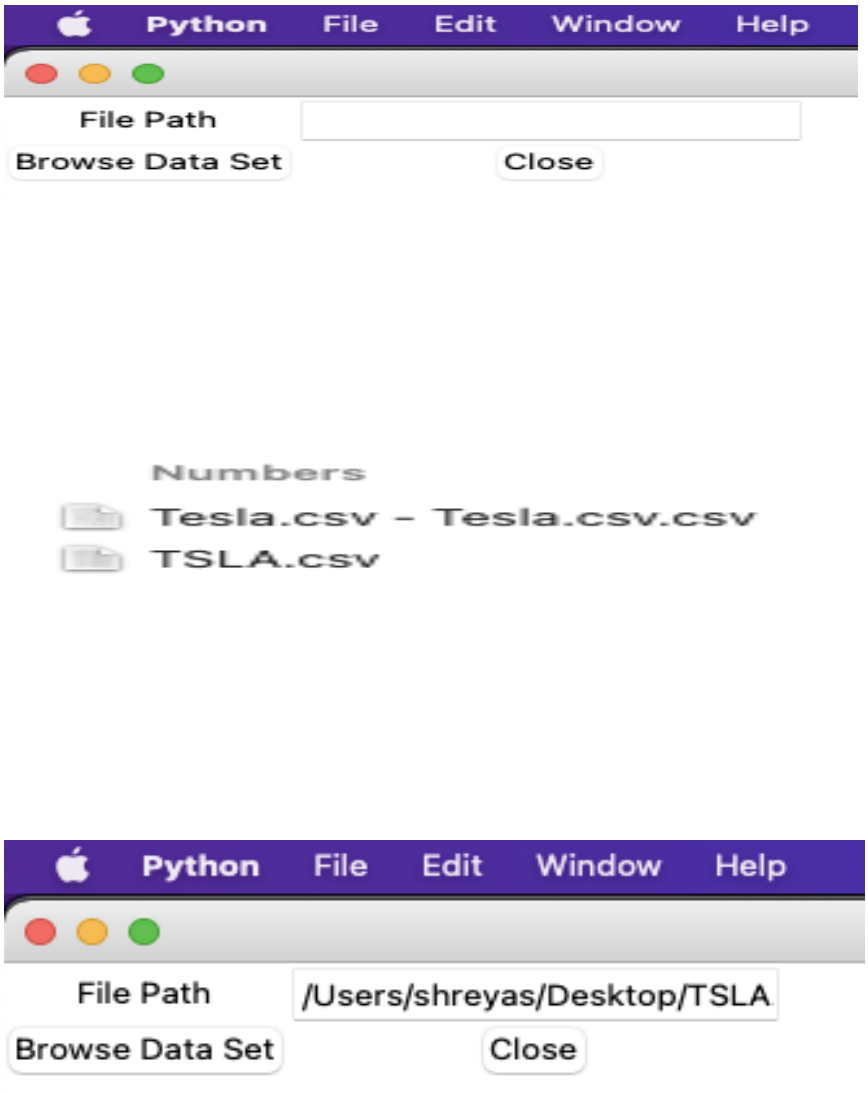

Fig 7.1: - Select the dataset

SCREENSHOT 7.2

```
/Users/shreyas/Desktop/TSLA.csv
         Date       Open   High        Low      Close  Adj Close    Volume
0  2010-06-29  19.000000  25.00  17.540001  23.889999  23.889999  18766300
1  2010-06-30  25.790001  30.42  23.299999  23.830000  23.830000  17187100
2  2010-07-01  25.000000  25.92  20.270000  21.959999  21.959999   8218800
3  2010-07-02  23.000000  23.10  18.709999  19.200001  19.200001   5139800
4  2010-07-06  20.000000  20.00  15.830000  16.110001  16.110001   6866900
```

Fig 7.2: - load and display the database.

SCREENSHOT 7.3

```
Raw data from tesla dataset :
60
/Users/shreyas/Desktop/TSLA.csv
```

Fig 7.3: - printing the file path link and number of rows

SCREENSHOT 7.4

Data after removing Date and Adj Close:

```
Data after removing Date and Adj Close :
         Open   High        Low      Close    Volume
0   19.000000  25.00  17.540001  23.889999  18766300
1   25.790001  30.42  23.299999  23.830000  17187100
2   25.000000  25.92  20.270000  21.959999   8218800
3   23.000000  23.10  18.709999  19.200001   5139800
4   20.000000  20.00  15.830000  16.110001   6866900
```

Fig 7.4: - Data after removing Date and Adj Close:

SCREENSHOT 7.5

Splitting the data into training and test sets.

```
Training Set
              Open         High          Low    Volume
1813   351.350006   363.709991   350.000000   7667100
434     35.259998    35.320000    34.540001   1015600
2391   435.000000   435.309998   426.109985   9945700
553     28.000000    28.900000    27.900000    841700
474     30.260000    30.959999    29.219999   1585700
1813     363.690002
434       34.980000
2391     430.380005
553       28.549999
474       29.430000
Name: Close, dtype: float64
Prediction Score :  0.9996787095828739
Mean Squared Error :  4.419021158826755
```

Fig 7.5: - Spitting the data into training and test set.

SCREENSHOT 7.6

Displaying the MSE AND Prediction score.

```
Prediction Score :  0.9996787095828739
Mean Squared Error :  4.419021158826755
```

Fig 7.6: - Displaying the MSE AND Prediction.

SCREENSHOT 7.7

Displaying the MSE and Prediction score using tkinter.

Fig 7.7: - Displaying the MSE and Prediction score using tkinter.

SCREENSHOT 7.8

Pressing the button created with tkinter to display the graph

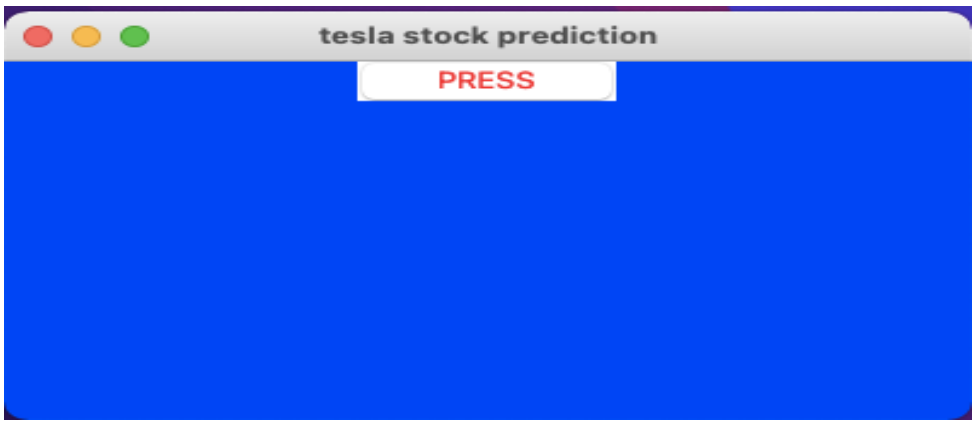

Fig 7.8: - Pressing the button created with tkinter to display the graph

SCREENSHOT 7.9

Displaying the graph.

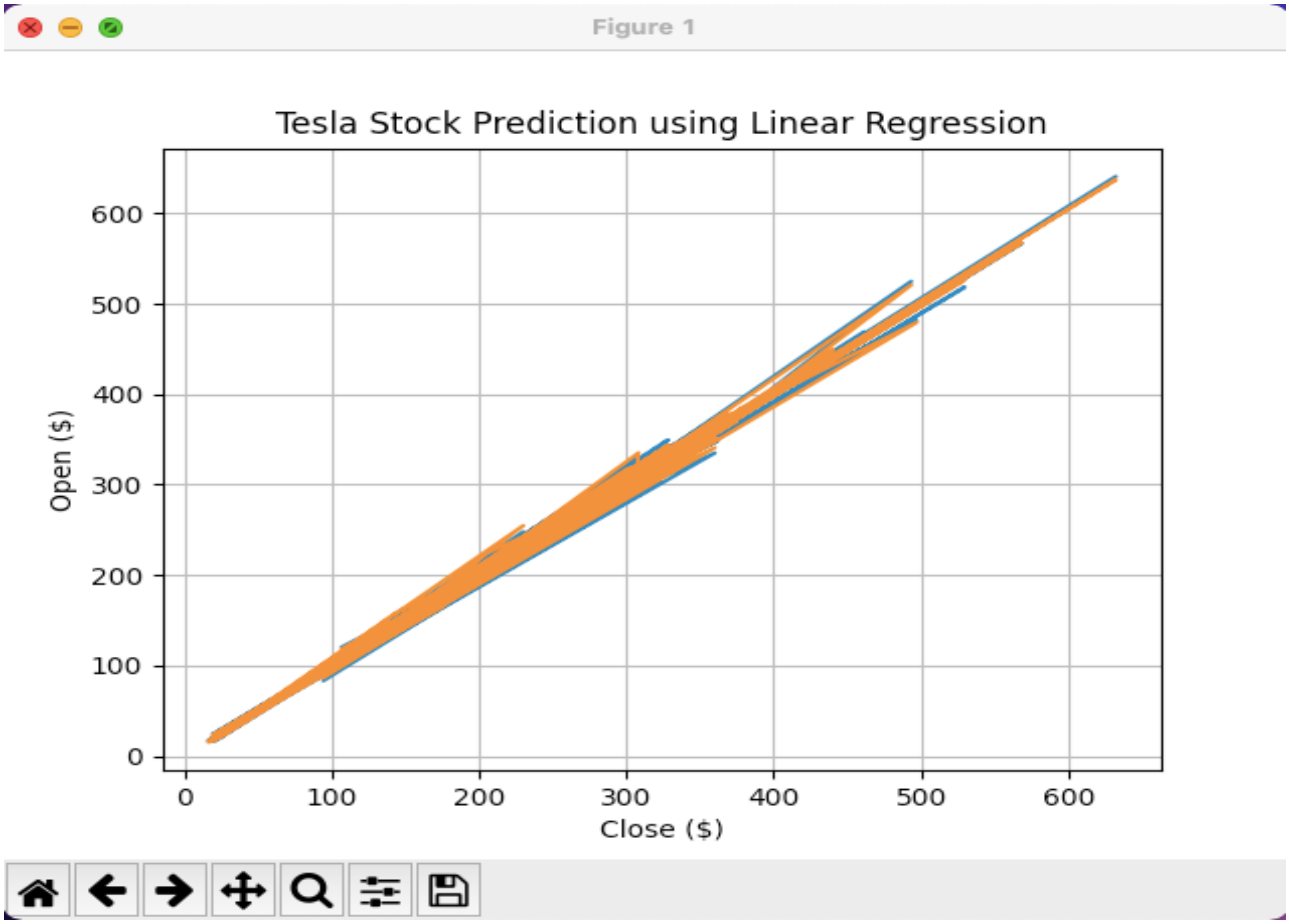

Fig 7.9: - Graph showing the open vs close price of the stock.

SCREENSHOT 7.10

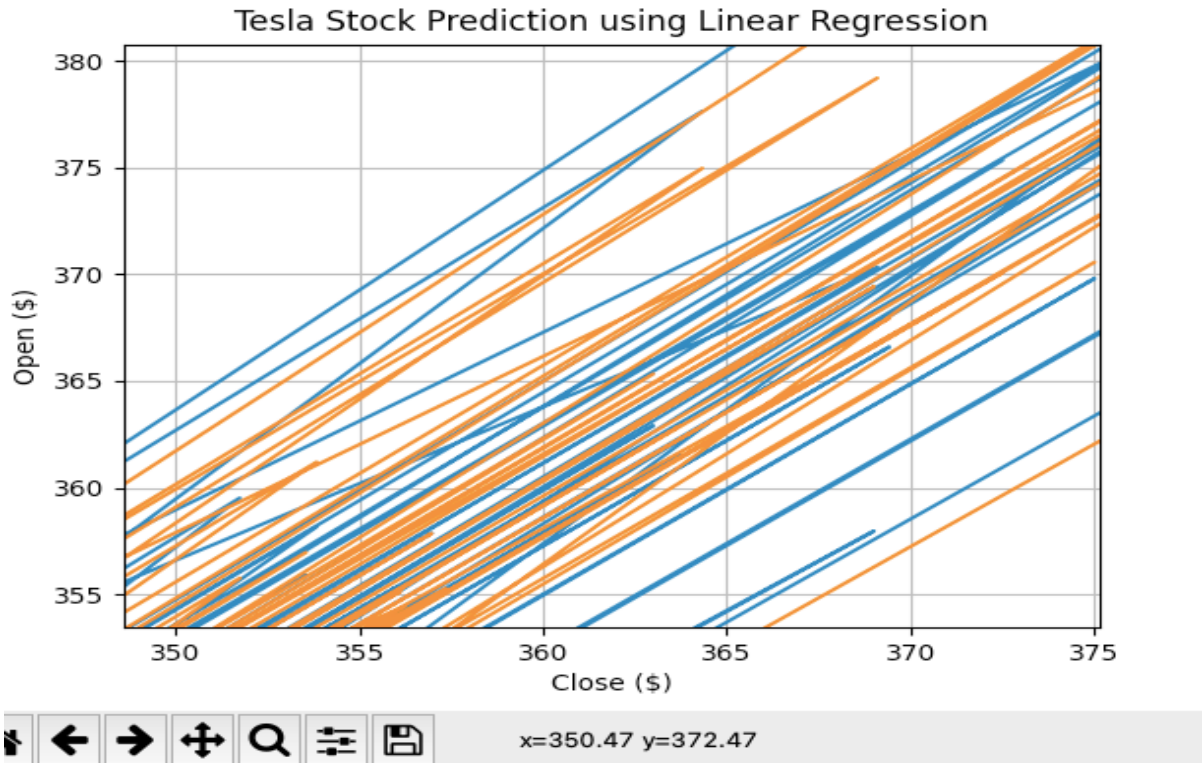

Fig 7.10: - open price on y-axis and close price on x-axis

SCREENSHOT 7.11

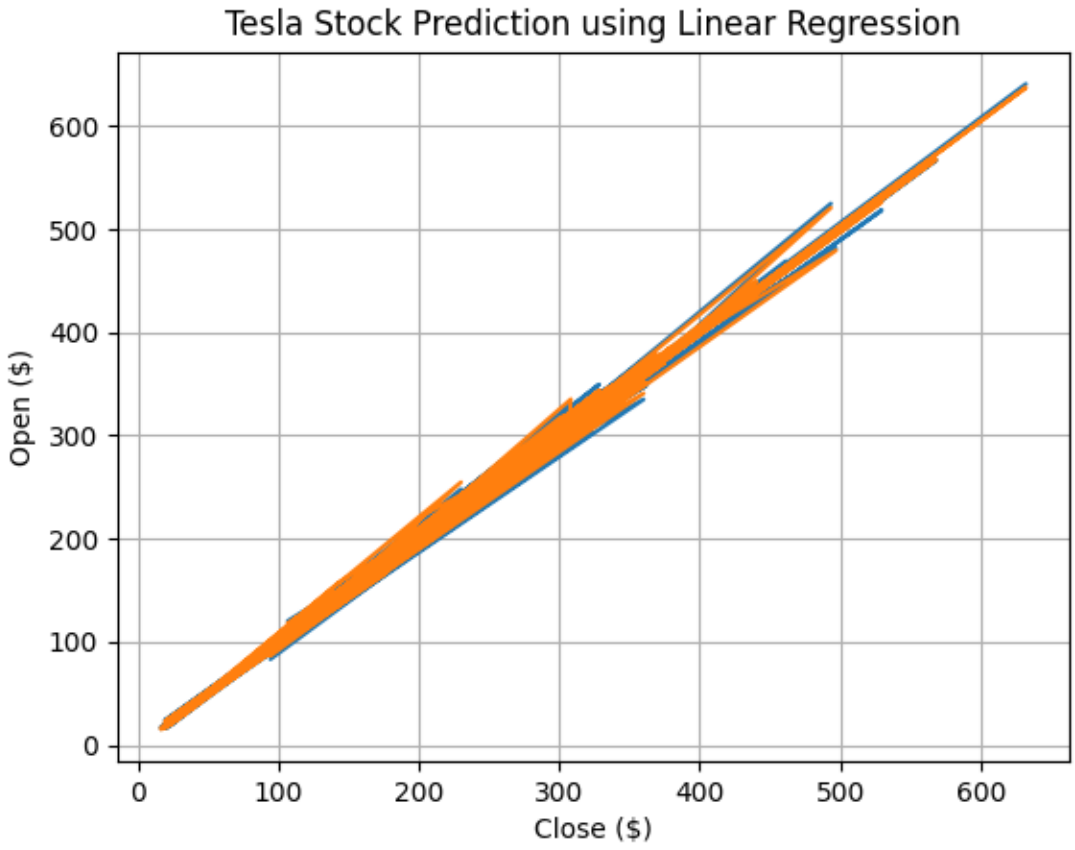

Fig 7.11: - Plotting the graph.