Simple stock predictor using python

By- kacham om sai Rusheel
17311A1939

CONTENT:-

Introduction:-

- Financial markets are highly volatile and generate huge amounts of data daily
- It is the most popular financial market instrument and its value changes quickly
- Stock prices are predicted to determine the future value of companies' stock or other financial instruments that are marketed on financial exchanges
- However, the stock market is influenced by many factors such as political events, economic conditions and traders' expectation

General steps taken to build a prediction system :-

- There are main 4 steps to predict any future output using python .
- It is important to follow them in a correct order.

1. Installing dependencies.
2. Different ways to collect data.
3. Main script.
4. Selecting a training model and training the model.

Steps taken to predict Stock prices:-

- By following the general steps to predict and train a model we will be using Google colab which is a cloud based data science work space similar to the jupyter notebook. Each colab session is equipped with a virtual machine running 13 GB of ram and either a CPU, GPU, or TPU processor.

- In which we will importing **CVS** , **NUMPY** and **SCIKIT LEARN.**

- Data for this will be collected from the Colab library for Facebook .

- The model will be written in python.

- The training model used is SVR (support vector regression) and Leaner Regression to check which model gives the accurate prediction.

Installing Dependencies:-

- **INSTALL CVS** (HELPS TO READ THE DATA).
- **INSTALL NUMPY** (IT IS USED TO RUN THE DATA AND MAKE CALCULATIONS ON THE DATA).
- **INSTALL SCIKIT-LEARN** (IT WILL HELP US TO BUILD THE PREDICTIVE MODEL).

Collecting data sets:-

- Data sets can be downloaded using google , which are available for free .
- In real life workspace's Data bases are used to retrieve the required data.
- Softwares like Sql are used.

Main script:-

- The main program will be written in python and will be using python libraries , quandl libraries, numpy libraries, scikit librabries and few Colab libraries.

Selecting a training model and training it:-

- The model uses in this stock predictor will be SVR (support vector regression) and Linear Regression.The reason for using two models is the accuracy for both the models depends upon the given data . So that we can predict the data more accurately. Accuracy is measured by 1 .

- **SVR**: SVR uses the same basic idea as Support Vector Machine (SVM), a classification algorithm, but applies it to predict real values rather than a class. SVR acknowledges the presence of non-linearity in the data and provides a proficient prediction model.

- **LINEAR REGRESSION**:linear regression is useful for finding relationship between two continuous variables.

Source code:-

1. Installing dependencies.

colab.research.google.com

Welcome To Colaboratory - Colaboratory | simplestockpredictor.ipynb - Colaboratory | Python/stock.ipynb at master · randerson112358/Python · GitHub

simplestockpredictor.ipynb

File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

RAM Disk Editing

```
pip install quandl
```

```
[4] import quandl
    import numpy as np
    from sklearn.linear_model import LinearRegression
    from sklearn.svm import SVR
    from sklearn.model_selection import train_test_split
```

2.Collecting the data.

colab.research.google.com

Welcome To Colaboratory - Colaboratory | simplestockpredictor.ipynb - Colaboratory | Python/stock.ipynb at master · randerson112358/Python · GitHub

simplestockpredictor.ipynb

File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

RAM Disk Editing

```
[5] df = quandl.get("WIKI/FB")
    print(df.head())
```

```
                Open   High    Low  ...  Adj. Low  Adj. Close  Adj. Volume
Date                                ...
2012-05-18  42.05  45.00  38.00  ...     38.00     38.2318  573576400.0
2012-05-21  36.53  36.66  33.00  ...     33.00     34.0300  168192700.0
2012-05-22  32.61  33.59  30.94  ...     30.94     31.0000  101786600.0
2012-05-23  31.37  32.50  31.36  ...     31.36     32.0000   73600000.0
2012-05-24  32.95  33.21  31.77  ...     31.77     33.0300   50237200.0

[5 rows x 12 columns]
```

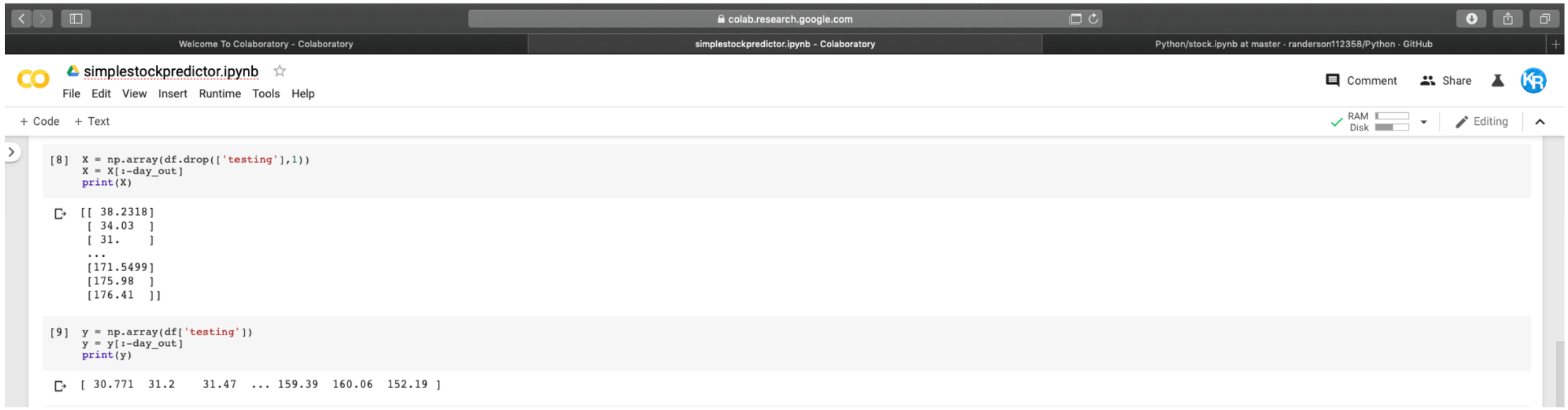
```
[6] df = df[['Adj. Close']]
    print(df.head())
```

```
            Adj. Close
Date
2012-05-18     38.2318
2012-05-21     34.0300
2012-05-22     31.0000
2012-05-23     32.0000
2012-05-24     33.0300
```

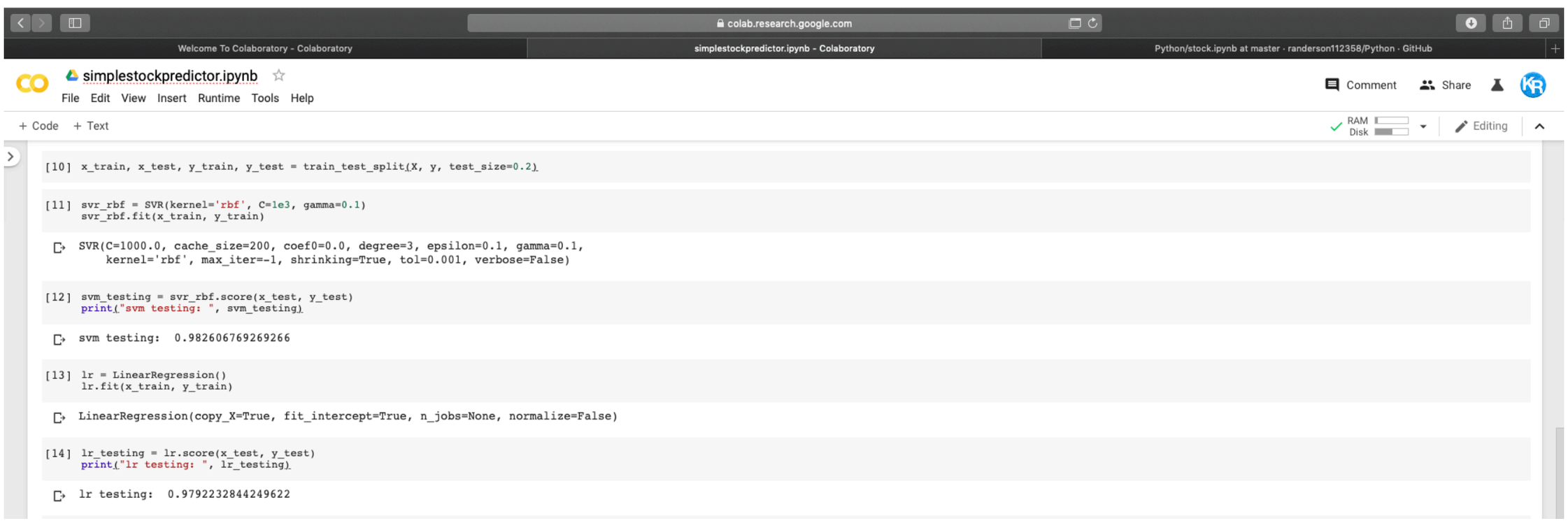
```
[7] day_out = 30
    df['testing'] = df[['Adj. Close']].shift(-day_out)
    print(df.tail())
```

```
            Adj. Close  testing
Date
2018-03-21      169.39      NaN
2018-03-22      164.89      NaN
2018-03-23      159.39      NaN
2018-03-26      160.06      NaN
2018-03-27      152.19      NaN
```

3. Scripting the data.



4.Training the data set.



4.Training the data set.

colab.research.google.com

Welcome To Colaboratory - Colaboratory | simplestockpredictor.ipynb - Colaboratory | Python/stock.ipynb at master · randerson112358/Python · GitHub

simplestockpredictor.ipynb

File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

RAM Disk Editing

```
[17] x_day = np.array(df.drop(['testing'],1))[-day_out:]
     print(x_day)
```

```
[[173.15]
 [179.52]
 [179.96]
 [177.36]
 [176.01]
 [177.91]
 [178.99]
 [183.29]
 [184.93]
 [181.46]
 [178.32]
 [175.94]
 [176.62]
 [180.4 ]
 [179.78]
 [183.71]
 [182.34]
 [185.23]
 [184.76]
 [181.88]
 [184.19]
 [183.86]
 [185.09]
 [172.56]
 [168.15]
 [169.39]
 [164.89]
 [159.39]
 [160.06]
 [152.19]]
```

```
[18] lr_testing = lr.predict(x_day)
     print(lr_testing)
     svm_testing = svr_rbf.predict(x_day)
     print(svm_testing)
```

```
[177.06929331 183.49594578 183.93985898 181.31673552 179.95472911
 181.87162702 182.96123215 187.29947479 188.95406035 185.45319943
 182.28527341 179.88410656 180.57015423 184.38377218 183.75825812
 187.72321011 186.34102583 189.25672844 188.78254843 185.87693476
 188.20747906 187.87454416 189.11548333 176.47404606 172.02482513
 173.27585323 168.73583187 163.18691687 163.8628756  155.92288268]
[174.51026452 181.44748283 181.58000911 175.2998493  175.02634992
 177.21762134 180.87256341 186.12429171 179.47024756 185.18693021
 178.87539467 175.11879301 174.43698522 181.93024749 181.52872163
 183.99898767 187.90316898 179.50239601 179.68082832 186.81862813
 181.53711736 183.19290807 179.42517801 175.34474712 171.18229799
 173.08481147 172.2350428  167.68135199 166.23712457 160.14930674]
```

Conclusion:-

- Thus, as we can see above in our proposed method, we train the data using existing stock dataset that is available. We use this data to predict and forecast the stock price of n-days into the future.
- The average performance of the model decreases with increase in number of days, due to unpredictable changes in trend.