DOCUMENTATION

SUBMITTED BY:-

Name: Ayushi Choudhary (Team Head)

Registration ID: SIRSS2316

College: Maharaja Surajamal Institute Of Technology, (Affiliated to GGSIPU,Delhi)

Name: Shivam Jha (Team Vice Head)

Registration ID: SIRSS2317

College: Maharaja Surajamal Institute Of Technology, (Affiliated to GGSIPU,Delhi)

Name: Aftab Ahmed

Registration ID: SIRSS2310

College: College of Engineering and Technology, Bhubaneswar

Name: K Manoj

Registration ID: SIRSS2309

College: East Point College of Engineering and Technology, Bangalore

Team effort together with precious words of encouragement and guidance makes daunting tasks achievable. It is a pleasure to acknowledge the direct and implied help I have received at various stages in the task of completing the project. I find it impossible to express my thanks to each one of them in words, for it seems too trivial when compare to the profound encouragement that they extended to us.

I hereby perceive this opportunity as a big milestone in my career development. I will use gained skills and knowledge in the best possible way and continue to work on their improvement.

ABSTRACT

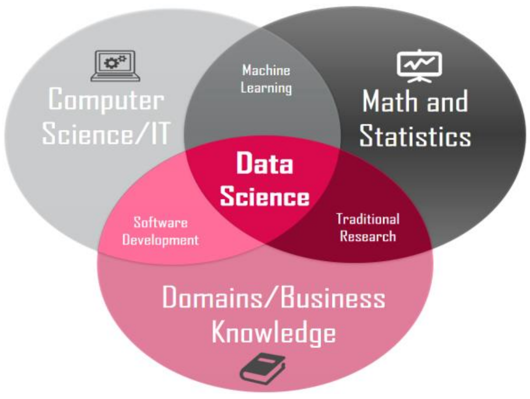
Every company will say they are doing a form of data science, but what exactly does that mean? The field is growing so rapidly, and revolutionizing so many industries, it's difficult to fence in its capabilities with a formal definition, but generally data science is devoted to the extraction of clean information from raw data for the formulation of actionable insights.

Machine learning is the study of computer algorithms that improve automatically through experience. It focuses on the development of computer programs that can access data and use it learn for themselves. Because of this, it facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs. In this project, we experimented with real world data set and explored how machine learning algorithm can be used to find patterns in data. We were expected to gain experiences using common data cleaning, machine learning libraries and were expected to submit a documentation about the data set and algorithm used. After performing the required task on data set up my choice here in lies my final document.

Project is based on applications in the stock price prediction. In this machine learning project, we will be talking about predicting the returns on stocks. We will analyse daily stocks status and predict for the next day

Data Science with Python

Data science is an inter-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data. It is a "concept to unify statistics, data analysis and their related methods" in order to "understand and analyze actual phenomena" with data. It is related to data mining, machine learning and big data.



Python is open source, interpreted, high level language and provides great approach for object-oriented programming. It is one of the best language used by data scientist for various data science projects/application. It provide great functionality to deal with mathematics, statistics and scientific function. It is a programming language with simple syntax that is commonly used for data science. There are a number of python libraries that are used in data science including Pandas, SciPy etc…

Python Fundamentals:-

The basic Python curriculum can be broken down into 4 essential topics that include:

- **Data Types(Int, Float, Strings& Boolean) :**

  Data types are the classification or categorization of data items. Data types represent a kind of value which determines what operations can be performed on that data. Numeric, non-numeric and Boolean (true/false) data are the most used data types.

  Integer: Positive or negative whole numbers (without a fractional part)

  Float: Any real number with a floating point representation in which a fractional component is denoted by a decimal symbol or scientific notation.

  String: A string value is a collection of one or more characters put in single, double or triple quotes.

  Boolean: Data with one of two built-in values True or False. Notice that 'T' and 'F' are capital. true and false are not valid Booleans and Python will throw an error for them.

- **Compound data structures(Lists, Tuples, Sets and Dictionaries) :**

  A compound data type in which the values are made up of components, or elements, that are themselves values. default value.

  List : A list object is an ordered collection of one or more data items, not necessarily of the same type, put in square brackets. It can have similar values in the same list.

  Ex:myset = [10, 20, 30,20,30,”hello”]

Tuple: A Tuple object is an ordered collection of one or more data items, not necessarily of the same type, put in parentheses.

Ex:mytuple = (10, 20, 30,”hello”)

Set : It is a collection which is unordered and unindexed. In Python, sets are written with curly brackets.They also don’t accept to have duplicate values.

Ex:myset = {10, 20, 30,”hello”}

Dictionary : It is an unordered collection of data in a key: value pair form. A collection of such pairs is enclosed in curly brackets.

Ex: {1:"Steve", 2:"Bill", "Ram":3.2 , 4: "Farha"}

An **External library** is something that comes from an outside source. These libraries are a set of useful functions that eliminate the need for writing codes from scratch. A Python library is a reusable chunk of code that you may want to include in your programs/ projects. Compared to languages like C++ or C, a Python libraries do not pertain to any specific context in Python. Here, a 'library' loosely describes a collection of core modules.

Python has a huge community providing support for its libraries and keeping them updated.

There are over 137,000 python libraries present today. Python libraries play a vital role in developing machine learning, data science, data visualization, image and data manipulation applications and more.

Ex: NumPy, Pandas, Matplotlib, Seaborn, SciPy etc.

Libraries used in the project

pandas

Data processing is important part of analyzing the data, because data is not always available in desired format. Various processing are required before analyzing the data such as cleaning, restructuring or merging etc. Numpy, Scipy, Cython and Panda are the tools available in python which can be used fast processing of the data. Further, Pandas are built on the top of Numpy.

Pandas provides rich set of functions to process various types of data. Further, working with Panda is fast, easy and more expressive than other tools. Pandas provides fast data processing as Numpy along with flexible data manipulation techniques as spreadsheets and relational databases. Lastly, pandas integrates well with matplotlib library, which makes it very handy tool for analyzing the data.

pandas_datareader

Via this module, users can download various economics and financial data via Yahoo! Finance, Google Finance, **Federal Reserve Economics Data** (**FRED**), and Fama-French factors. pandas_datareader provides a consistent simple API for us to collect data from these platforms

Numpy

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

At the core of the NumPy package, is the *ndarray* object. This encapsulates *n*-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance.

Matplotlib

Matplotlib is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. Matplotlib is written in Python and makes use of NumPy, the numerical mathematics extension of Python. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as PyQt, WxPythonotTkinter. It can be used in Python and IPython shells, Jupyter notebook and web application servers also.

Matplotlib has a procedural interface named the Pylab, which is designed to resemble MATLAB, a proprietary programming language developed by MathWorks. Matplotlib along with NumPy can be considered as the open source equivalent of MATLAB.

Tensorflow

TensorFlow is an open-source end-to-end platform for creating Machine Learning applications. It is a symbolic math library that uses dataflow and differentiable programming to perform various tasks focused on training and inference of deep neural networks. It allows developers to create machine learning applications using various tools, libraries, and community resources.
Currently, the most famous deep learning library in the world is Google's TensorFlow. Google product uses machine learning in all of its products to improve the search engine, translation, image captioning or recommendations.

How TensorFlow Works

TensorFlow enables you to build dataflow graphs and structures to define how data moves through a graph by taking inputs as a multi-dimensional array called Tensor. It allows you to construct a flowchart of operations that can be performed on these inputs, which goes at one end and comes at the other end as output.

TensorFlow Architecture

Tensorflow architecture works in three parts:

- Preprocessing the data
- Build the model
- Train and estimate the model

It is called Tensorflow because it takes input as a multi-dimensional array, also known as **tensors**. You can construct a sort of **flowchart** of operations (called a Graph) that you want to perform on that input. The input goes in at one end, and then it flows through this system of multiple operations and comes out the other end as output.

This is why it is called TensorFlow because the tensor goes in it flows through a list of operations, and then it comes out the other side.

Keras

Keras is an Open Source Neural Network library written in Python that runs on top of Theano or Tensorflow. It is designed to be modular, fast and easy to use. It was developed

by François Chollet, a Google engineer. Keras doesn’t handle low-level computation. Instead, it uses another library to do it, called the “Backend”.
Keras is high-level API wrapper for the low-level API, capable of running on top of TensorFlow, CNTK, or Theano. Keras High-Level API handles the way we make models, defining layers, or set up multiple input-output models. In this level, Keras also compiles our model with loss and optimizer functions, training process with fit function. Keras in Python doesn’t handle Low-Level API such as making the computational graph, making tensors or other variables because it has been handled by the “backend” engine.

Scikit-learn

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon **NumPy, SciPy** and **Matplotlib**.

Features of Scikit-learn

Rather than focusing on loading, manipulating and summarising data, Scikit-learn library is focused on modeling the data. Some of the most popular groups of models provided by Sklearn are as follows −

Supervised Learning algorithms − Almost all the popular supervised learning algorithms, like Linear Regression, Support Vector Machine (SVM), Decision Tree etc., are the part of scikit-learn.

Unsupervised Learning algorithms − On the other hand, it also has all the popular unsupervised learning algorithms from clustering, factor analysis, PCA (Principal Component Analysis) to unsupervised neural networks.

Clustering − This model is used for grouping unlabeled data.

Cross Validation − It is used to check the accuracy of supervised models on unseen data.

Dimensionality Reduction − It is used for reducing the number of attributes in data which can be further used for summarisation, visualisation and feature selection.

Ensemble methods − As name suggest, it is used for combining the predictions of multiple supervised models.

Feature extraction − It is used to extract the features from data to define the attributes in image and text data.

Feature selection − It is used to identify useful attributes to create supervised models.

Open Source − It is open source library and also commercially usable under BSD license.

What is a machine learning algorithm?

A machine learning algorithm is an algorithm that is able to learn from data. But what do we mean by learning? Mitchell (1997) provides a succinct definition: “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

What is Natural Language Processing?

Natural language processing (NLP) is the use of human languages, such as English or French, by a computer. Computer programs typically read and emit specialized languages designed to allow efficient and unambiguous parsing by simple programs. More naturally occurring languages are often ambiguous and defy formal description. Natural language processing includes applications such as machine translation, in which the learner must read a sentence in one human language and emit an equivalent sentence in another human language. Many NLP applications are based on language models that define a probability distribution over sequences of words, characters, or bytes in a natural language.

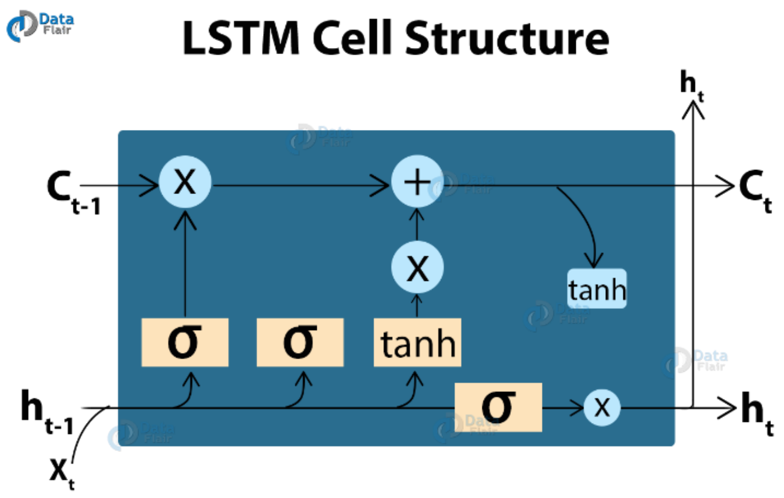
RNN

Deep feedforward networks, also called feedforward neural networks, or multilayer perceptrons (MLPs), are the quintessential deep learning models. The goal of a feedforward network is to approximate some function f^*. For example, for a classifier, $y = f^*(x)$ maps an input x to a category y. A feedforward network defines a mapping $y = f(x; \boldsymbol{\theta})$ and learns the value of the parameters $\boldsymbol{\theta}$ that result in the best function approximation.

LSTM

LSTM stands for **Long short term memory**, and they are a type of RNN (**recurrent neural network**) which is well suited for sequence prediction problems. Based on the previous text, we can predict what the next word will be. It has proven itself effective from the traditional RNN by overcoming the limitations of RNN which had short term memory. LSTM can carry out relevant information throughout the processing of inputs and with a forget gate, it discards non-relevant information.

The following image shows what an LSTM cell looks like –



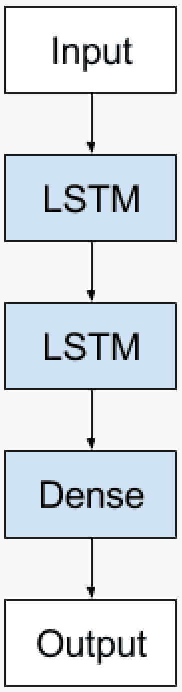
LSTMs are widely used for sequence prediction problems and have proven to be extremely effective. The reason they work so well is that LSTM can store past important information and forget the information that is not.
LSTM has three gates:

- The input gate: The input gate adds information to the cell state,
- The forget gate: It removes the information that is no longer required by the model,
- The output gate: Output Gate at LSTM selects the information to be shown as output

While Implementing any LSTM, we should always reshape our X train in 3-D, add 1 the reason behind is the time step and the 1 is given to the LSTM.

Stacked LSTMs

Stacked LSTMs are now a stable technique for challenging sequence prediction problems. A Stacked LSTM architecture can be defined as an LSTM model comprised of multiple LSTM layers. An LSTM layer above provides a sequence output rather than a single value output to the LSTM layer below. Specifically, one output per input time step, rather than one output time step for all input time steps.



Stacked Long Short-Term Memory Archiecture

Stock Prediction

Trying to predict how the securities exchange will work is one of the most difficult tasks. There are so many variables involved with the expectation – physical elements versus psychological factors, rational and irrational behaviour, and so on.
All of these factors combine to make share costs unpredictable and difficult to predict with any degree of certainty.
Is it possible to use AI to our advantage in this space?AI approaches will potentially reveal examples and insights we hadn't seen before, and these can be used to make unerringly exact expectations, using features like the most recent declarations of an organization, their quarterly income figures, and so on.
We will work with published information regarding a freely recorded organization's stock costs in this report.
We'll use a combination of AI calculations to forecast this company's future stock price with LSTM.

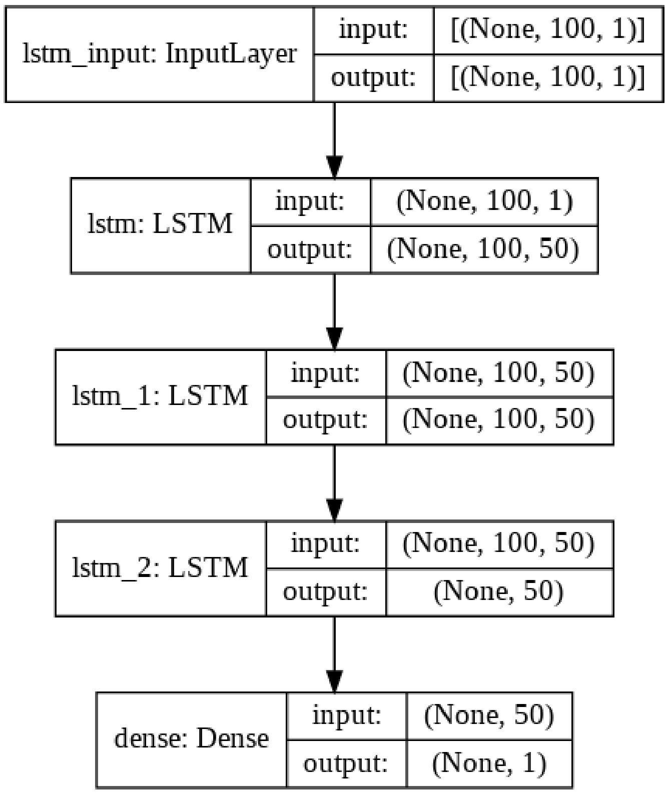
The model we are used is based on microsoft (MSFT) stock prediction now this is an RNN model that we saw a couple of places floating around so we are not sure who originally did it but it's a great way to understand how we can use RNN and LSTM to do stock prediction.

Steps involved in making, training and testing our stock prediction model

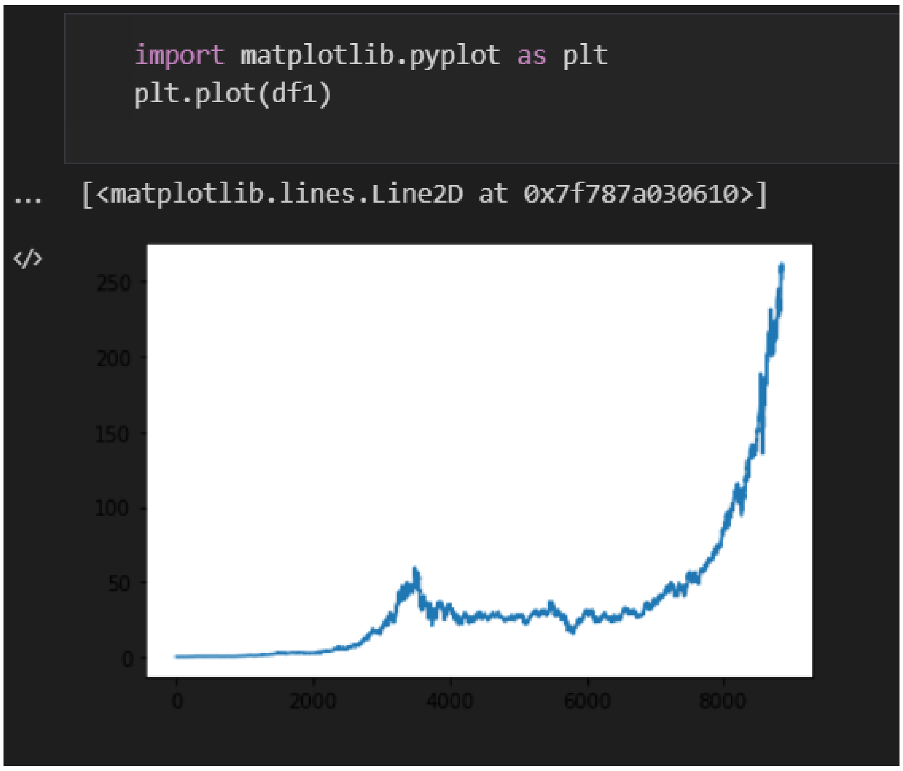
1. Loading the Data
2. LSTM is sensitive to the scale of the data, so we apply MinMax scaler
3. Train and Test Split

4. Data Preprocessing
5. Convert an array of values into a dataset matrix and reshaping the input for LSTM
6. Implementation of stacked LSTM
7. Prediction and checking performance metrics(RMSE)
8. Plotting
9. Demonstrating prediction for next 10 days
10. Checking final performance metric(R2 Score)

The following diagram shows the structure of the model that we have made:

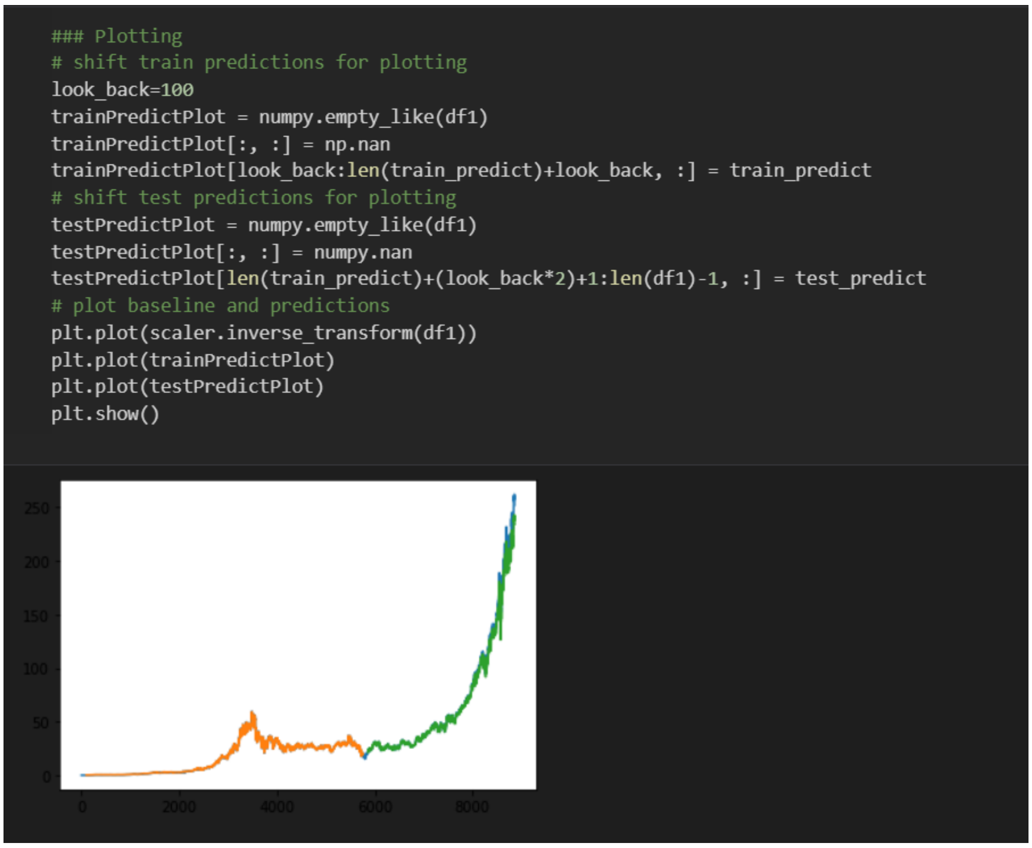


Output Plots



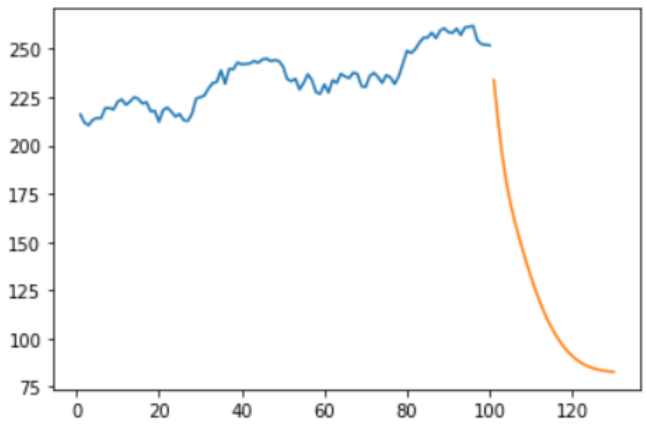
```
import matplotlib.pyplot as plt
plt.plot(df1)
```

```
[<matplotlib.lines.Line2D at 0x7f787a030610>]
```



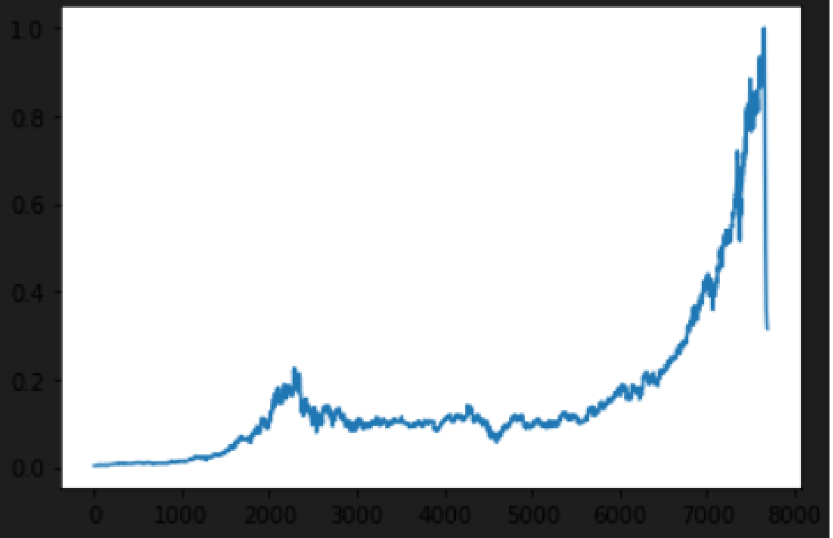
```
### Plotting
# shift train predictions for plotting
look_back=100
trainPredictPlot = numpy.empty_like(df1)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[look_back:len(train_predict)+look_back, :] = train_predict
# shift test predictions for plotting
testPredictPlot = numpy.empty_like(df1)
testPredictPlot[:, :] = numpy.nan
testPredictPlot[len(train_predict)+(look_back*2)+1:len(df1)-1, :] = test_predict
# plot baseline and predictions
plt.plot(scaler.inverse_transform(df1))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
plt.plot(day_new,scaler.inverse_transform(df1[8757:]))
plt.plot(day_pred,scaler.inverse_transform(lst_output)
```

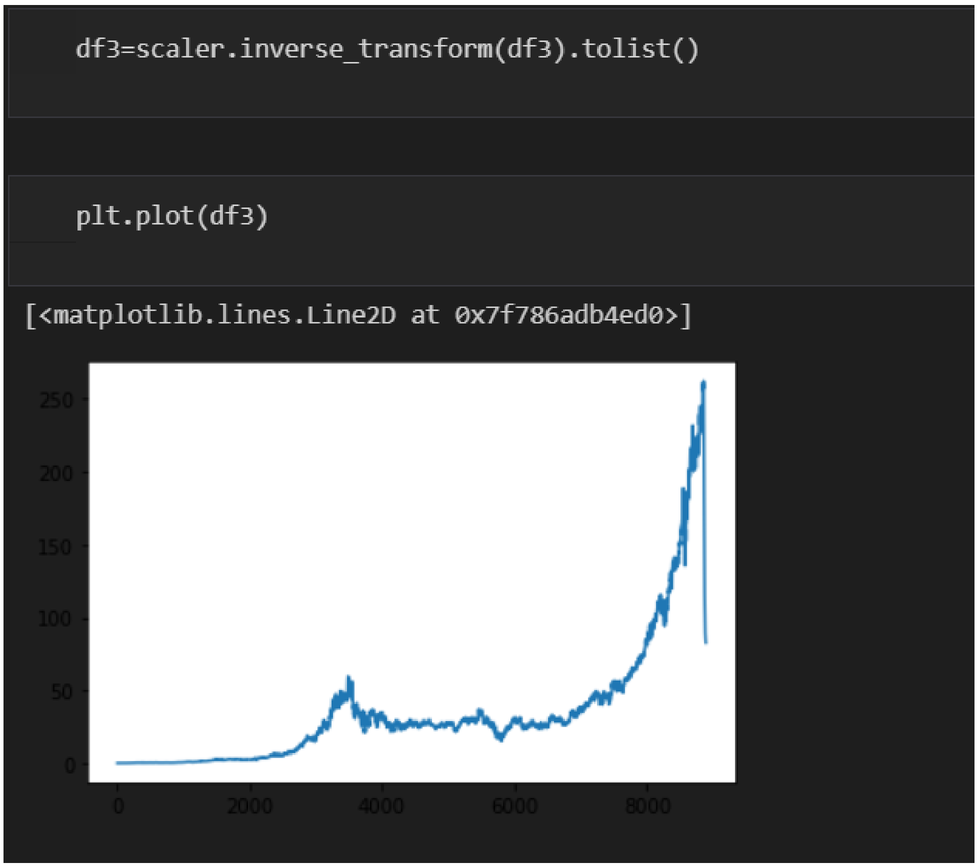
```
[<matplotlib.lines.Line2D at 0x7f786b0cb190>]
```



```
df3=df1.tolist()
df3.extend(lst_output)
plt.plot(df3[1200:])
```

```
... [<matplotlib.lines.Line2D at 0x7f786af47fd0>]
```

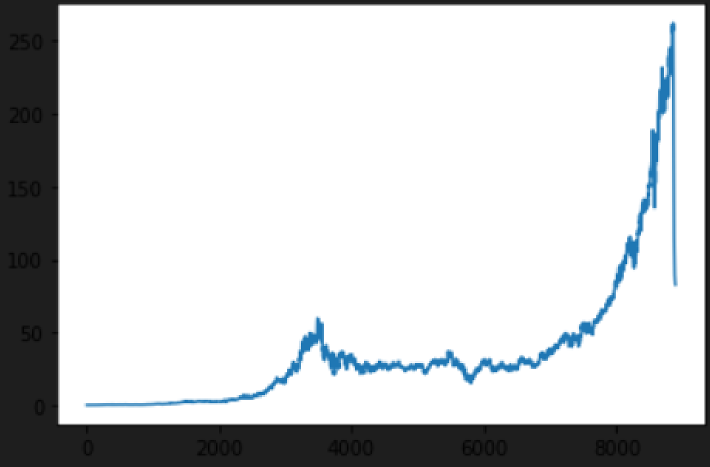




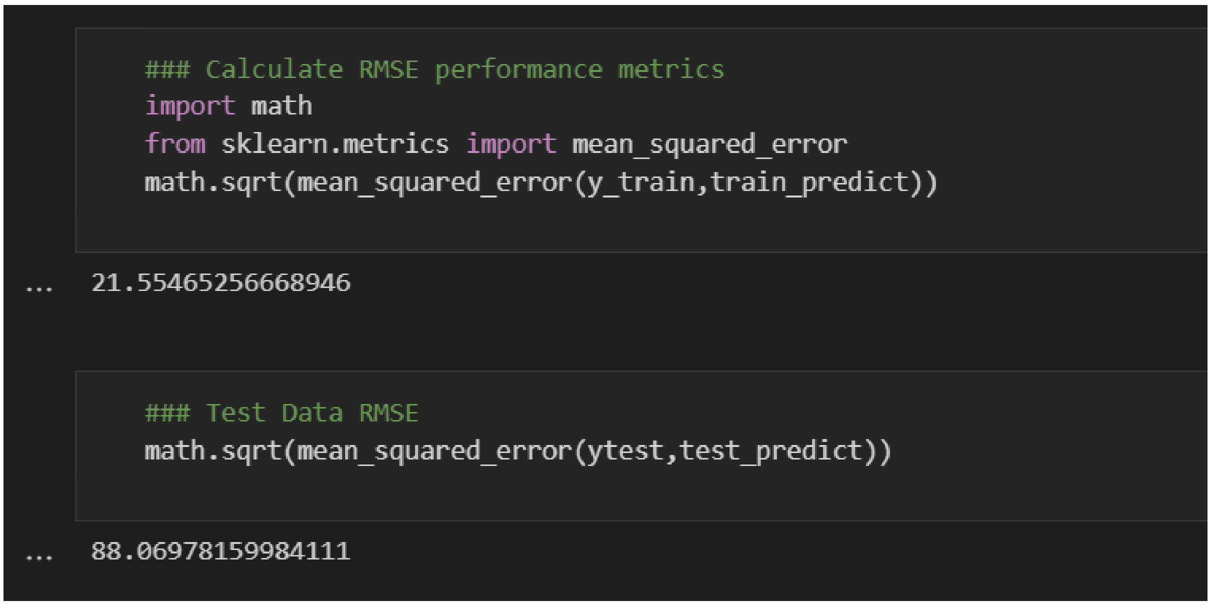
```
df3=scaler.inverse_transform(df3).tolist()
```

```
plt.plot(df3)
```

```
[<matplotlib.lines.Line2D at 0x7f786adb4ed0>]
```



Performance metrics



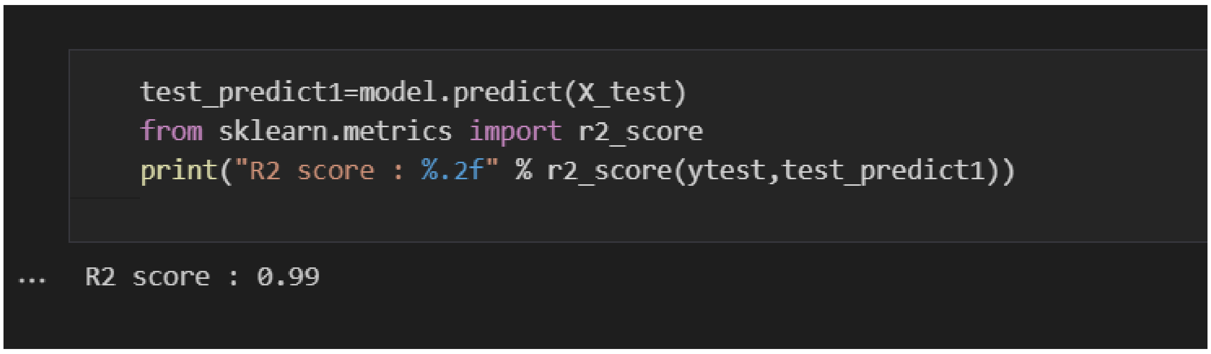
```
### Calculate RMSE performance metrics
import math
from sklearn.metrics import mean_squared_error
math.sqrt(mean_squared_error(y_train,train_predict))
```

```
... 21.55465256668946
```

```
### Test Data RMSE
math.sqrt(mean_squared_error(ytest,test_predict))
```

```
... 88.06978159984111
```

RMSE



```
test_predict1=model.predict(X_test)
from sklearn.metrics import r2_score
print("R2 score : %.2f" % r2_score(ytest,test_predict1))
```

```
... R2 score : 0.99
```

R2 Score