SUMMER INTERNSHIP PROJECT REPORT

Area Of Online Internship	**AI/ML/DL**
Intern Name	**Nitu Saha**
Name Of Institution	**Indian Institute of Technology, Indore (IIT Indore)**
Faculty Mentor Name	**Dr. Vimal Bhatia**
Duration	**2 Months (01/08/2021 TO 30/09/2021)**
Date Of Submission	**30/09/2021**

Acknowledgement

In performing my project, I had to take the help and guideline of some respected persons, who deserve my greatest gratitude. The completion of this project gives me much pleasure. I would like to show my gratitude to **Prof. Vimal Bhatia, Department of Electrical Engineering, IIT Indore** for giving me a good guideline for my project throughout numerous consultations. I would also like to expand my deepest gratitude to the research scholars for guiding me throughout the project and all those who have directly and indirectly involved in this project.

I would like to thank **Shubham Bisen**, **PhD**. **Research Scholar, Signal & Software Group (SaSg) in the Discipline of Electrical Engineering** , **IIT Indore** for his enthusiastic encouragement and precious instructions during my internship period. He gave me in-time feedback on my research and helped to organize an interesting presentation in which I could present my ideas and achievements.

I would also like to express my very great appreciation for the many software that I used throughout the development of the project as well in writing this report. Like Anaconda Navigator Environment, Jupyter Notebook as a code editor; Google docs, as a collaborative platform and to write this report. And many more small open source projects.

I thank all the people for their help directly and indirectly to complete my project.

Nitu Saha
IIEST Shibpur

Abstract

In a financially volatile market, as the stock market, it is important to have a very precise prediction of a future trend. Because of the financial crisis and scoring profits, it is mandatory to have a secure prediction of the values of the stocks. Predicting a non-linear signal requires advanced algorithms of machine learning. The literature contains studies with different machine learning algorithms such as ANN (artificial neural networks) with different feature selection.

I made the whole prediction by creating a stacked LSTM model. Forecasting stock market prices has always been a challenging task for many business analysts and researchers. In fact, stock market price prediction is an interesting area of research for investors. For successful investment, many investors are interested in knowing about the future situation of the market. Effective prediction systems indirectly help traders by providing supportive information such as the future market direction. Data mining techniques are effective for forecasting the future by applying various algorithms to data.

Introduction

Stock market prediction is the act of trying to determine the future trend of a company stock or other financial instrument traded on an exchange. The successful prediction or classification of a stock's trend could yield significant profit. In this paper, I will select some factors that are strongly related to the trend of a stock, and try to analyze the relationship between the selected factors and the trend of a stock by using machine learning techniques to train a classifier on some observed data. I will further generalize the classifier to the prediction of the trend of a stock in the future. The prediction is achieved by labeling stocks with “good” or “bad”, which is actually a binary classification problem in machine learning. Logistic regression for binary classification is considered to be used with L2 regularization to avoid overfitting. Also I am using a stacked LSTM model.

As the stock price, volume and other data contains a large amount of information affecting the stock price changes, ANN can learn the historical data of the stock, so as to find the law of stock prices. But the financial data is affected by many factors in reality, and the time series formed by it is more random and random, and it usually has multi level and multi scale characteristics. Therefore, The prediction model of a single neural network has limitations and has a certain impact on the prediction accuracy of stock prices.

LSTM neural network is a special kind of recurrent network. LSTM can keep the error, for the reverse pass along the time and layer. LSTM keeps the error at a more constant level, so that a recursive network can take a lot of time to learn, so as to open the establishment of a long distance causal link. In this paper, we use characteristics of the LSTM neural network algorithm on time series to predict short-term changes of the corresponding stock transaction.

LSTM Neural Network

Long-Short-Term Memory (LSTM) Recurrent Neural Network, one of the popular deep learning models, used in stock market prediction. In this task, we will fetch the historical data of stock automatically using python libraries and fit the LSTM model on this data to predict the future prices of the stock.
Long-Short-Term Memory Recurrent Neural Network belongs to the family of deep learning algorithms. It is a recurrent network because of the feedback connections in its architecture. It has an advantage over traditional neural networks due to its capability to process the entire sequence of data. Its architecture comprises the ***cell***, ***input gate***, ***output gate*** and ***forget gate.***

Modeling

In this task, the future stock prices of Tata Global are predicted using the LSTM Recurrent Neural Network. My task is to predict stock prices for a few days, which is a time series problem. The LSTM model is very popular in time-series forecasting, and this is

the reason why this model is chosen in this task. The historical prices of Tata Global are collected automatically using the *nsepy* library of python. I have used 5 years of historical price data, from 01.10.2013 to 01.10.2018.

```
In [3]: df.head()
```

Out[3]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2018-10-08	208.00	222.25	206.85	216.00	215.15	4642146.0	10062.83
1	2018-10-05	217.00	218.60	205.90	210.25	209.20	3519515.0	7407.06
2	2018-10-04	223.50	227.80	216.15	217.25	218.20	1728786.0	3815.79
3	2018-10-03	230.00	237.50	225.75	226.45	227.60	1708590.0	3960.27
4	2018-10-01	234.55	234.60	221.05	230.30	230.90	1534749.0	3486.05

```
In [4]: df.tail()
```

Out[4]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
1230	2013-10-14	160.85	161.45	157.70	159.3	159.45	1281419.0	2039.09
1231	2013-10-11	161.15	163.45	159.00	159.8	160.05	1880046.0	3030.76
1232	2013-10-10	156.00	160.80	155.85	160.3	160.15	3124853.0	4978.80
1233	2013-10-09	155.70	158.20	154.15	155.3	155.55	2049580.0	3204.49
1234	2013-10-08	157.00	157.80	155.20	155.8	155.80	1720413.0	2688.94

Data Preprocessing

This data set contains 1235 observations with 8 attributes. After preprocessing, only dates and OHLC (Open, High, Low, Close) columns, a total of 5 columns, are taken as these columns have

main significance in the dataset. The LSTM model is trained on this entire dataset, and for the testing purpose, a new dataset is fetched for the next 30 days. The stock prices for this new duration will be predicted by the already trained LSTM model, and the predicted prices will be plotted against the original prices to visualise the model's accuracy.

```
In [20]: #reshape input
         X_train = X_train.reshape(X_train.shape[0],X_train.shape[1],1)
         X_test  = X_test.reshape(X_test.shape[0],X_test.shape[1],1)

In [21]: #create LSTM model
         import numpy as np
         import tensorflow as tf
         from tensorflow import keras
         from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Dense
         from tensorflow.keras.layers import LSTM

In [22]: model=Sequential()
         model.add(LSTM(50,return_sequences=True,input_shape=(100,1)))
         model.add(LSTM(50,return_sequences=True))
         model.add(LSTM(50))
         model.add(Dense(1))
         model.compile(loss='mean_squared_error',optimizer='adam')

In [23]: model.summary()
         Model: "sequential"
         _________________________________________________________________
         Layer (type)                 Output Shape              Param #
         =================================================================
         lstm (LSTM)                  (None, 100, 50)           10400
         _________________________________________________________________
         lstm_1 (LSTM)                (None, 100, 50)           20200
         _________________________________________________________________
         lstm_2 (LSTM)                (None, 50)                20200
         _________________________________________________________________
         dense (Dense)                (None, 1)                 51
         =================================================================
         Total params: 50,851
         Trainable params: 50,851
         Non-trainable params: 0
         _________________________________________________________________
```

Fig : **LSTM model creation**

Implementation

1. Import the required libraries.
2. I fetched 5 years of historical prices of Tata Global from **01.10.2013** to **01.10.2018** So I set the start and end dates and passed these parameters to the function for fetching the data.
3. We can visualise the fetched data. For simplicity, only the day-wise closing prices are visualised.
4. There are 8 columns in the fetched data. Many of the columns are not of our interest so only significant columns are selected to create the main dataset.
5. Preprocessed the data in order to prepare it for the LSTM model. The data fetched in step one is used for training the purpose only. For testing purposes, different data will be fetched later.
6. Defined the LSTM Recurrent Neural Network. Here, I can add more LSTM layers and adjust the dropout in order to improve the accuracy of the model.

7. Compiled and trained the model defined in the above step. Iteratively, you can increase or decrease the epochs and batch size to get more accuracy.
8. Now, after training the model it needs to be tested on the testing data. For this purpose, I fetched the new data for a different period. Preprocessing steps are similar as we have done with training data.
9. Tested the LSTM model on the new dataset.
10. Visualized the predicted stock prices with original stock prices.

Final Plotting

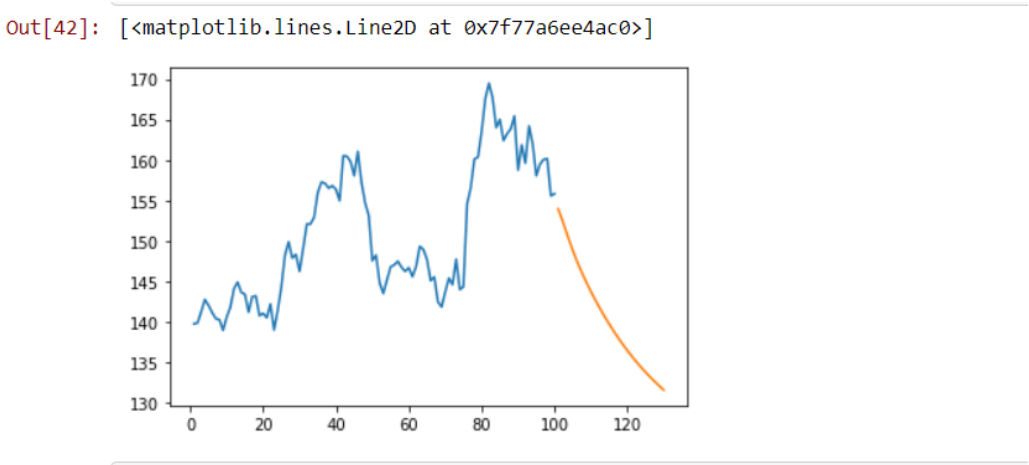

Fig:1(a)

```
Out[43]: [<matplotlib.lines.Line2D at 0x7f77a6e46700>]
```

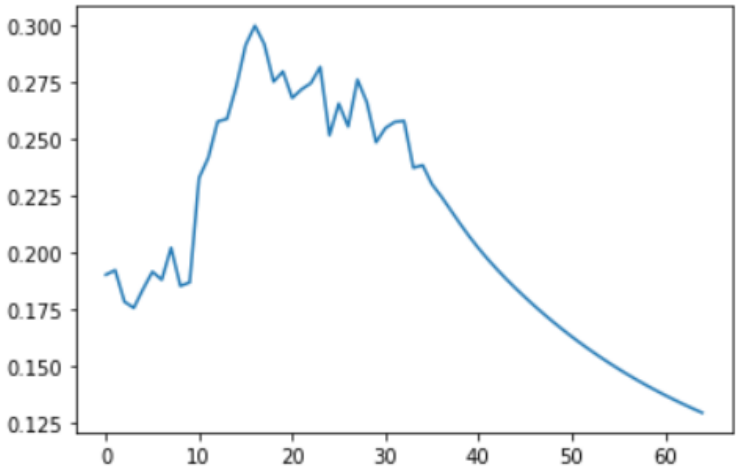

Fig: 1(b)

```
In [45]: plt.plot(df3)
Out[45]: [<matplotlib.lines.Line2D at 0x7f77a6e7e610>]
```

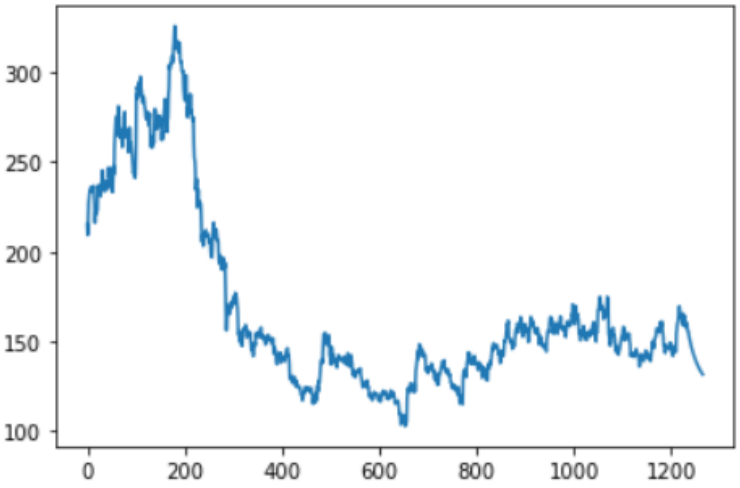

Fig: 1(c)

Experiment Discussions

There are several technologies that have been used in this project. Such as:

Python:

Python was the language of choice for this project. This was an easy decision for multiple reasons.

Python as a language has an enormous community behind it. Any problems that might be encountered can be easily solved with a trip to Stack Overflow. Python is among the most popular languages on the site which makes it very likely there will be a direct answer to any query.

1. Python has an abundance of powerful tools ready for scientific computing. Packages such as Numpy, Pandas, and SciPy are freely available and well documented. Packages such as these can dramatically reduce, and simplify the code needed to write a given program. This makes iteration quick.
2. Python as a language is forgiving and allows for programs that look like pseudo code. This is useful when pseudocode given in academic papers needs to be implemented and tested. Using Python, this step is usually reasonably trivial.

However, Python is not without its flaws. The language is dynamically typed and packages are notorious for Duck Typing. This can be frustrating when a package method returns something that, for example, looks like an array rather than being an actual array. Coupled with the fact that standard Python documentation does not explicitly state the return type of a method, this can lead to a lot of trials and error testing that would not otherwise happen in a strongly typed language. This is an issue that makes learning to use a new Python package or library more difficult than it otherwise could be.

Numpy:

Numpy is a python module which provides scientific and higher level mathematical abstractions wrapped in python. In most programming languages, we can't use mathematical abstractions such as $f(x)$ as it would affect the semantics and the syntax of the code. But by using Numpy we can exploit such functions in our code.

Numpy's array type augments the Python language with an efficient data structure used for numerical work, e.g., manipulating matrices. Numpy also provides basic numerical routines, such as tools for finding Eigenvectors.

Scikit Learn:

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machine, random forest, gradient boosting, k-means etc. It is mainly designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Tensorflow:

TensorFlow is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API. TensorFlow was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well.

Keras:

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.

Keras allows for easy and fast prototyping (through user friendliness, modularity, and extensibility). Supports both convolutional networks and recurrent networks, as well as combinations of the two. Runs seamlessly on CPU and GPU.

Compiler Option:

Anaconda is a freemium open source distribution of the Python and R programming languages for large-scale data processing, predictive analytics, and scientific computing, that aims to simplify package management and deployment. Package versions are managed by the package management system conda.

Conclusion

In this project, I used the LSTM recurrent neural networks to extract feature value and analyze the stock data. The experimental results show our model can play a better forecasting effect, even though the accuracy is not very high, only about 72% for the short period of data. But I believe the model still has a lot of improvements to improve its accuracy. In the future, I will extract more feature values to train our model and improve the model. At the same time, I will also compare the pros and cons of different neural networks models in stock forecasting to find a better model to improve the prediction accuracy.

References

I. Stock Market Behaviour Prediction using Stacked LSTM Networks by Samuel Olusegun Ojo*, Pius Adewale Owolawi†, Maredi Mphahlele and Juliana Adeola Adisa§ Department of Computer Systems Engineering, Tshwane University of Technology Pretoria, South Africa.

II. Stock market prediction & efficiency analysis using Recurrent Neural Network by Joish Bosco.

III. Stock Price prediction using LSTM and SVR by Gourav Bhatla.

IV. Understanding LSTM Networks from Colah's Blog.