औद्योगिक प्रशिक्षण के लिए राष्ट्रीय संस्थान

National Institute for Industrial Training

One Premier Organization with Non Profit Status | Registered Under Govt. of WB
Empanelled Under Planning Commission Govt. of India
Inspired By: National Task Force on IT & SD Government of India

National Institute for Industrial Training- One Premier Organization with Non Profit Status Registered Under Govt.of West Bengal,Empanelled Under Planning Commission Govt.of India , Empanelled Under Central Social Welfare Board Govt. of India , Registered with National Career Services , Registered with National Employment Services.

STOCK MARKET PREDICTION USING PYTHON & MACHINE LEARNING

REPORT

Shreya Sarkar

<shreyasarkar0002@gmail.com>

Supervised By: Soumotanu Mazumdar

STUDENT PROFILE

NAME: SHREYA SARKAR

COLLEGE NAME: B.P.PODDAR INSTITUTE OF MANAGEMENT AND TECHNOLOGY

STREAM: B.TECH ECE (3rd Year)

EMAIL: shreyasarkar0002@gmail.com

PROJECT: STOCK MARKET PREDICTION USING PYTHON & MACHINE LEARNING

GITHUB PROFILE: https://github.com/tutulshreya

ABSTRACT:

It has never been easy to invest in a set of assets, the abnormally of financial market does not allow simple models to predict future asset values with higher accuracy. Machine learning, which consist of making computers perform tasks that normally requiring human intelligence is currently the dominant trend in scientific research. This article aims to build a model using Recurrent Neural Networks (RNN) and especially Long-Short Term Memory model (LSTM) to predict future stock market values. The main objective of this paper is to see in which precision a Machine learning algorithm can predict and how much the epochs can improve our model.

ACKNOWLEDGEMENTS:

The final outcome of this project required a lot of guidance and assistance from many people and internet source .I am extremely privileged to have got this all along the completion of my project All that I have done is only due to such supervision and assistance and I would not forget to thank them.

I would like to thank my supervisor Mr. Soumotanu Mazumdar Of National Institute for Industrial Training who gave me this golden opportunity to work on this project. I got to learn a lot of research from this project about prediction of stock market in daily life. I would also like to thank him for guiding me in an exemplary manner.

Introduction

Several studies have been the subject of using machine learning in the quantitative financial, predicting prices of managing and constricting entire portfolio of assets, as well as, investment process, and many other operations can be covered by machine learning algorithms. In general machine learning is a term used for all algorithm's methods using computers to reveal patterns based only on data and not using any programming instructions. For quantitative finance and specially assets selections several models supply a large number of methods that can be used with machine learning to forecast future assets value. This type of models offers a mechanism that combine weak sources of information and make it a strange tool that can be used efficiently. Recently, the combination of statistics and learning models have polished several machine learning algorithms, such as aritificial neural networks, gradient boosted regression trees, support vector machines and, random forecast. These algorithms can reveal complex patterns characterized by non-linearity as well as some relations that are difficult to detect with linear algorithms. These algorithms also prove more effectiveness and multi collinearity than the linear regressions ones. A large number of studies is currently active on the subject of machine learning methods used in finance, some studies used tree-based models to predict portfolio returns, others used deep learning in the production of future values of financial assets. Also, some authors overviewed the forecasting of returns using of ADaBoost algorithm .

Others proceeds to forecast stock returns using unique decision-making model for day trading investments on the stock market the model developed and use the support vector machine (SVM) method, and the mean-variance (MV) method for portfolio selection. Another paper conversed deep learning models for smart indexing. Also, some study

has covered a large number of trends and Applications of Machine Learning in Quantitative Finance, the literature review covered by this paper consist of return forecasting portfolio construction, ethics, fraud detection, decision making, language processing and sentiment analysis. These models don't depend one long term memory (passed sequences of data), in this regard a class of machine learning algorithms based on Recurrent Neural Network proves to be very useful in financial market price prediction and forecasting. A paper has compares the accuracy of autoregressive integrated moving average ARIMA and LSTM, as illustrative techniques when forecasting time series data. These techniques were executed on a set of financial data and the results showed that LSTM was far more superior to ARIMA. Our aim is to use ML algorithm based on LSTM RNN to forecast the adjusted closing prices for a portfolio of assets. The main objective here is to obtain the most accurate trained algorithm, to predict future values .

OBJECTIVES:

The main objectives of the proposed system are:

(i) To see in which precision a Machine learning algorithm can predict and how much the epochs can improve our model.

Programming Language - Python:

Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object- oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library. Two major versions of Python are currently in active use:Python 3.x is the current version and is under active development. Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

Advantages of Python

1. Easy Syntax
2. Readability
3. High-Level Language
4. Object-oriented programming
5. It's Open source and Free
6. Cross-platform
7. Widely Supported
8. It's Safe
9. Extensible

Easy Syntax of Python

Python's syntax is easy to learn, so both non- programmers and programmers can start programming right away.

Very Clear Readability of Python

Python's syntax is very clear, so it is easy to understand program code. (Python is often referred to as "executable pseudo-code" because its syntax mostly follows the conventions used by programmers to outline their ideas without the formal verbosity of code in most programming languages.

Python High-Level Language

Python looks more like a readable, human language than like a low-level language. This gives you the ability to program at a faster rate than a low-level language will allow you.

Python Is Open-Source and Free

Python is both free and open-source. The Python Software Foundation distributes pre -made binaries that are freely available for use on all major operating systems called CPython. You can get CPython's source-code, too. Plus, we can modify the source code and distribute as allowed by CPython's license.

Python is a Cross-platform

Python runs on all major operating systems like Microsoft Windows, Linux, and Mac OS X.

Python Object-oriented programming

Object-oriented programming allows you to create data structures that can be reused, which reduces the amount of repetitive work that you'll need to do. Programming languages usually define objects with namespaces, like class or def, and objects can edit themselves by using keyword, like this or self. Most modern programming languages are object-oriented (such as Java, C++, and C#) or have support for OOP features (such as Perl version 5 and later). Additionally, object-oriented techniques can be used in the design of almost any non-trivial software and implemented in almost any programming or scripting language.

Python Widely Supported Programming Language .Python has an active support community with many websites, mailing lists, and USENET "net news" groups that attract a large number of knowledgeable and helpful contributors.

Python is a Safe

Python doesn't have pointers like other C-based languages, making it much more reliable. Along with that, errors never pass silently unless they're explicitly silenced. This allows you to see and read why the program crashed and where to correct your error.

Python Batteries Included Language

Python is famous for being the "batteries are included" language. There are over 300 standards library modules which contain modules and classes for a wide variety of programming tasks. For example, the

standard library contains modules for safely creating temporary files (named or anonymous), mapping files into memory (including use of shared and anonymous memory mappings), spawning and controlling sub-processes, compressing and decompressing files (compatible with gzip or PK-zip) and archives files (such as Unix/Linux “tar”). Accessing indexed “DBM”(database) files, interfacing to various graphical user interfaces (such as the TK toolkit and the popular WxWindows multi-platform windowing system), parsing and maintaining CSV (comma-separated values) and “.cfg” or “.in configuration files (similar in syntax to the venerable WIN.INI files from MS-DOS and MS-Windows), for sending e-mail, fetching and parsing web pages, etc. It’s possible, for example, to create a custom web server in Python using less than a dozen lines of code, and one of the standard libraries, of course.

Python is Extensible

In addition to the standard libraries there are extensive collections of freely available add-on modules, libraries, frameworks, and tool-kits. These generally conform to similar standards and conventions. For example, almost all of the database adapters (to talk to almost any client-server RDBMS engine such as MySQL, Postgres, Oracle, etc) conform to the Python DBAPI and thus can mostly be accessed using the same code. So, it’s usually easy to modify a Python program to support any database engine.

Future Scopes of Python

Python is one of the fastest growing languages and has undergone a successful span of more than 25 years as far as its adoption is concerned. This success also reveals a promising future scope of python

programming language. In fact, it has been continuously serving as the best programming language for application development, web development, game development, system administration, scientific and numeric computing, GIS and Mapping etc.

Popularity of python

The reason behind the immense popularity of python programming language across the globe is the features it provides. Have a look at the features of python language.

(1) Python Supports Multiple Programming Paradigms:

Python is a multi-paradigm programming language including features such as object-oriented, imperative, procedural, functional, reflective etc.

(2) Python Has Large Set of Library and Tools

Python has very extensive standard libraries and tools that enhance the overall functionality of python language and also helps python programmers to easily write codes. Some of the important python libraries and tools are listed below.

● Built-in functions, constants, types, and exceptions.

● File formats, file and directory access, multimedia services.

● GUI development tools such as Tkinter

● Custom Python Interpreters, Internet protocols and support, data compression and archiving, modules etc.

● Scrappy, wxPython, SciPy, matplotlib, Pygame, PyQT, PyGTK etc.

(3) Python Has a Vast Community Support

This is what makes python a favourable choice for development purposes. If you are having problems writing python a program, you can post directly to python community and will get the response with the

solution of your problem. You will also find many new ideas regarding python technology and change in the versions.

(4) Python is Designed for Better Code Readability

Python provides a much better code readability as compared to another programming language. For example, it uses whitespace indentation in place of curly brackets for delimiting the block of codes.

(5) Python Has a Vast Community Support

This is what makes python a favourable choice for development purposes. If you are having problems writing python a program, you can post directly to python community and will get the response with the solution of your problem. You will also find many new ideas regarding python technology and change in the versions.

(6) Python is Designed for Better Code Readability

Python provides a much better code readability as compared to another programming language. For example, it uses whitespace indentation in place of curly brackets for delimiting the block of codes.

(7) Python Contains Fewer Lines of Codes

Codes are written in python programming language complete in fewer lines thus reducing the efforts of programmers. Let’s have a look on the following “Hello World” program written in C, C++, Java, and Python. While, C, C++, and Java take six, seven and five lines respectively for a

simple “Hello World” program. Python takes only a single line which means, less coding effort and time is required for writing the same program.

Future Technologies Counting on Python

Generally, we have seen that python programming language is extensively used for web development, application development, system administration, developing games etc.

But do you know there are some future technologies that are relying on python? As a matter of fact, Python has become the core language as far as the success of these technologies is concerned. Let’s dive into the technologies which use python as a core element for research, production and further developments.

(1) Artificial Intelligence (AI)

Python programming language is undoubtedly dominating the other languages when future technologies like Artificial Intelligence (AI) come into the play.

There are plenty of python frameworks, libraries, and tools that are specifically developed to direct Artificial Intelligence to reduce human efforts with increased accuracy and efficiency for various development purposes. It is only the Artificial Intelligence that has made it possible to develop speech recognition system, autonomous cars, interpreting data like images, videos etc.

We have shown below some of the python libraries and tools used in various Artificial Intelligence branches.

● Machine Learning- PyML, PyBrain, scikit-learn, MDP Toolkit, GraphLab Create, MIPy etc.

● General AI- pyDatalog, AIMA, EasyAI, SimpleAI etc.

● Neural Networks- PyAnn, pyrenn, ffnet, neurolab etc.

● Natural Language & Text Processing- Quepy, NLTK, gensim

(2) Big Data

The future scope of python programming language can also be predicted by the way it has helped big data technology to grow. Python has been successfully contributing in analysing a large number of data sets across computer clusters through its high-performance toolkits and libraries. Let's have a look at the python libraries and toolkits used for Data analysis and handling other big data issues.

● Pandas

● Scikit-Learn

● NumPy

● SciPy

● GraphLab Create

● IPython

● Bokeh

● Agate

● PySpark

● Dask

(3) Networking

Networking is another field in which python has a brighter scope in the future. Python programming language is used to read, write and configure routers and switches and perform other networking automation tasks in a cost-effective and secure manner. For these purposes, there are many libraries and tools that are built on the top of the python language. Here we have listed some of these python libraries and tools especially used by network engineers for network automation.

● Ansible

● Netmiko

● NAPALM (Network Automation and Programmability Abstraction Layer with Multivendor Support)

● Pyeapi

● Junos PyEZ

● PySNMP

● Paramiko SSH

Real-Life Python Success Stories

Python has seemingly contributed as a core language for increasing productivity regarding various development purposes at many of the IT organizations. We have shown below some of the real-life python success stories.

● Australia's RMA Department D-Link has successfully implemented python for creating DSL Firmware Recovery System.

● Python has helped Gusto.com, an online travel site, in reducing development costs and time.

● ForecastWatch.com also uses python in rating the accuracy of weather forecast reports provided by companies such as Accuweather, MyForecast.com and The Weather Channel.

● Python has also benefited many product developments companies such as Acqutek, AstraZeneca, GravityZoo, Carmanah Technologies Inc. etc in creating autonomous devices and software.

● Test&Go uses python scripts for Data Validation.

● Industrial Light & Magic(ILM) also uses python for batch processing that includes modeling, rendering and compositing thousands of picture frames per day.

There is a huge list of success stories of many organizations across the globe which are using python for various purposes such as software development, data mining, unit testing, product development, web development, data validation, data visualization etc. These success stories directly point towards a promising future scope of python programming language.

Top Competitors of Python

The future scope of python programming language also depends on its competitors in the IT market. But, due to the fact that it has become a core language for future technologies such as artificial intelligence, big

data, etc., it will surely rise further and will be able to beat its competitors.

Competitors and Alternatives to Python Programming Language

∑ Microsoft.

∑ Oracle.

∑ IBM.

∑ Tableau.

∑ SAP.

∑ Alteryx.

∑ Blue Yonder.

∑ Gurobi.2

Imports

The libraries that have been imported for this project are stated as follows:

1. NumPy: NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

2. matplotlib: matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general- purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. It is used to plot for data visualization.

Predicting stock prices is an uncertain task which is modelled using machine learning to predict the return on stocks. There are a lot of methods and tools used for the purpose of stock market prediction. The stock market is considered to be very dynamic and complex in nature. An accurate prediction of future prices may lead to a higher yield of profit for investors through stock investments. As per the predictions, investors will be able to pick the stocks that may give a higher return.

Over the years, various machine learning techniques have been used in stock market prediction, but with the increased amount of data and expectation of more accurate prediction, the deep learning models are being used nowadays which have proven their advantage over traditional machine learning methods in terms of accuracy and speed of prediction. In this article, we will discuss the 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.

LSTM Recurrent Neural Network

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*.

The cell remembers values over arbitrary time intervals, and the three gates regulate the flow of information into and out of the cell. The cell of the model is responsible for keeping track of the dependencies between the elements in the input sequence. The input gate controls the

extent to which a new value flows into the cell, the forget gate controls the extent to which a value remains in the cell, and the output gate controls the extent to which the value in the cell is used to compute the output activation of the LSTM unit.

However, there are some variants of the LSTM model such as Gated Recurrent Units (GRUs) that do not have the output gate. LSTM Networks are popularly used on time-series data for classification, processing, and making predictions. The reason for its popularity in time-series application is that there can be several lags of unknown duration between important events in a time series.

Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is one of many types of Recurrent Neural Network RNN, it's also capable of catching data from past stages and use it for future predictions . In general, an Artificial Neural Network (ANN) consists of three layers:

1) input layer,

2) Hidden layers,

3) output layer.

In a NN that only contains one hidden layer the number of nodes in the input layer always depend on the dimension of the data, the nodes of the input layer connect to the hidden layer via links called 'synapses'. The relation between every two nodes from (input to the hidden layer), has a coefficient called weight, which is the decision maker for signals. The process of learning is naturally a continues adjustment of weights, after completing the process of learning, the Artificial NN will have optimal weights for each synapses. The hidden layer nodes apply a sigmoid or

tangent hyperbolic (tanh) function on the sum of weights coming from the input layer which is called the activation function, this transformation will generate values, with a minimized error rate between the train and test data using the SoftMax function. The values obtained after this transformation constitute the output layer of our NN, these value may not be the best output, in this case a back propagation process will be applied to target the optimal value of error, the back propagation process connect the output layer to the hidden layer, sending a signal conforming the best weight with the optimal error for the number of epochs decided. This process will be repeated trying to improve our predictions and minimize the prediction error. After completing this process, the model will be trained. The classes of NN that predict future value base on passed sequence of observations is called Recurrent Neural Network (RNN) this type of NN make use of earlier stages to learn of data and forecast futures trends. The earlier stages of data should be remembered to predict and guess future values, in this case the hidden layer act like a stock for the past information from the sequential data. The term recurrent is used to describe the process of using elements of earlier sequences to forecast future data. RNN can't store long time memory, so the use of the Long Short-Term Memory (LSTM) based on "memory line" proved to be very useful in forecasting cases with long time data. In a LSTM the memorization of earlier stages can be performed trough gates with along memory line incorporated.

The ability of memorizing sequence of data makes the LSTM a special kind of RNNs. Every LSTM node most be consisting of a set of cells responsible of storing passed data streams, the upper line in each cell links the models as transport line handing over data from the past to the present ones, the independency of cells helps the model dispose filter of add values of a cell to another. In the end the sigmoidal neural network layer composing the gates drive the cell to an optimal value by disposing

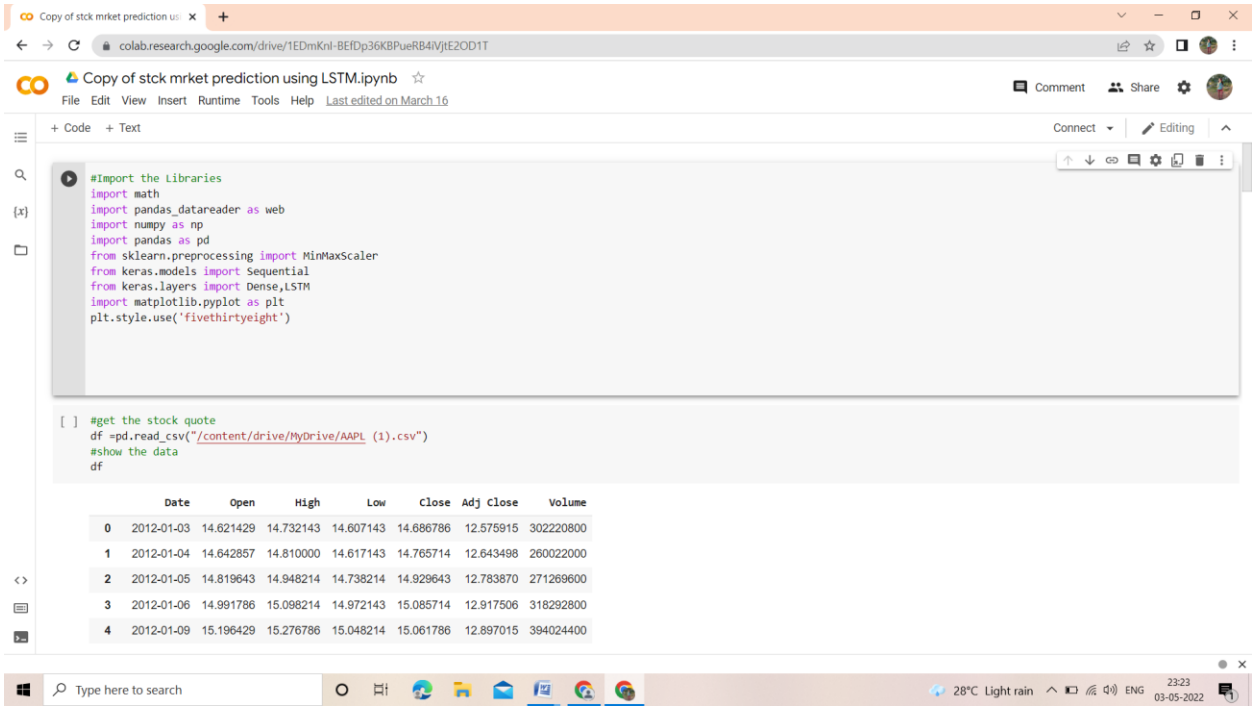
or letting data pass through. Each sigmoid layer has a binary value (0 or 1) with 0 “let nothing pass through”; and 1 “let everything pass through.” The goal here is to control the state of each cell, the gates are controlled as follow:

- Forget Gate outputs a number between 0 and 1, where 1 illustration “completely keep this”; whereas, 0 indicates “completely ignore this.”

- Memory Gate chooses which new data will be stored in the cell. First, a sigmoid layer “input door layer” chooses which values will be changed. Next, a tanh layer makes a vector of new candidate values that could be added to the state.

- Output Gate decides what will be the output of each cell. The output value will be based on the cell state along with the filtered and freshest added data.

SOME CODE SNIPPETS & OUTPUTS:



Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

Comment Share Connect Editing

+ Code + Text

```
#Import the Libraries
import math
import pandas_datareader as web
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense,LSTM
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
```

```
[ ] #get the stock quote
    df =pd.read_csv("/content/drive/MyDrive/AAPL (1).csv")
    #show the data
    df
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2012-01-03	14.621429	14.732143	14.607143	14.686786	12.575915	302220800
1	2012-01-04	14.642857	14.810000	14.617143	14.765714	12.643498	260022000
2	2012-01-05	14.819643	14.948214	14.738214	14.929643	12.783870	271269600
3	2012-01-06	14.991786	15.098214	14.972143	15.085714	12.917506	318292800
4	2012-01-09	15.196429	15.276786	15.048214	15.061786	12.897015	394024400

Type here to search 28°C Light rain ENG 23:23 03-05-2022

Copy of stck mrket prediction us

colab.research.google.com/drive/1EDmKnI-BEfDp36KBPueRB4iVjtE2OD1T

Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

Comment Share

+ Code + Text

Connect Editing

...	...	...	...	...	...	...	...	...
1998	2019-12-11	67.202499	67.775002	67.125000	67.692497	66.616829	78756800
1999	2019-12-12	66.945000	68.139999	66.830002	67.864998	66.786591	137310400
2000	2019-12-13	67.864998	68.824997	67.732498	68.787498	67.694427	133587600
2001	2019-12-16	69.250000	70.197502	69.245003	69.964996	68.853210	128186000
2002	2019-12-17	69.892502	70.442497	69.699997	70.102501	68.988541	114158400

2003 rows × 7 columns

```
#get the no of rows and columns in the dataset
df.shape
```

```
(2003, 7)
```

```
#visualize the closing price history
plt.figure(figsize=(16,8))
plt.title('Close Price History')
plt.plot(df['Close'])
plt.xlabel('Date',fontsize=18)
plt.ylabel(' Close Price USD($)', fontsize=18)
plt.show()
```

Type here to search

28°C Light rain ENG 23:23 03-05-2022

Copy of stck mrket prediction us

colab.research.google.com/drive/1EDmKnI-BEfDp36KBPueRB4iVjtE2OD1T

Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

Comment Share

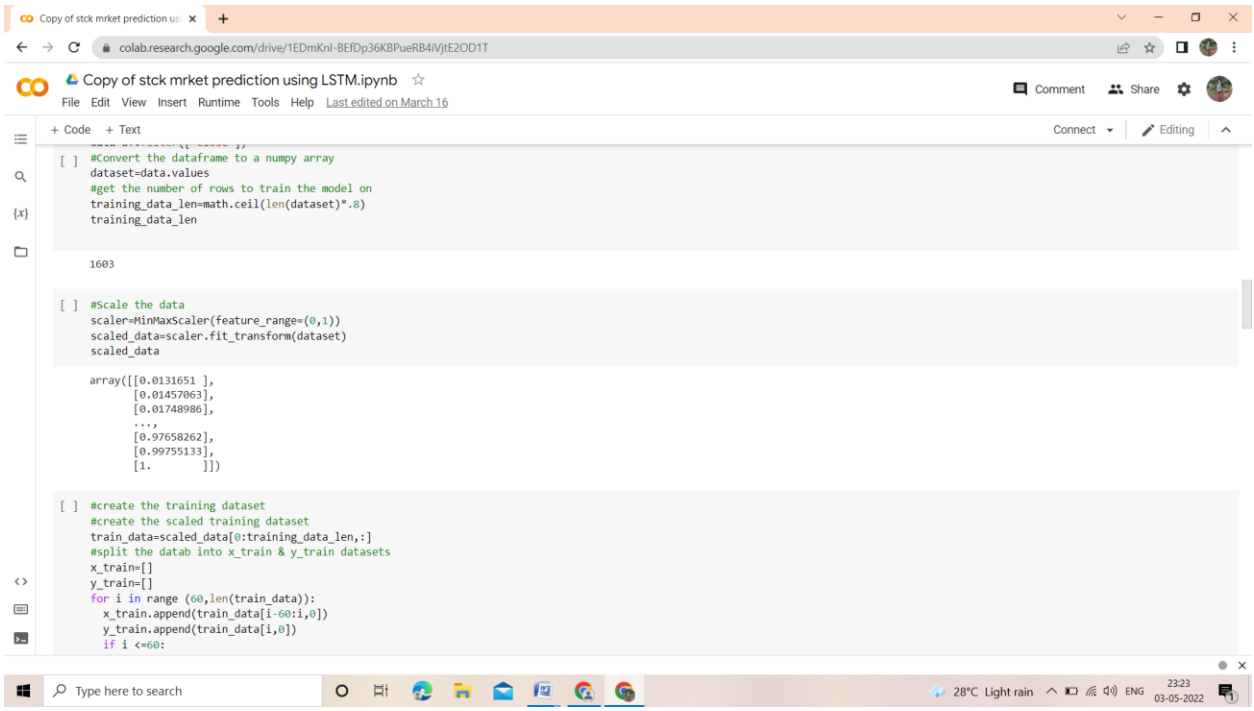
+ Code + Text

Connect Editing


```
#creating a new dataframe with only for the 'Close' Column
data=df.filter(['Close'])
#Convert the dataframe to a numpy array
```

Type here to search

28°C Light rain ENG 23:23 03-05-2022



Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

Comment Share Connect Editing

+ Code + Text

```
[ ] #Convert the dataframe to a numpy array
    dataset=data.values
    #get the number of rows to train the model on
    training_data_len=math.ceil(len(dataset)*.8)
    training_data_len
```

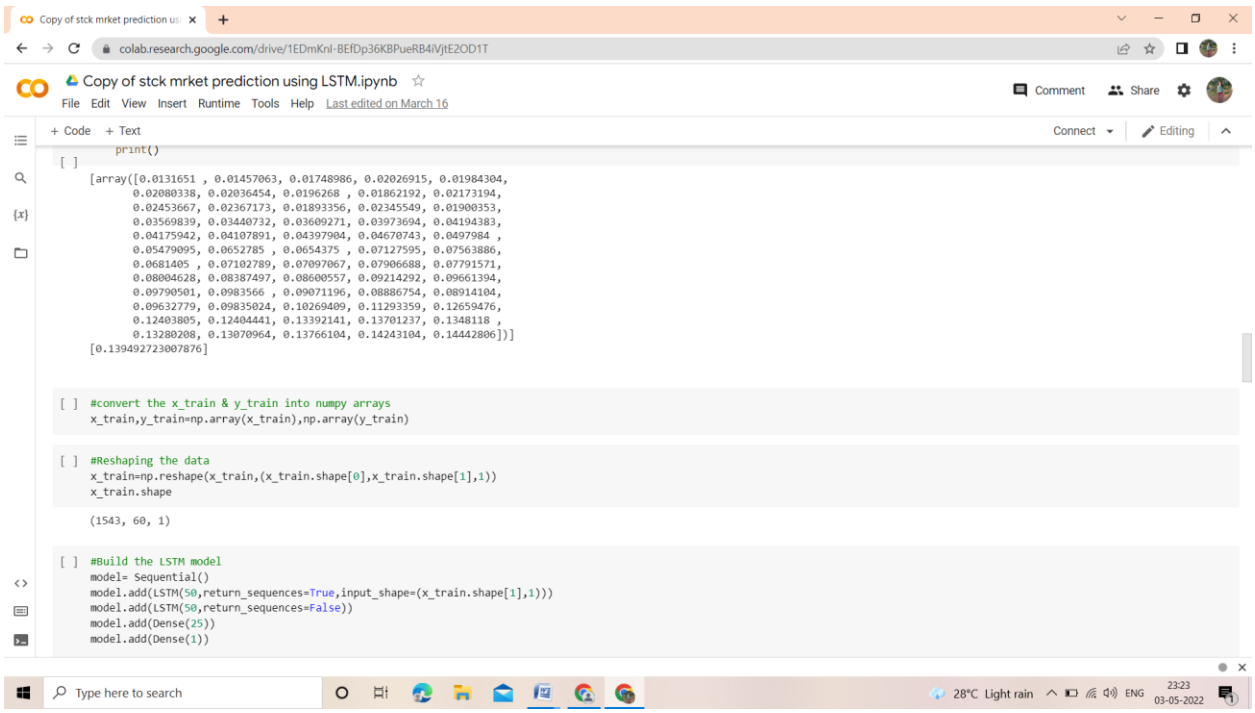
```
1603
```

```
[ ] #Scale the data
    scaler=MinMaxScaler(feature_range=(0,1))
    scaled_data=scaler.fit_transform(dataset)
    scaled_data
```

```
array([[0.0131651 ],
       [0.01457063],
       [0.01748986],
       ...,
       [0.97658262],
       [0.99755133],
       [1.        ]])
```

```
[ ] #create the training dataset
    #create the scaled training dataset
    train_data=scaled_data[0:training_data_len,:]
    #split the datab into x_train & y_train datasets
    x_train=[]
    y_train=[]
    for i in range (60,len(train_data)):
      x_train.append(train_data[i-60:i,0])
      y_train.append(train_data[i,0])
      if i <=60:
```

Type here to search 28°C Light rain ENG 23:23 03-05-2022



Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

Comment Share Connect Editing

+ Code + Text

```
[ ]   print()
```

```
[array([0.0131651 , 0.01457063, 0.01748986, 0.02026915, 0.01984304,
        0.02080338, 0.02036454, 0.0196268 , 0.01862192, 0.02173194,
        0.02453667, 0.02367173, 0.01893356, 0.02345549, 0.01900353,
        0.03569839, 0.03440732, 0.03609271, 0.03973694, 0.04194383,
        0.04175942, 0.04107891, 0.04397904, 0.04670743, 0.0497984 ,
        0.05479095, 0.0652785 , 0.0654375 , 0.07127595, 0.07563886,
        0.0681405 , 0.07102789, 0.07097067, 0.07906688, 0.07791571,
        0.08004628, 0.08387497, 0.08600557, 0.09214292, 0.09661394,
        0.09790501, 0.0983566 , 0.09071196, 0.08886754, 0.08914104,
        0.09632779, 0.09835024, 0.10269409, 0.11293359, 0.12659476,
        0.12403805, 0.12404441, 0.13392141, 0.13701237, 0.1348118 ,
        0.13280208, 0.13070964, 0.13766104, 0.14243104, 0.14442806])]
[0.13949272300787 6]
```

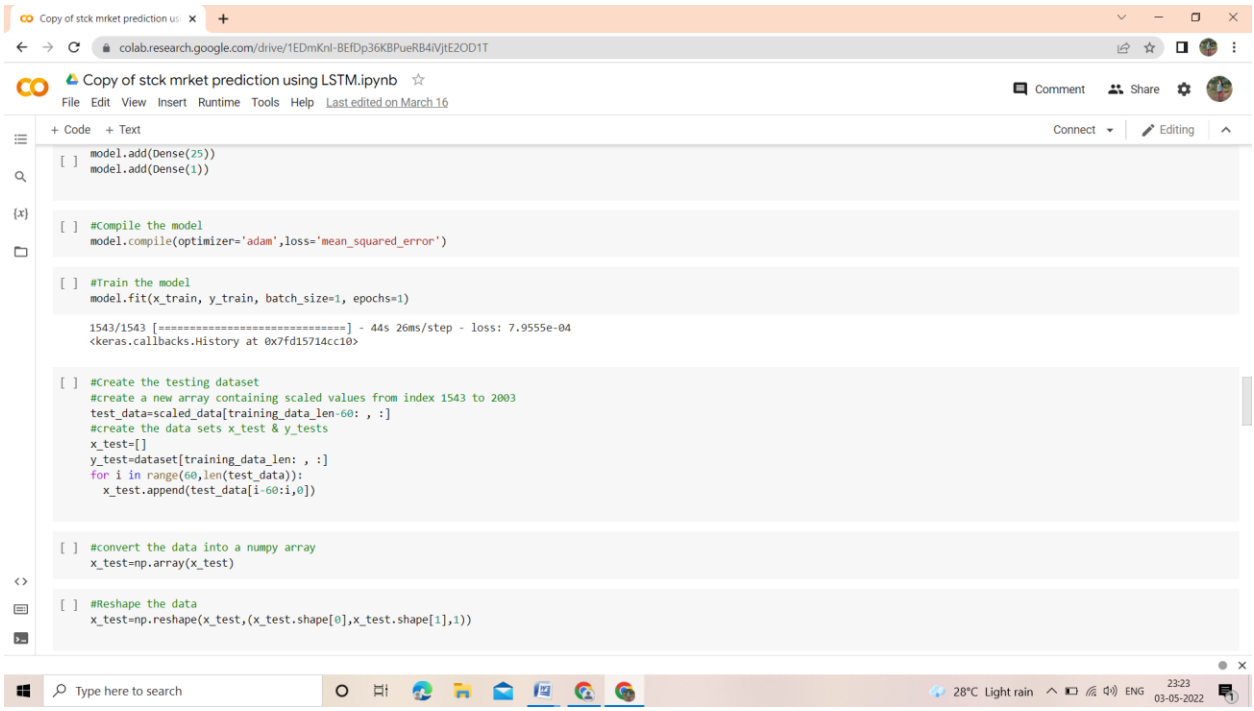
```
[ ] #convert the x_train & y_train into numpy arrays
    x_train,y_train=np.array(x_train),np.array(y_train)
```

```
[ ] #Reshaping the data
    x_train=np.reshape(x_train,(x_train.shape[0],x_train.shape[1],1))
    x_train.shape
```

```
(1543, 60, 1)
```

```
[ ] #Build the LSTM model
    model= Sequential()
    model.add(LSTM(50,return_sequences=True,input_shape=(x_train.shape[1],1)))
    model.add(LSTM(50,return_sequences=False))
    model.add(Dense(25))
    model.add(Dense(1))
```

Type here to search 28°C Light rain ENG 23:23 03-05-2022



Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

Comment Share Connect Editing

+ Code + Text

```
[ ] model.add(Dense(25))
    model.add(Dense(1))
```

```
[ ] #Compile the model
    model.compile(optimizer='adam',loss='mean_squared_error')
```

```
[ ] #Train the model
    model.fit(x_train, y_train, batch_size=1, epochs=1)
```

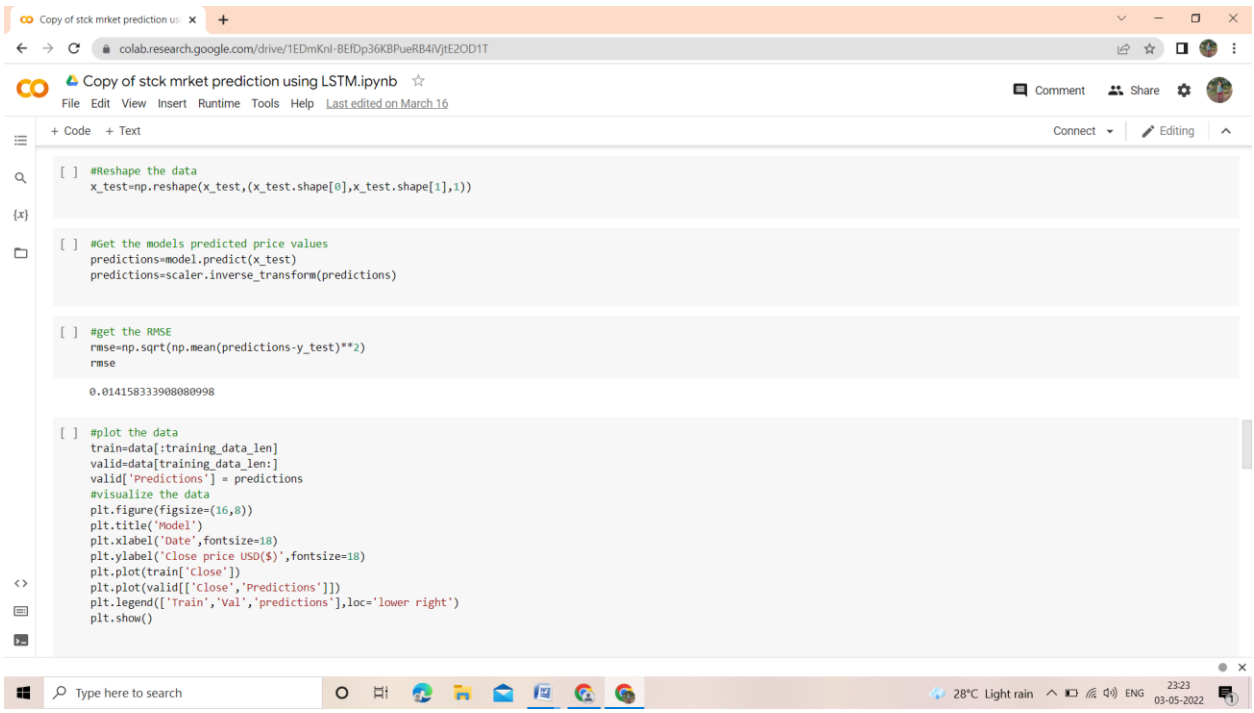
```
1543/1543 [==============================] - 44s 26ms/step - loss: 7.9555e-04
<keras.callbacks.History at 0x7fd15714cc10>
```

```
[ ] #Create the testing dataset
    #create a new array containing scaled values from index 1543 to 2003
    test_data=scaled_data[training_data_len-60: , :]
    #create the data sets x_test & y_tests
    x_test=[]
    y_test=dataset[training_data_len: , :]
    for i in range(60,len(test_data)):
      x_test.append(test_data[i-60:i,0])
```

```
[ ] #convert the data into a numpy array
    x_test=np.array(x_test)
```

```
[ ] #Reshape the data
    x_test=np.reshape(x_test,(x_test.shape[0],x_test.shape[1],1))
```

Type here to search 28°C Light rain ENG 23:23 03-05-2022



Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

Comment Share Connect Editing

+ Code + Text

```
[ ] #Reshape the data
    x_test=np.reshape(x_test,(x_test.shape[0],x_test.shape[1],1))
```

```
[ ] #Get the models predicted price values
    predictions=model.predict(x_test)
    predictions=scaler.inverse_transform(predictions)
```

```
[ ] #get the RMSE
    rmse=np.sqrt(np.mean(predictions-y_test)**2)
    rmse
```

```
0.014158333908080998
```

```
[ ] #plot the data
    train=data[:training_data_len]
    valid=data[training_data_len:]
    valid['Predictions'] = predictions
    #visualize the data
    plt.figure(figsize=(16,8))
    plt.title('Model')
    plt.xlabel('Date',fontsize=18)
    plt.ylabel('Close price USD($)',fontsize=18)
    plt.plot(train['Close'])
    plt.plot(valid[['Close','Predictions']])
    plt.legend(['Train','Val','predictions'],loc='lower right')
    plt.show()
```

Type here to search 28°C Light rain ENG 23:23 03-05-2022

Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

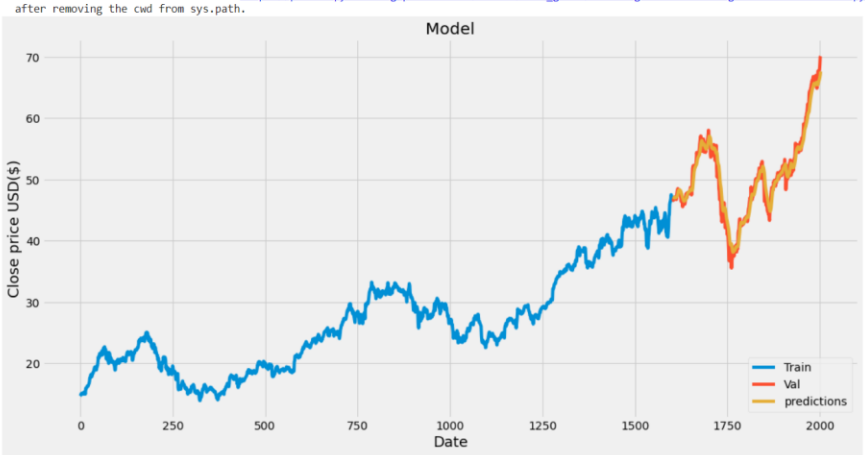
Comment Share

+ Code + Text

Connect Editing

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  after removing the cwd from sys.path.
```



Type here to search

28°C Light rain ENG 23:23 03-05-2022

Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

Comment Share

+ Code + Text

Connect Editing

```
#Show the valid & predicted prices
valid
```

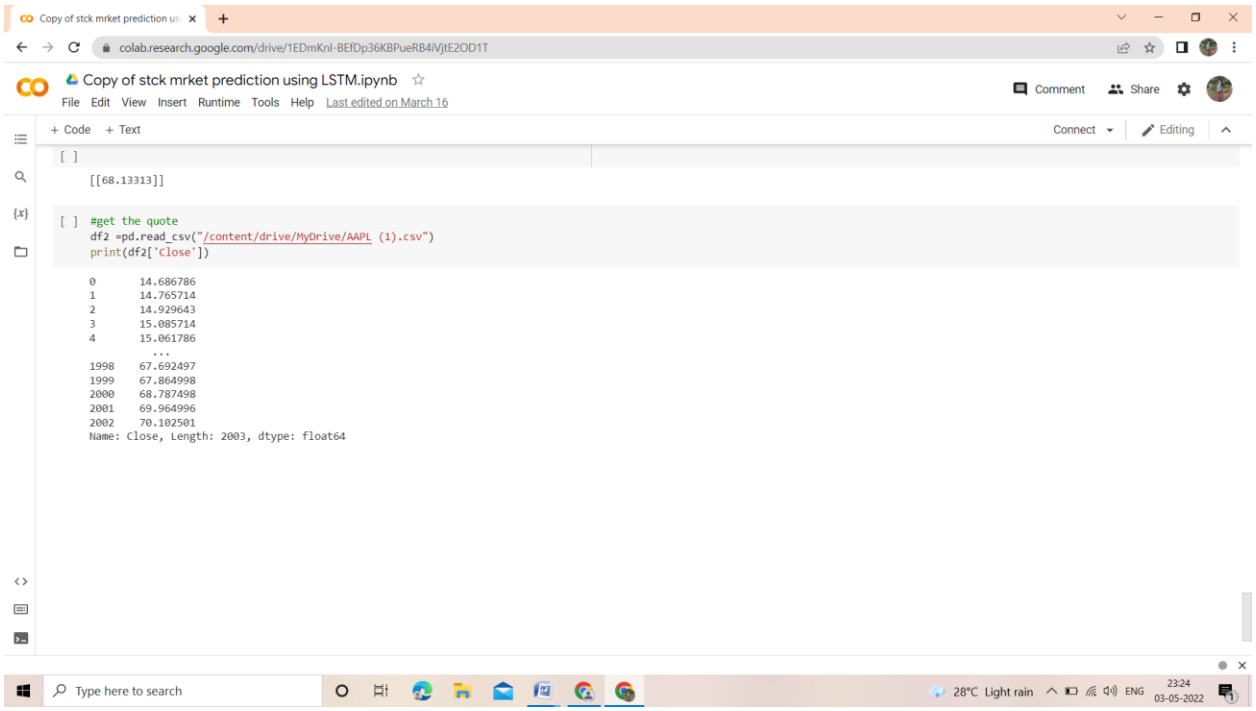
	Close	Predictions
1603	46.747501	47.206738
1604	46.577499	47.166935
1605	46.907501	47.073898
1606	46.790001	47.021233
1607	47.090000	46.976311
...	...	...
1998	67.692497	66.136826
1999	67.864998	66.405334
2000	68.787498	66.678093
2001	69.964996	67.055069
2002	70.102501	67.596741

400 rows × 2 columns

```
#get the quote
df =pd.read_csv("/content/drive/MyDrive/AAPL (1).csv")
#Create a new dataframe
new_df=df.filter(['Close'])
#get the last 60 days closing price values and convert the dataframe to an array
last_60_days=new_df[-60:].values
```

Type here to search

28°C Light rain ENG 23:23 03-05-2022


Copy of stck mrket prediction us
colab.research.google.com/drive/1EDmKnI-BEfDp36KBPueRB4iVjtE2OD1T
Copy of stck mrket prediction using LSTM.ipynb
File Edit View Insert Runtime Tools Help Last edited on March 16
Comment
Share
+ Code + Text
Connect
Editing
[ ]
[[68.13313]]
[ ] #get the quote
df2 =pd.read_csv("/content/drive/MyDrive/AAPL (1).csv")
print(df2['Close'])
0 14.686786
1 14.765714
2 14.929643
3 15.085714
4 15.061786
...
1998 67.692497
1999 67.864998
2000 68.787498
2001 69.964996
2002 70.102501
Name: Close, Length: 2003, dtype: float64
Type here to search
28°C Light rain
ENG
23:24
03-05-2022

Conclusion

This paper proposes RNN based on LSTM built to forecast future values, the result of our model has shown some promising result. The testing result conform that our model is capable of tracing the evolution of opening prices for both assets. For our future work we will try to find the best sets for bout data length and number of training epochs that beater suit our assets and maximize our predictions accuracy.

References

[1] Batres-Estrada, B. (2015). Deep learning for multivariate financial time series.

[2] Emerson, S., Kennedy, R., O'Shea, L., & O'Brien, J. (2019, May). Trends and Applications of Machine Learning in Quantitative Finance. In 8th International Conference on Economics and Finance Research (ICEFR 2019).

[3] Heaton, J. B., Polson, N. G., & Witte, J. H. (2017). Deep learning for finance: deep portfolios. Applied Stochastic Models in Business and Industry, 33(1), 3-12.

[4] Moritz, B., & Zimmermann, T. (2016). Tree-based conditional portfolio sorts: The relation between past and future stock returns. Available at SSRN 2740751.

[5] Olah, C. (2015). Understanding lstm networks–colah's blog. Colah. github. io.

[6] Paiva, F. D., Cardoso, R. T. N., Hanaoka, G. P., & Duarte, W. M. (2018). Decision-Making for Financial Trading: A Fusion Approach of Machine Learning and Portfolio Selection. Expert Systems with Applications.

[7] Patterson J., 2017. Deep Learning: A Practitioner's Approach, O'Reilly Media.

[8] Siami-Namini, S., & Namin, A. S. (2018). Forecasting economics and financial time series: Arima vs. lstm. arXiv preprint arXiv:1803.06386.

[9] Takeuchi, L., & Lee, Y. Y. A. (2013). Applying deep learning to enhance momentum trading strategies in stocks. In Technical Report. Stanford University.

[10] Wang, S., and Y. Luo. 2012. “Signal Processing: The Rise of the Machines.” Deutsche Bank Quantitative Strategy (5 June).