STOCK MARKET DATA ANALYSIS AND PREDICTION

Jahoon Koo
106082022
CSCI 6502 - 001 B
MS in Computer Science
jahoon.Koo@colorado.edu

Jhansi Saketa B V
110168859
CSCI 6502 - 001
MS in Computer Science
JhansiSaketa.BhuvanagiriVenkata@colorado.edu

ABSTRACT

Stocks are unpredictable, it is because of the nature of changing quickly. As technology is enhancing day by day, making lives better. Using modern technology, the stock market patterns are predictable. In this Project, we developed a Stock Market Analysis and Prediction system using the present Google Cloud Platform data tools.

This Project uses the google cloud platform tools like Compute Engine, Pub/Sub, AI Platform, Bucket Storage and Bigquery. There are two data sources used, twitter tweets and stock quotes. Our main motive is to use the streaming data and have hourly predictions. The deep learning model used for our project is the LSTM data model for the prediction. The final results of the prediction can be seen in the BigQuery table in the Google Cloud Platform.

INTRODUCTION

For the businesses to operate, Corporations need funds. To raise the funds, Corporations issue stocks (sell stocks). Basically, stock is a form of security that indicates the holder has proportionate ownership in the issuing corporation. Units of stock are called "shares." Stocks are bought and sold predominantly on stock exchanges. The nature of the Market can be clear, vulnerable and highly unpredictable. Recently, Meta has lost 230 Billion Dollars in a day which is the biggest one day market wipeout in whole History. With this it is clearly visible that Shares or stock value can possibly improve and fall in value very quickly.

It is true that the stock value is unpredictable which is because of its nature of changing quickly. But as technology is enhancing day by day, which is making everything possible. So predicting the stock market value is also possible with the technology. Using technologies, stock market patterns can be studied and analyzed. Using machine learning, deep learning and various technologies one can predict the stock market.

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on an exchange. The successful prediction of a stock's future price could yield significant profit.

To reach the full potential of deep learning in real-life stock market prediction, it needs an infrastructure with the following requirements: 1) reliable and secure environments to handle incoming data from various sources, 2) fast and scalable data transfer to stream stock quotes in real-time, 3) large amount of historical datasets of the stock market, and 4) substantial computing power to train a deep learning model; cloud computing meets all the requirements.

Stock market prediction using cutting-edge technologies like deep learning and cloud computing can be helpful for two category people. One, those who buy the stock from the corporations and other are those who are selling the stock i.e. corporations. For those who are buying the stocks they can know whether they can invest in that particular company and on the other side, the corporations can know their place in the stock market and analyze how the corporation can top in the stock market.

RELATED WORK

1. Stock market using data mining algorithm, in this work the data is first fetched and preprocessed. It is then transformed from high frequency data to a ratio matrix and then the outlier algorithm is performed to find the anomalies in it. The prediction is based on the position of the anomalies and the result is evaluated. The evaluations on real exchange data shows that this method is more efficient in predicting than the other data mining algorithms. [1]
2. Stock Prediction considering historical volatility of stocks, in this work the data is collected from past 8 years and which presents stocks and indexes. After collecting the data, it is arranged and processed. The standard deviation of quarter, 4 period and 8 period is calculated and compared with the 50th index, the stock with greater standard deviation is considered. Based on the standard deviations, quarterly and yearly wise returns on stocks are

predicted. [2]

3. Stock Prediction using Big Data (Cloudera-Hadoop using Machine Learning module of spark), in this work a methodology on cloudera-hadoop based information pipeline is proposed to perform investigations for any scale and any kind of information in which US stocks are examined to foresee every day increases. The information is picked and tested to anticipate the stocks with high day to day pickups utilizing Machine Learning Modules of Spark. [3]
4. Stock Prediction using Big Data Processing tools, machine learning and social media analytics. Using the social media data can improve the accuracy because the data will be the most recent one. The data preprocessing and analysis is done on the social media data using the big data processing tools and machine learning algorithm is performed to predict the stock rates. [4]

PROPOSED WORK

Stock Market Analysis and Prediction using Big Data processing tools, and Deep Learning algorithms. Our project has three main stages:

1. Data streaming
2. Data processing
3. Analysis/Prediction of big data.

In data streaming, we are going to work on two data sources which are real time streaming data. The first data source is social media and the second data source is stock market quotes.

The Data is then going to be pre-processed and the Deep learning model is going to use numerical and text features extracted from data sources to predict the stock market rates.

METHODOLOGY

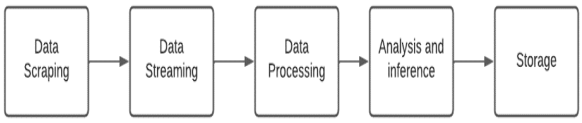

Figure 1.1 Workflow Diagram

Figure 1.1. above depicts the workflow we have designed for our stock market prediction and analysis application. There are five steps present at the general-level of this workflow. There is some degree of variability within this workflow because the specific cloud tools can be varied based on the need of the application and the cost of cloud tools.

The workflow begins with data scraping. This step encapsulates the definition of technologies and methods used for taking in the source data. This includes data scraping from both Twitter and a Finance website. This incoming data will be applied to the deep learning model during the analysis and inference stage. This step requires Cloud VMs to run Twitter API and Finance API scripts to continuously scrape data from sources. Google Cloud Compute Engine is a considerable service to host VMs.

The Data Scraping stage passes forward into the Data Streaming. This stage encapsulates the technologies and methods used for streaming the input data from two data sources to a cloud data processing tool. This stage requires the ability to stream data from multiple sources simultaneously. Also, the scalability and performance of the data streaming tool are important. Google Cloud PubSub is a good candidate for now.

The data streaming stage then flows forward, delivering incoming data into the data processing stage. This stage encapsulates the technologies and methods used for transforming the input data and the data used for training models before the application and deep learning model respectively are able to operate. This stage may need two separate Cloud Data processing systems depending on the complexity of data coming from two sources. Google Cloud Dataflow is a suitable data processing tool for our project.

The data processing stage then flows forward, passing its completed resources into the Analysis and Inference stage. This stage specifies the "brains" of the application. It encapsulates the deep learning trained model information, as well as any supporting technologies needed for the model to operate. Within this stage is where the technical aspects of the application's brain are used. This stage will define deep learning algorithms and other complex internal processes used to predict stock market prices. Since two data sources have distinct types of data (Tweets are textual data, but stock quotes are numerical data), training a model with a combination of textual features and numerical features will keep the project simple and solid.

Finally, the Inference and analysis stage flows into the storage stage. This step stores predicted stock prices from the inference and analysis stage in a Google BigQuery table. Then, the deep learning model will be evaluated based on the accuracy of these predicted stock prices compared to real stock prices.

ARCHITECTURE

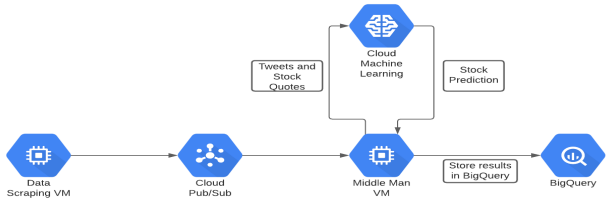

Figure 1.2 Architecture with Data Tools

Figure 1.2. shows the architecture of the project with the data tools. In the first stage, Data Scraping VM will collect the data tweets and stock quotes from the twitter API and Yahoo Finance respectively. The VM will run the program for an hour and collect the twitter tweets for an hour. After an hour it calculates the polarity for all tweets by performing sentiment analysis using the textblob python library. The polarity is then combined with the stock quotes and then in the second stage using Pub/Sub it sends the data to the MiddleMan VM.

The trained deep learning model (LSTM) is deployed in the AI Platform of the Google Cloud Platform. The MiddleMan VM, after receiving the data from the Pub/Sub, predicts the stock rate of the Apple organization using the deployed model in the AI Platform. After the prediction, the data is sent to the Big Query table.

DATA

The Project uses the two main data sources. They are the Twitter API and Yahoo Finance where we get streaming twitter tweets and Stock quotes respectively.

For the twitter API, we created an account in the developer portal of the twitter. In there we created our project and accessed the Keys and Tokens required for the project.

In Figure 1.3, we can see the project created in the twitter developer portal. We had elevated access for this project. Using the keys and tokens, we were able to pull the tweets.

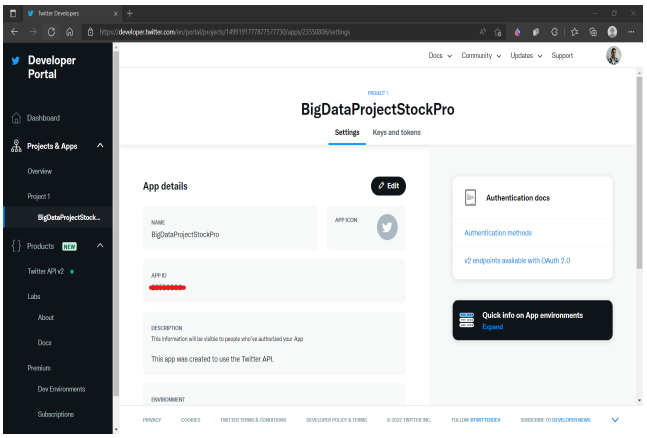

Figure 1.3 Twitter Developer Portal

For the stock quotes, we used yahoo finance to get the recent stock rates for the AAPL (APPLE) organization.

In figure 1.4, we can see the stock quotes for the AAPL (APPLE) organization.

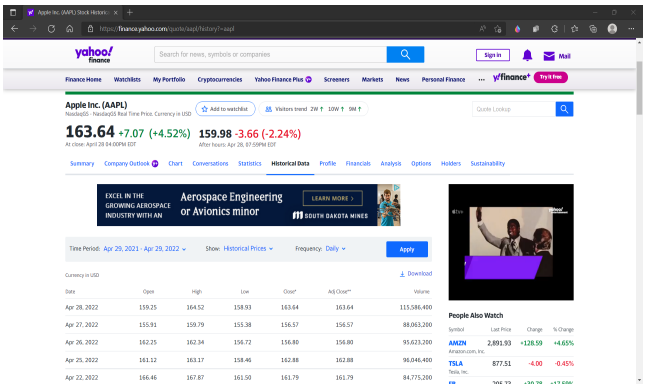

Figure 1.4 Yahoo Finance

TRAINING DATA

For training our model, we used the historical data of the stock quotes and the twitter analysis data.

Figure 1.5 shows the training data of the model used

.

Date	Open	High	Low	Close	Adj Close	ts_polarity
2019-08-26	51.465000	51.797501	51.264999	51.622501	50.650135	0.072340
2019-08-27	51.965000	52.137501	50.882500	51.040001	50.078602	0.117541
2019-08-28	51.025002	51.430000	50.830002	51.382500	50.414650	0.061477
2019-08-29	52.125000	52.330002	51.665001	52.252499	51.268261	0.056460
2019-08-30	52.540001	52.612499	51.799999	52.185001	51.202038	0.106096

Figure 1.5 Training Data

DATA MODEL

The Project uses the Long short-term memory (LSTM) Model for the stock prediction. This model takes the stock quotes and polarity from the twitter tweets and predicts the stock rates for an hour.

LSTM (Long Short Term Memory) is a highly reliable model that considers long term dependencies as well as identifies the necessary information out of the entire available dataset. It is generally used for time-series based analysis such as sentiment analysis, stock market prediction, etc. LSTM is a special category of RNN that possesses the capability to capture long-term dependencies and their selective remembering

property which enables them to focus only on the important parts for prediction.

The LSTM algorithm (figure 1.6) is intended for use in applications where the input is an ordered sequence and information from earlier in the series is relevant. LSTMs are a type of recurrent network in which the output from one step is reused as the input for the next phase, along with the next elements. The internal state of LSTM RNN nodes serves as a working memory space, allowing information to be stored and retrieved over many time steps. Because stock prices are mostly controlled by its past behavior, we believe LSTM was the best model for our stock price prediction project.

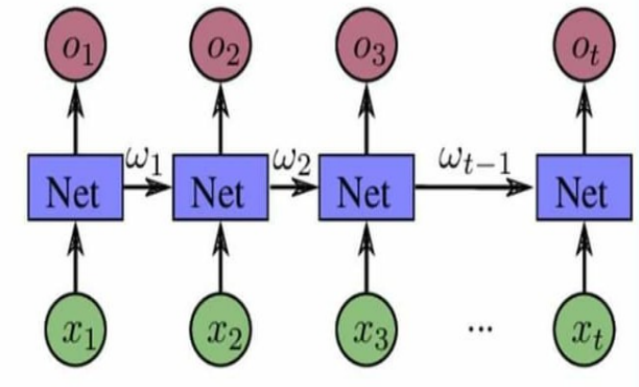

Figure 1.6 LSTM Model

DATA TOOLS

1. Compute Engine:

According to the Google Cloud Documentation [7], "Compute Engine is an Infrastructure-as-a-Service product offering flexible, self-managed virtual machines (VMs) hosted on Google's infrastructure. Compute Engine includes Linux and Windows based VMs running on KVM, local and durable storage options, and a simple REST based API for configuration and control. The service integrates with Google Cloud technologies such as Cloud Storage, App Engine, and BigQuery to extend beyond the basic computational capability to create more complex and sophisticated apps".

Figure 1.7 shows the compute engines we created for this project. We created two VM for the project. One is used for Data Scraping and the other is the MiddleMan VM used for prediction and sending the data to the BigQuery tool.

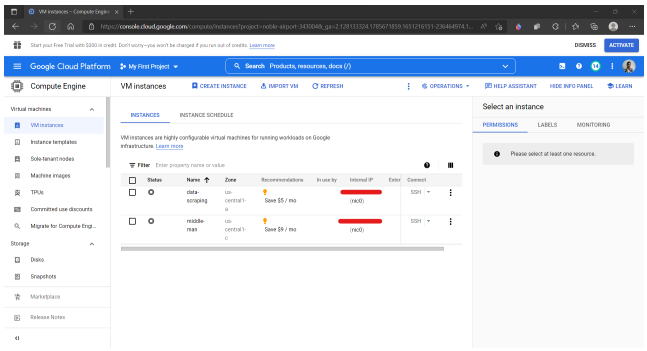

Figure 1.7 Compute Engine

2. Google Cloud Platform Pub/Sub:

According to the Google Cloud documentation [9], "Pub/Sub is used for streaming analytics and data integration pipelines to ingest and distribute data. It is equally effective as a messaging- oriented middleware for service integration or as a queue to parallelize tasks".

Figure 1.8, shows the Pub/Sub that we created for the project.

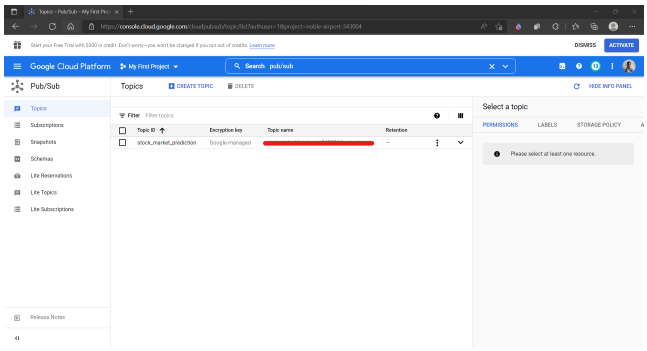

Figure 1.8 Pub/Sub

3. Google Cloud Storage [6]:

The Buckets resource represents a bucket in Cloud Storage. There is a single global namespace shared by all buckets. For more information, see Bucket Name Requirements.

Buckets contain objects which can be accessed by their own methods. In addition to the acl property, buckets contain bucketAccessControls, for use in fine-grained manipulation of an existing bucket's access controls.

Figure 1.9 shows the Bucket of our Project in GCP.

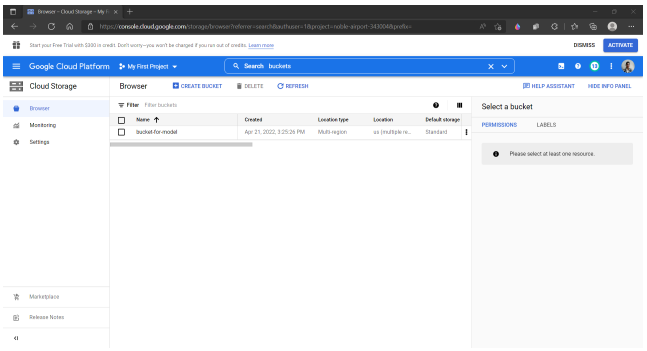

Figure 1.9 Bucket Storage

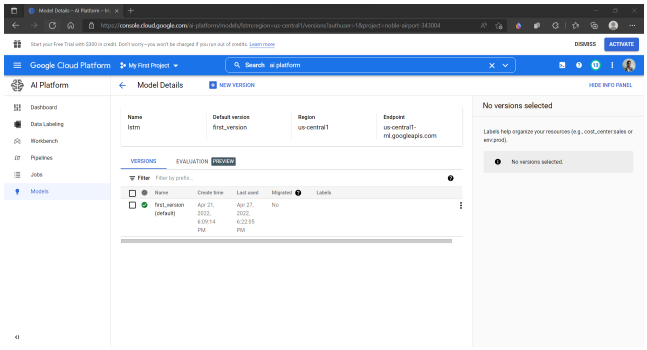

Figure 1.10 GCP AI Platform

4. Google AI Platform:

According to the Google Cloud documentation [8], Google AI Platform is a one-stop solution for machine learning developers and data scientists to take their ML projects from experiment to production and deployment. AI Platform integrated with several easy-to-use tools like BigQuery and Data Labeling Service to help you build and run your own machine learning applications quickly. You can store and manage the large amount of data with BigQuery and then prepare or label this data for model training using Data Labeling Service.

AI Platform also supports Kubeflow to build portable ML pipelines and access to Google's cutting-edge technology like TensorFlow, TPUs, and TFX tools to deploy your AI applications to production.

Figure 1.10 shows the AI Platform created in our GCP Project. We Deployed our trained deep learning model. We deployed our main packages and model files in the Bucket of the Cloud. and we deployed our first version of our LSTM model for the prediction of the stocks of the Apple organization. We used this deployed model in the middleman VM for the prediction and sent the predicted stock rates to the big query model. The deployed AI Platform model uses the json files for the prediction.

5. Google BigQuery:

According to the internet [5], BigQuery is a "fully managed enterprise data warehouse that helps you manage and analyze your data with built-in features like machine learning, geospatial analysis, and business intelligence. BigQuery's serverless architecture lets you use SQL queries to answer your organization's biggest questions with zero infrastructure management. BigQuery's scalable, distributed analysis engine lets you query terabytes in seconds and petabytes in minutes."

Figure 1.11 shows the BigQuery table of our stock market hourly prediction.

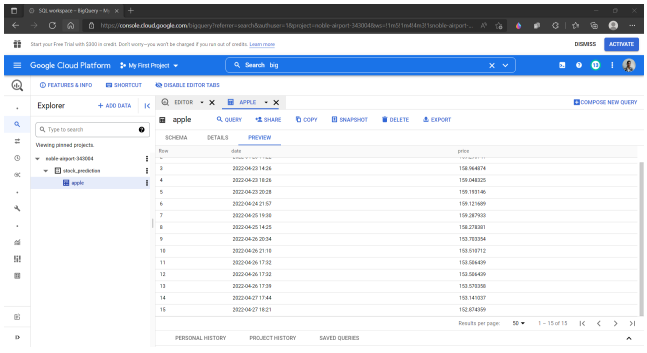

Figure 1.11 BigQuery

CODE

The Below code is for training the model.

1. Importing the libraries

```
#Importing the Libraries
import pandas as pd
import numpy as np
# %matplotlib inline
import matplotlib. pyplot as plt
import matplotlib
from sklearn.preprocessing import MinMaxScaler
from keras.layers import LSTM, Dense, Dropout
from sklearn.model_selection import TimeSeriesSplit
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib. dates as mandates
from sklearn.preprocessing import MinMaxScaler
from sklearn import linear_model
from keras.models import Sequential
from keras.layers import Dense
import keras.backend as K
from keras.callbacks import EarlyStopping
import pickle
from tensorflow.keras.optimizers import Adam
from keras.models import load_model
from keras.layers import LSTM
from keras.utils.vis_utils import plot_model
```

2. Loading the Historical Data for training the model

```
# from google.colab import drive
# drive.mount('/content/drive')
stock_prices_data = pd.read_csv
("/content/stock_prices.csv", index_col= 0,
parse_dates= True)
stock_prices_data.shape
stock_prices_data.head()
stock_prices_data.tail()
twitter_sentiment_analysis_data =
 pd.read_csv("/content/twitter_sentiment.csv",
 index_col= 0, parse_dates= True)
twitter_sentiment_analysis_data.shape
twitter_sentiment_analysis_data.head()
twitter_sentiment_analysis_data.tail()
```

3. Merging the stock quotes and twitter analysis data

```
data=stock_prices_data.merge(twitter_sentiment_analysis
_data,on = "Date")
data.shape
data.head()
data.tail()
```

4. Setting Target Variable and Scaling

```
#Set Target Variable
output_var = pd.DataFrame(data["Adj Close"]).iloc[2:,:]
#Selecting the Features
features = ["Open", "High","Low","Adj Close",
"Close","ts_polarity"]
# adj close value the day after
output_var["Adj Close"]
#Scaling
scaler = MinMaxScaler()
feature_transform = scaler.fit_transform(data[features])
feature_transform= pd.DataFrame(columns=features,
data=feature_transform, index=data.index)
feature_transform = feature_transform.iloc[1:-1 , :]
feature_transform.head()
feature_transform.tail()
```

5. Splitting to Training set and Test set

```
#Splitting to Training set and Test set
timesplit = TimeSeriesSplit(n_splits=10)
for train_index, test_index in
timesplit.split(feature_transform):
    X_train, X_test =
feature_transform[:len(train_index)],
```

```
        feature_transform[len(train_index):
(len(train_index)+len(test_index))]
    y_train, y_test =
output_var[:len(train_index)].values.ravel(),
        output_var[len(train_index):
(len(train_index)+len(test_index))].values.ravel()
X_train.head()
y_train[:5]
```

6. Processing and Building the LSTM Model

```
#Process the data for LSTM
trainX =np.array(X_train)
testX =np.array(X_test)
X_train = trainX.reshape(X_train.shape[0], 1,
X_train.shape[1])
X_test = testX.reshape(X_test.shape[0], 1, X_test.shape[1])
#Building the LSTM Model
lstm = Sequential()
lstm.add(LSTM(32, input_shape=(1, trainX.shape[1]),
  activation='relu', return_sequences=False))
lstm.add(Dense(1))
lstm.compile(loss='mean_squared_error',
optimizer='adam')
plot_model(lstm, show_shapes=True,
show_layer_names=True)
#Train the model
history=lstm.fit(X_train, y_train, epochs=150,
batch_size=8,
   verbose=1, shuffle=False)
y_pred= lstm.predict(X_test)
```

7. Visualizing the Actual and Predicted Stocks

```
#Predicted vs True Adj Close Value – LSTM
plt.plot(y_test, label='True Value')
plt.plot(y_pred, label='LSTM Value')
plt.title("Prediction by LSTM")
plt.xlabel("Time Scale")
plt.ylabel("Scaled USD")
plt.legend()
plt.show()

def training_loss_graph(history):
    plt.plot(history.history['loss'], label = 'Training  Loss')
    plt.legend()
    plt.xlabel("Epochs")
    plt.ylabel('Loss')
    plt.show()
training_loss_graph(history)
from sklearn.metrics import mean_squared_error
from numpy import sqrt
rmse = sqrt(mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error: {}".format(rmse))
```

EVALUATION

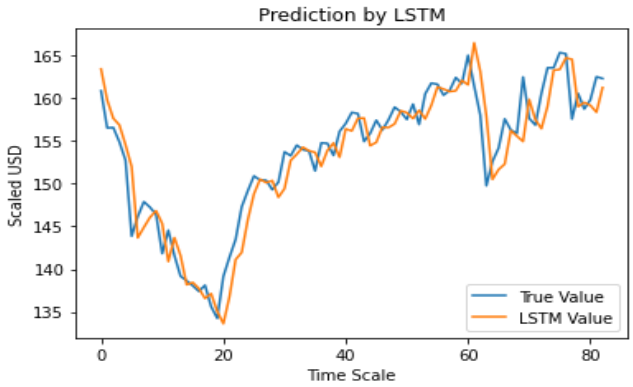

Figure 1.12 Prediction by LSTM

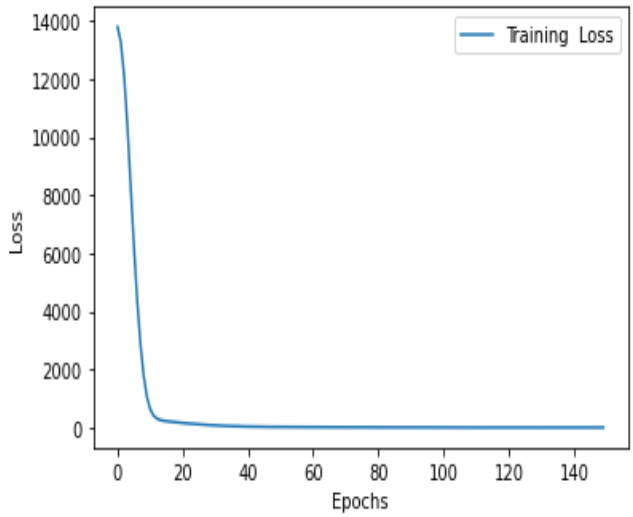

Figure 1.13 Training Loss Graph

We used 150 epochs and 8 batch sizes for the training. The training loss graph (Figure 1.13) shows that as the number of epochs increases, the training loss decreases significantly. When the model was tested against the testing dataset, the prediction by LSTM graph (Figure 1.12) shows how accurate it was. The real stock prices are represented by the blue graph, and the predicted stock prices are represented by the orange graph, which are very similar. This model had a root mean squared error of 2.89.

After the data scraping VM runs the program and collects Twitter tweets for an hour, it uses the textblob python library to perform sentiment analysis and calculates the polarity for all tweets. The polarity and stock quotes are then combined, and the data is sent to the Middle-Man VM via Pub/Sub. After receiving data from the Pub/Sub, the Middle-man VM uses the AI Platform's deployed model to forecast the Apple organization's stock rate. After the prediction, the data is sent to the BigQuery table. The execution times for the tasks above are as follows:

sentiment analysis and polarity calculation: 0.07 secs

Pub/Sub data transfer: 1.24 secs

send data and receive prediction from AI platform: 0.22 secs

insert data into BigQuery: 0.63 secs

This entire procedure took roughly 2 seconds.

DISCUSSIONS

There are major two Challenges that we faced in our projects. They are:

1. Integration of various GCP data tools:

Integrating all the data tools was challenging for our team because we were new to all the data tools that we used in GCP. We had to take lots of time to learn about the particular tool and know how to combine each tool with each other.

2. Limitation of Twitter API request pulls:

We faced another problem in twitter API request pull. We are using elevated access for our project which would allow us to request 50 request pulls per 15 mins.

3. Limitation of outdated data::

We trained and tested our LSTM model with the historical data of Apple stock and Twitter tweets from Jan 2016 to Aug 2019. The historical data was out of scale of current Apple stock quotes. To solve this problem, we had to manipulate the historical data to be in the same scale as current Apple stock quotes. If more resources and time permit, we would train the model with the latest stock quotes of Apple to increase the model accuracy against real-time stock quotes..

The major change that we did to our project is to replace dataflow (GCP data processing tool) with the middle-man VM. Initially we thought of using the dataflow which is the google cloud platform's data processing tool. But while developing our project we did not need data processing because the data that is sent from pub/sub is already clean and required data for our model. So, we did see any use of the dataflow tool in it. So, we replaced the tool with a Virtual Machine called MiddleMan VM.

PROJECT PLANNING

Figure 1.14 shows the milestones of our project. There are 8 milestones for the project to achieve. The project starts from 1st of march and ends on 30th of April.

Date	Milestone
3/1/2022	Project Start
3/8/2022	VM setup with API scripts
3/15/2022	Data Streaming system setup and integrate with VMs
3/29/2022	Data processing tools setup and integrate with data streaming
4/5/2022	Train a deep learning model and deploy on Cloud
4/12/2022	Integrate Cloud ML with data processing
4/19/2022	Storage setup and integrate with Cloud ML
4/26/2022	Testing and Evaluation
4/29/2022	Finalize
4/30/2022	Project End

Figure 1.14 Milestones of the project

CONCLUSION

For businesses to operate, Corporations need funds. Corporations issue stocks to raise funds. Stock is a form of security that indicates the holder has proportionate ownership in the issuing corporation. Stocks are bought and sold predominantly on stock exchanges. Stock exchanges are primarily used to buy and sell stocks. The stock market's character can be apparent, vulnerable, and highly unexpected.

However, as technology advances, predicting the stock market value is becoming possible through the use of cutting-edge technology like deep learning. Deep learning uses historical datasets of the stock market to train algorithms that predict stock price accurately.

To acheive fast and accurate stock market prediction, infrastructure with the following requirements is needed: 1) reliable and secure environments to handle incoming data from various sources, 2) fast and scalable data transfer to stream stock quotes in real-time, 3) large amount of historical datasets of the stock market, and 4) significant computing power to train a deep learning model. Cloud computing satisfies all of the requirements. We have integrated various Google Cloud Platform data tools to create an infrastructure that supports real-time data streaming, data processing, and prediction for stock price. The entire process of data streaming, data processing, and making a prediction took roughly 2 seconds in Google Cloud Platform. With the utilization of the LSTM algorithm, the deep learning model was trained with historical datasets of Apple stock and Twitter polarity and achieved a root mean square error of 2.89. This prediction has room for improvement if more historical datasets of the stock market and Twitter tweets are available.

Stock market prediction using cutting-edge technologies such as deep learning and cloud computing can be beneficial for two types of people. One, those who buy the stock from the corporations and others are those who are selling the stock i.e. corporations. Those purchasing the stocks can determine whether they can invest in that particular company, and the corporations can know their place in the stock market and analyze how the corporation can top in the stock market. A successful prediction of a stock's future price could result in a large profit.

APPENDIX

Honor Code Pledge: "On my honor, as a University of Colorado Boulder student, I have neither given nor received unauthorized assistance."

Work done by individual group members:

Jahoon Koo: Data collection and processing for stock quotes, LSTM model train, and Compute Engine VMs, Cloud Pub/Sub, and BigQuery set ups

Jhansi Saketa B V: Data collection and processing for Twitter tweets, Sentiment analysis for Twitter polarity, Integration of stock quotes and Twitter polarity data, and GCP AI Platform model deployment

REFERENCES

[1] *L. Zhao and L. Wang, "Price Trend Prediction of Stock Market Using Outlier Data Mining Algorithm," in 2015 IEEE Fifth International Conference on Big Data and Cloud Computing, Dalian, China, 2015, pp. 93–98*

[2 *JS. Tiwari, A. Bharadwaj, and S. Gupta, "Stock price prediction using data analytics," in 2017 International Conference on Advances in Computing, Communication and Control (ICAC3), Mumbai, 2017, pp. 1–5*

[3] *Z. Peng, "Stocks Analysis and Prediction Using Big Data Analytics," in 2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), Changsha, China, 2019, pp. 309–312*

[4] *G. V. Attigeri, Manohara Pai M M, R. M. Pai, and A. Nayak, "Stock market prediction: A big data approach," in TENCON 2015 - 2015 IEEE Region 10 Conference, Macao, 2015, pp. 1–5.*

[5] *Google. (n.d.). What is BigQuery? | google cloud. Google. Retrieved April 30, 2022, from https://cloud.google.com/bigquery/docs/introduction#:~:text=BigQuery%20is%20a%20fully%20managed,geospatial%20analysis%2C%20and%20business%20intelligence.*

[6] *Google. (n.d.). Cloud storage | google cloud. Google. Retrieved April 29, 2022, from https://cloud.google.com/storage*

[7] *Google. (n.d.). Compute Engine: Virtual Machines (VMS) | google cloud. Google. Retrieved April 29, 2022, from https://cloud.google.com/compute*

[8] *Google. (n.d.). Data Science and Machine Learning on Cloud Ai Platform | google developers. Google. Retrieved April 30, 2022, from https://developers.google.com/learn/topics/datascience*

[9] *Google. (n.d.). Pub/Sub for Application & Data Integration | google cloud. Google. Retrieved April 29, 2022, from https://cloud.google.com/pubsub*