**Stock Forecasting**

**Without and with Sentiment Analysis**

Introduction

A stock (also known as equity) is a security that represents the ownership of a fraction of a corporation. This entitles the owner of the stock to a proportion of the corporation's assets and profits equal to how much stock they own.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.

In this project we use variations of RNN(Recurrent Neural Network) i.e. LSTM(Long Short Term Memory) Model and GRU(Gated Recurrent Unit) Model ,CNN(Convolution Neural Network) and Vanilla Neural Network to compare their results and predictions.

In the second part of the project we use Sentiment analysis on a dataset of news articles about a specific company and generate the data corresponding to the sentiment/emotion for the company on each specific day.We couple this data along with the historical stock prices data for that company to predict the trend of stocks for that company.

Literature Review

Work Approach

1. Historical Data Collection

We use **Yahoo Finance** for collecting the historical stock price data which will be used in all of the models as a means to record and observe the previous trends.The market is a lot affected by the crowd psychology. The demand and supply factors which in turn affect the price are affected by the psychology of the market. And it's believed that these psychological patterns recur over time. Thus the price also tends to move in patterns. With the help of data analysis one can  understand these price movements and be on the correct side of the market most of the time. Using technical analysis or charting helps you find out the odds in which price could probably move.

Sample data for Google(GOOG):-



1. Sentiment Analysis Data Generation & Processing

A. News Article Generation

Web Scraper made in Python is used to get the list of links of news articles.This uses the python module “scrapy”.We use the website “reuters.com” for this purpose.The role of this file is to scrap out a list of news articles from the website about a specific company mentioned by us date wise and output it in a csv file with 2 columns date and url of news article.

Below is a subset of articles extracted for Apple company from reuters:-

“PICTURE”

1. Sentiment Score Generation

The output csv file from Step A is used as an input in this step. Here we use “Rossete” API for text analysis for the contents for each article. This file outputs a csv file which contains the columns date, label,confidence,entity-label and entity-confidence. The column label can take 3 values:-

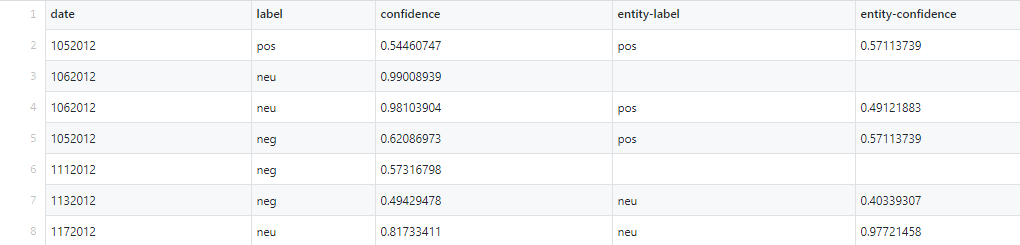
1. pos:-expressing the article is positive for company and its share
2. neg:- expressing the article is negative for company and its share
3. neu:- expressing the article is neutral for company and its share

The confidence colomn presents the probability/confidence with which the label is decided for that specific date

“ENTITY\_LABEL”

“ENTITY\_CONFIDENCE”

Below is a subset of sentiment scores generated for Apple company news articles:-



**Introduction**

Recently many Research papers have come forward to predicting stock market using sentiment analysis.

1. Calculating Bullishness and Sorting it in Chronological Order

Using the sentiment score calculated for each article in the previous step, We calculate the bullishness score of the selected stock on per day basis. Bullishness for each day is calculated as:

Bullishness score = (Sum of positive confidence – sum of Negative confidence)/(total articles)

In the next file, we arrange these scores in a chronological order.

“Picture”

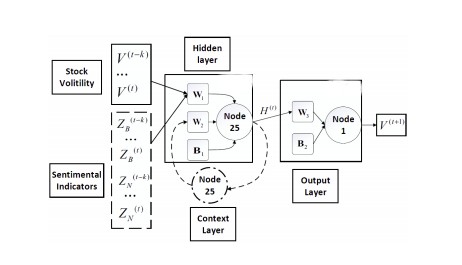
D. Combining sentiment Analysis with Stock Prices

For each day, combine the sentiment analysis of that day with the corresponding stock price. In case, the sentiment score of a day is not available, mark the sentiment score as the score of latest priovious day available. The output file after this step contains sentiment score and stock price for each day.

E. RNN Model

For training our model, the sentiment score goes z score normalization followed by min-max normalization. Then, we construct a 2 layer LSTM followed by 2 Dense layers. Input of which will be previous k days stock price and it’s corresponding sentiment score where k is the window size. The output is the prediction of stock price. Hyperparameters of this model are as follows batch\_size=128,

epochs=200, validation\_split=0, verbose=0)



F. Results

G. Conclusion

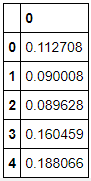
After completing the prediction, we can clearly observe that the results using sentiment analysis offers marginally better results as compared to predicting without them. Hence, we conclude that analyzing sentiments in order to predict the volatility in stock market is not worth the effort.

1. Historical Stock Price Data Processing
2. Normalization

The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. For machine learning, every dataset does not require normalization. It is required only when features have different ranges.

So we perform MinMaxScalar Normalization across all columns. MinMaxScalar scales all the data features in the range [0, 1] or else in the range [-1, 1] if there are negative values in the dataset.

Below is the subset of open price column normalized:-



1. Train Test Split

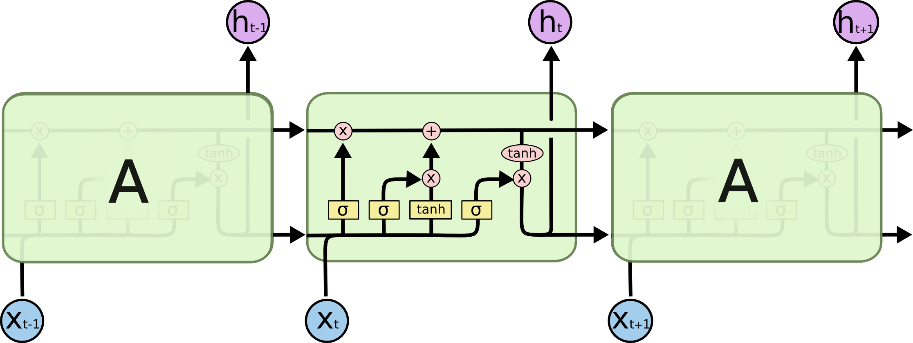
We take the stock prices of the past 250 days from Yahoo Finance for a particular company, first 220 of which are used in training and the last 30 day data is used for testing.

1. Model Creation based on only Historical Stock Data
2. RNN Based
3. Long Short Term Memory(LSTM) Model

Long Short-Term Memory (LSTM) is a specific recurrent neural network (RNN) architecture that was designed to model temporal sequences and their long-range dependencies more accurately than conventional RNNs.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior.

Basic Cell of LSTM:-



As the cell state goes on its journey, information get’s added or removed to the cell state via gates. The gates are different neural networks that decide which information is allowed on the cell state. The gates can learn what information is relevant to keep or forget during training.

Variations:-

1. Bidirectional LSTM: - Bi-LSTMs train two LSTMs instead of one LSTMs on the input sequence. The first on the input sequence as-is and the other on a reversed copy of the input sequence. Can be trained using all available input info in the past and future of a particular time-step.
2. 2 Path LSTM:-
3. Gated Recurrent Unit (GRU)   
     
   GRU is like a long short-term memory (LSTM) with a forget gate, but has fewer parameters than LSTM, as it lacks an output gate. GRUs have been shown to exhibit better performance on certain smaller and less frequent datasets.

Variations:-

1. Bidirectional GRU: - It allows for the use of information from both previous time steps and later time steps to make predictions about the current state.
2. 2 Path GRU:-
3. CNN Based

CNNs are powerful image processing, artificial intelligence (AI) that use deep learning to perform both generative and descriptive tasks, often using machine vison that includes image and video recognition, along with recommender systems and natural language processing

We use the “keras” and “tensorflow” inbuilt functions to construct the model

We use three main types of layers to build ConvNet architectures:

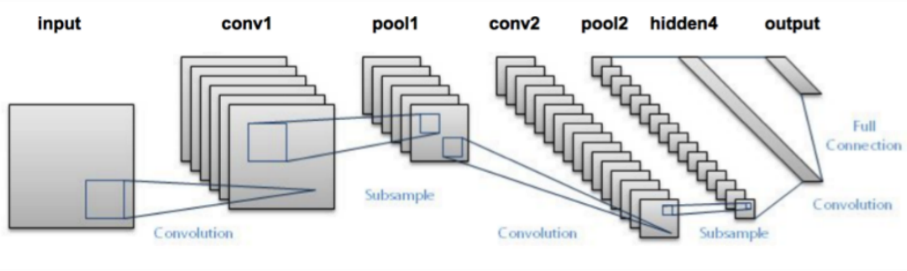
-Convolutional Layer: - Simple application of a filter to an input that results in an activation

-Pooling Layer: - Pooling layers are generally used to reduce the size of the inputs and hence speed up the computation.

-Fully-Connected Layer:-exactly as seen in regular Neural Networks

We will stack these layers to form a full ConvNet architecture.

Sample CNN model:-



Experimental Steps

1. Model Training
2. After normalization of data we take the 4the column of the dataset i.e. “Close” indicating the closing stock price data for each specific date.We will use only this stock price across all models to maintain uniformity.
3. Out of the 252 data points 222 are used for training and the most recent 30 days are used for testing during the train test split.
4. The 222 data points are divided into batch size of 5 stored in the variable batch\_X and sent as a placeholder for X in the model contructed using the feed\_dict object.batch\_Y is also formed of 5 data points but the data starts from the position 1 ahead of starting position of that of batch\_X.These are also sent to the model object and is used in the calculation of loss function.



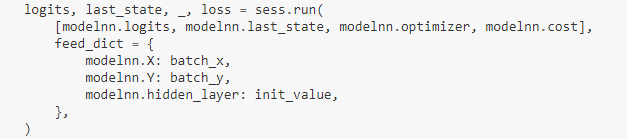
Where,

cost=loss function (mean of squared errors),

Y=batch\_Y(containing real values),

logits=containing the predicted values using the batch\_X

1. The model is run on the batch\_X and batch\_Y values and other inputs given and it gives 4 outputs variables/vectors



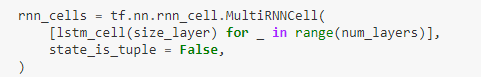
Where,

modelnn.logits= containing the predicted values using the batch\_X,

modelnn.last\_state=containing outputs for each data point from last layer only

modelnn.cost=containing the loss function calculated above

1. The outputs obtained in the last\_state are set to the initial state variable and supplied to the model for the next 5 data points as it contains the memory/important information obtained from the previous 5 data points training.
2. Model Testing
3. ?????
4. Model Construction
5. LSTM:-



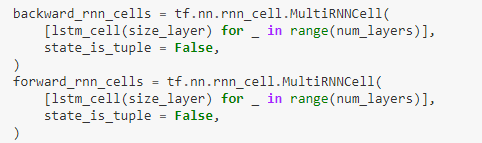
Creates a collection of LSTM cells for each of the num\_layers number of layers.Here num\_layers=1,so it is a single layer structure interms of layers of lstm cells.

The function lstm\_cell(size\_layer) creates a single lstm cell with size\_layer number of lstm basic blocks/memory units .The function is defined as follows:-



Uses the inbuilt Tensorflow function LSTMCell for a lstm cell construction.Here the lstm cell is constructed with 128 units.

1. LSTM Bidirectional:-



Creates a collection of LSTM cells each for forward path and backward path.Here num\_layers =1,so it’s a single layer structure for bothe directions.Now apart from storing info from past data we also take info from the future data and dependencies generated by the LSTM path in backward\_rnn\_cells,thus improving our prediction.

The function lstm\_cell(size\_layer) creates a single lstm cell with size\_layer number of lstm basic blocks/memory units .The function is defined as follows:-



Uses the inbuilt Tensorflow function LSTMCell for a lstm cell construction.Here the lstm cell is constructed with 128 units.

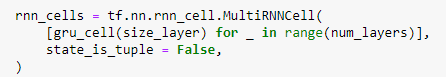
1. LSTM 2 Path:-
2. GRU:-

Similar to LSTM model,just in place of using a LSTMCell function we use the GRUCell function of tensorflow



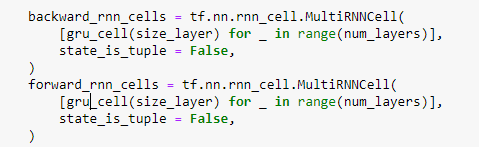
size\_layer=128 that is it has 128 GRU basic units taken here.

These cells are used to make a layer using the MultiRNNCell function as in above:-



Here also num\_layers =1 and gru\_cell function defined above is used for the contents of the layer

1. GRU Bidirectional:-



Like Bi-LSTM,This Creates a collection of GRU cells each for forward path and backward path.Here num\_layers =1,so it’s a single layer structure for both directions.Now apart from storing info from past data we also take info from the future data and dependencies generated by the GRU path in backward\_rnn\_cells,thus improving our prediction.

The function gru\_cell(size\_layer) creates a single lstm cell with size\_layer number of lstm basic blocks/memory units .The function is defined as follows:-

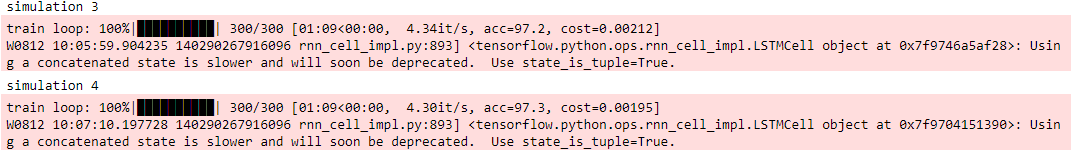


Similar to LSTM model,just in place of using a LSTMCell function we use the GRUCell function of tensorflow

size\_layer=128 that is it has 128 GRU basic units taken here.

1. GRU 2 Path:-
2. Running Simulations

We run 10 simulations of the model to ensure the perfect training and testing accuracies.Each simulation trains over the training data for epoch(10) times i.e. we train over the data 100 times and we calculate the accuracy and cost side by side in each simulation using the functions of the tqdm library.Here’s one of the simulation runs:-

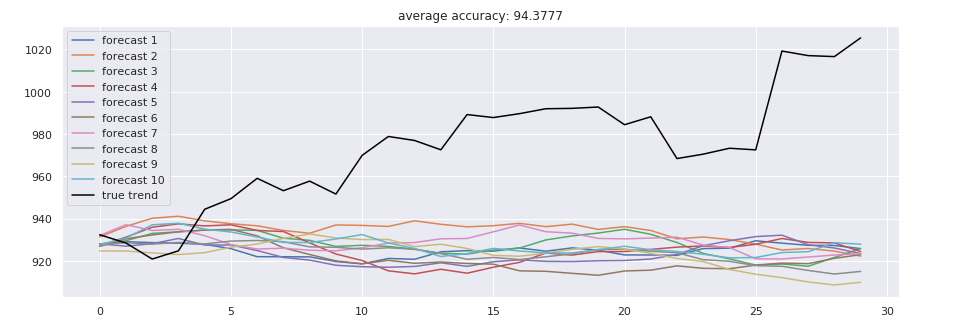


This shows the run of simulation 3 & 4 and shows their respective costs and accuracies.

1. Graph Plots

”results” vector conatins the result of all the 10 trainings/forecasts for the 10 epochs .We plot each of the forecast on the same graph and we also plot the actual true trend plot on the same graph colored black .This graph can be used to check where the model has undergone overfitting and where its still underfit and thus also determine a perfect forcast which most closely matches the true trend.

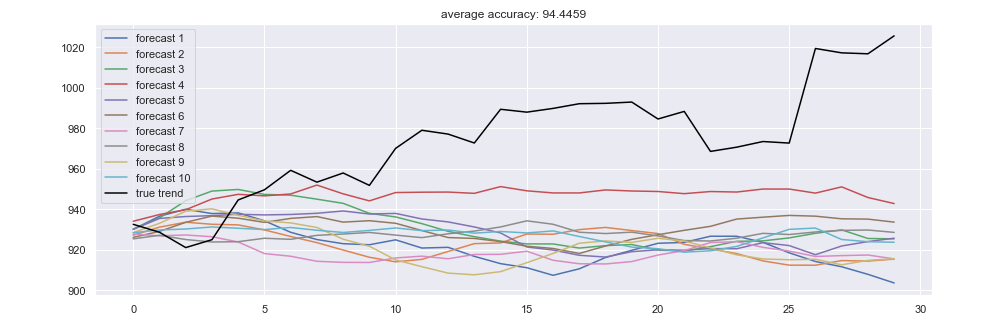
Here is the graph for LSTM forcasts:-



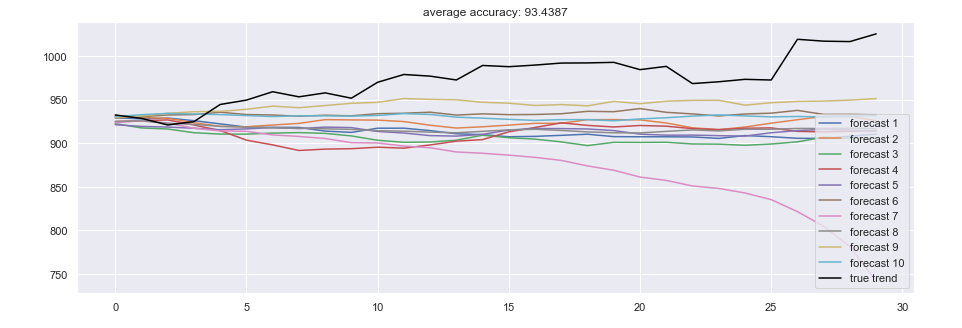
Here black line is the true trend and others are forecast trends

Results

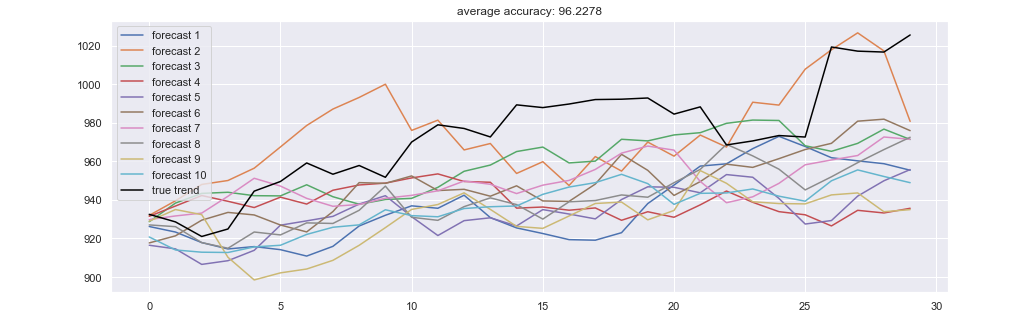
LSTM:-



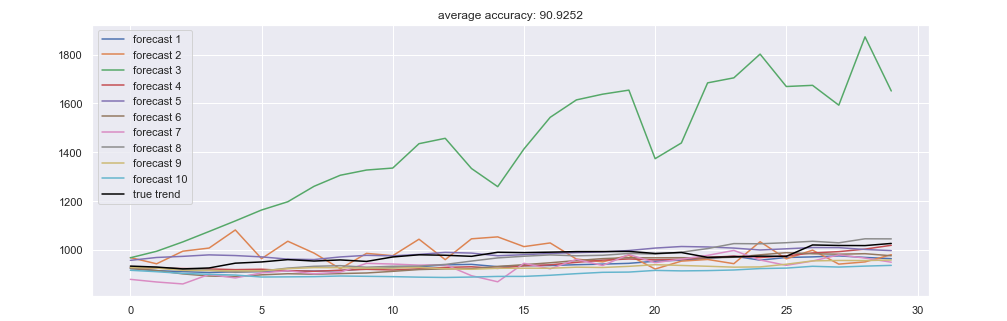
Bi-LSTM:-



GRU:-



Bi-GRU:-



After simulating the above all the models here are the results in a tabular form:-

|  |  |  |
| --- | --- | --- |
| MODEL NAME | DESCRIPTION/FEATURES | ACCURACY |
|  |  |  |
| Long Short Term Memory Model(LSTM) |  |  |
| Long Short Term Memory Model(LSTM) Bidirectional |  |  |
| Long Short Term Memory Model(LSTM) 2 Path |  |  |
| Gated Recurrent Unit(GRU) |  |  |
| Gated Recurrent Unit(GRU) Bidirectional |  |  |
| Gated Recurrent Unit(GRU) 2 Path |  |  |