**Forecasting of stock Prices using ARIMA, RNN, LSTM**

**Team: *Quad Squad***

Sri Nikhitha Boddapati - 16322565

Sukumar Bodapati - 16326105

Pavan Kumar Jonnadula - 16324822

Vamsi Alapaty - 18230326

**Introduction:**

In today’s world, we have multiple models to forecast the stock market.

* Out of all the techniques we have, which strategies are the most effective?
* Is it better to try to predict stock prices or to trade strategies based on stock price movement?
* Can we predict and tell when we can buy a stock using these models?
* What parameters can we use for various situations?

**DataSet:**

In this project, we are using Amazon Stock data.

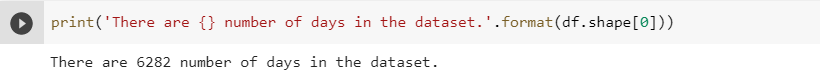
**Data Exploration:**

**Graphical user interface, text, application

Description automatically generated**

As we are dealing with the prediction, if we examine the closing prices and dates. The goal of time series analysis is to locate closing prices. The reason for this is that closing prices, as opposed to opening or average pricing, more accurately reflect how business was doing. So, consider the Close data for prediction.

Let’s have a look at how many days we have in this dataset and visualize the data to consider training and testing samples.



In this dataset, it has 6282 number of days in this dataset.

**Visualization:**

**Chart, line chart

Description automatically generated**

From the above graphical observations, most of the Stock price from 2015 is exponential growing. Amazon’s business really took off after 2010 and peaked around 2015, therefore the testing will be very interesting. For training, normalize the training data so that related data points across time are condensed to a single data point, allowing the model to train in a way that anticipates exponential growth beyond 2015.

**Feature Extraction**:

Designed the below indicators for the feature Extraction:

* **Momentum:** This indicator will be useful in predicting future stock prices as well as determining whether we should acquire the stock. Momentum is possibly the simplest and most straightforward oscillator (financial analysis instrument) to comprehend and apply. It is the measurement of the rate of change in price movement for a certain item, as well as the speed or velocity of price movements.

**Momentum(m)=Latest Price -Closing price (at Number of days ago)**

* **Bollinger bands:** For Quantitative Analysis of Stock market, this indicator is useful. These are used to characterize the trading range of a financials’ by defining the current high and low values in a market. They are measure of volatility and a Moving Average (MA) line, an upper band, and a lower band make up the bands. MA simply adds and subtracts standard deviation to get the upper and lower bands.
* **EMA:** The exponential moving average (EMA) is a better variant of the simple moving average (SMA). Moving averages simply average out the data over a period, allowing us to see how the company's closing price has changed over time. For example, if the price was 32,33,45,1 for four days (the company downs on the fourth day), the average would be 32. Now, because 32 is a lower-than-average number, it suggests that 45 was a fluke and that the company was always losing money.

A picture containing text

Description automatically generated

For Generating Technical Indicators:

Text

Description automatically generated

Calculated indicators:

Graphical user interface, application

Description automatically generated

**Visualizing the indicators:**

**Chart, histogram

Description automatically generated**

The above graph indicates the technical indicators that we designed for the last 100 days.

**Chart

Description automatically generated**

The threshold between MACD and momentum is displayed in this graph. As can be seen, momentum gives the MACD an average value between the peak values and the highest or lowest values. MACD is dependent on the above calculated moving average features.

**ARIMA Model:**

One of the most prominent strategies for predicting future values of time series data was Autoregressive Integrated Moving Average (ARIMA) (in the pre-neural networks ages). Let's try it out and see whether it turns out to be a useful predictive feature.

**Key aspects of this model:**

* *‘AR’* stands for autoregression. The dependent relationship between an observation and a set of lagged observations is used in this model.
* *‘I’* stand for "integrated." To make the time series steady, differencing raw observations (e.g. subtracting an observation from an observation from the preceding time step) is used.
* *‘MA’* stands for Moving Average. A model that takes advantage of the relationship between an observation and the residual error from a moving average model when applied to lagged observations.

**Parameters used in this model:**

* *‘p’*: The lag order, or the number of lag observations incorporated in the model.
* *‘d’*: The degree of differencing is the number of times the raw observations are differenced.
* *‘q’*: The order of moving average, also known as the size of the moving average window.

Text

Description automatically generated

Define the model and set the lag value to 5 for regression and used difference order of 1 to make stationary time series and moving average model to zero.

**Model Results:**

**Table

Description automatically generated**

**Summary of the model:**

**Chart, line chart

Description automatically generated**

* We found that 5 was a decent starting point for the AR parameter of the model.
* Except for the last two, most P-values are greater than 0.05, according to the ARIMA summary. The model working is pretty good.
* In the statistics of the model the difference between Bayesian information criterion and Akaike information criterion is very low which depicts that this model is good.
* In the autocorrelation below, we can see that there is a positive correlation with the first 0-to-500 lags, which is possibly significant for the first 250 lags.

**Visualization of ARIMA model:**

Chart, line chart

Description automatically generated

If we observe the above graph, the model is very good. The real and predicted values are closer.

**LSTM Model:**

The LSTM algorithm excels in forecasting stock market data. We'll first try to forecast closing prices with only one feature, Open (which has the strongest connection to closing prices), and then with many features (using some form of one-hot encoding) to see what we can come up with.

**Cleaned the dataset and count the features, samples**:

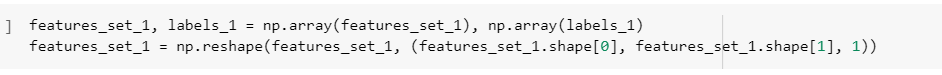


**Normalized the data:**

Text

Description automatically generated

**Creating label and feature set:**



**LSTM Model:**

**Text

Description automatically generated**

Used 100 epochs of Open training data to try to predict Open. Because this is more of a regression concern, utilized MSE and mean absolute error instead of accuracy. Because there was some overfitting, there we need to normalize the data.

****

*MAE was****: 0.1567***

This suggests that for all 2265 datapoints, the average difference between input and output is 0.1567.

**Visualization of the model:**

**Chart, bar chart

Description automatically generated**

If we observe, this model is not so great.

**LSTM with multiple features:**

In the previous, we have considered only one feature. In this we will try to predict using multiple features.

Table

Description automatically generated

The above data is used to train the model.

**Model:**

**Graphical user interface, text, application

Description automatically generated**

With one feature, the mean absolute error is smaller than the previous model. The error is just under 0.0026. As a result, the training model should be quite like the testing model.

*Prediction:*

Chart, line chart

Description automatically generated

*Predicting the next day values:*

Text, letter

Description automatically generated

**LSTM (Vanilla) Model:**

We merely normalized the closing prices before splitting it into train and test datasets. We basically used a simple lookback window to give all similar data the same movement (movement is simply the same kind of normalization applied to the data points) and input them into price points. Have changed the hyper parameters from the past mode to give the best output.

Chart

Description automatically generated*Data visualization for close values:*

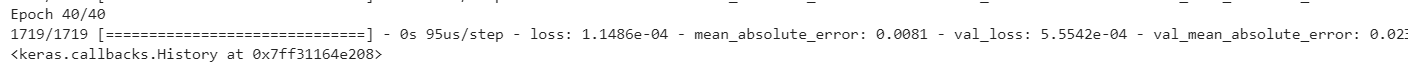
Normalize the data so that prices are reduced to normalized values, and instead of actual prices, we can forecast stock movement.

**Model:**

Text

Description automatically generated with medium confidence

**Error:**



**Visualization of the model:**

Chart, histogram

Description automatically generated

When we normalize the prices and estimate the stock price movement, LSTM works wonderfully for predicting Closing Prices.

Graphical user interface, text

Description automatically generated with medium confidence

The error is too low i.e., ***0.000175.***

**Conclusion:**

In this project, used ARIMA, LSTM, LSTM with multiple features to forecast multiple models of the time series OHLCV data, perform feature extraction, hyper parameter tweaking, and train on ARIMA, LSTM. Based on the above conclusions, LSTM works well.

From this model we can assume that the next day value as **119.30**

**Future work:**

* Stock prediction can be done using GAN. We tried to implement using TimeGAN but due to complex architecture and time constraints, we couldn’t complete it.
* We can predict the next day values for different datasets from Day 1 to till today.
* This model can be integrated into any web application where we can show the predict of next day stock value for all the trending stocks (Creating the stock market website).