# Timeseries Analysis for Stock Price Prediction Using ARIMA

**Ranga Chari Vinjamuri • Vinitha Ravichandran**


**Abstract** Stock prices are highly volatile due to the chaotic market and has always been an attractive topic to investors and researchers. This paper proposes the use of ARIMA model to analyze stock data which is a timeseries data and to forecast the Close price for the stocks. The initial implementation is to be done in Python, which shall then be run in Spark clusters in Google Data Procs to handle bigdata and improve computational speed. Stock prices being highly volatile and very difficult to predict, needs data transformation before fitting into the model. A series of data transformation is done to prepare the data for prediction. To compare the accuracy of ARIMA model, an implementation of deep learning technique, LSTM is done in TensorFlow for the same data and the scores are compared to select the best model for Stock Price Prediction.

**Keywords:** ARIMA, Google Data Procs, LSTM, Python, Spark, Stock Price Prediction, Time Series Analysis

**1. Introduction** Time series data is a series of data points indexed in time. Stock market prices belong to the category of timeseries data as it contains the change in prices of the stock over a period in specific intervals (daily, weekly, monthly, yearly). Time series data needs to be handled and analyzed in a different way compared to other data as the trend in changes plays a significant role in prediction. Time series analysis extracts statistically meaningful characteristics of data and generate other useful insights applied in business situation. Time series analysis helps understand time-based patterns of a set of metric data points which is critical for any business. Time series data has different components which needs to be taken into consideration: Seasonality or seasonal variations are those that repeat over a specific period such as day, week, month, season, etc; Trend variations is an upward or downward shifts in dataset in a predictable pattern over a period of time; Cyclical variations that correspond with business or economic 'boom-bust' cycles or follow their own peculiar cycles; Random variations that do not fall under any of the above three classifications. Machine Learning is one of the powerful tools to estimate stock prices as it eliminates human rational limitations.

## 2.Methodology

**2.1 Block Diagram** Figure 2.1 shows the complete block of implementation that can be deployed in Jupyter notebook, Databricks Spark Cluster, Google Cloud VM instance.

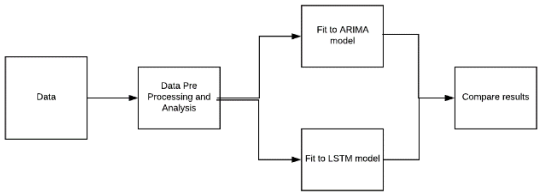


Figure 2.1: Block Diagram

**2.2 Data Acquisition** In this project, the Stock prices of Amazon is used. The historical dataset is obtained real time from Yahoo Finance by stating start and end dates as today's date and the date obtained by subtracting 3 years from today respectively. The dataset contains the following features: Date, High, Low, Open, Close, Adjusted Close, Volume. Figure 1 shows the trend of Amazon daily stock prices over 3 years.

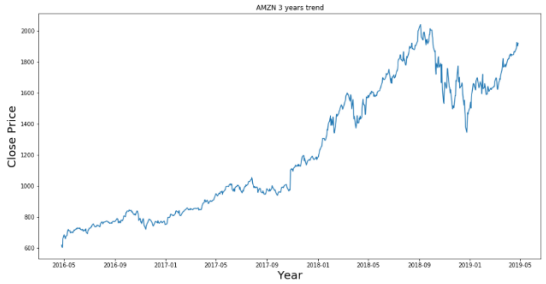


Figure 2.2: Trend of Amazon Stocks over 3 years

**2.3 Data Analysis** Stock price data needs to be treated and transformed before fitting to the prediction models. Time series data needs to be stationary which means that the statistical properties of a process generating a time series do not change over time. It does not mean that the series does not change over time, just that the way it changes does not itself change over time. The algebraic equivalent is thus a linear function, perhaps, and not a constant one; the value of a linear function changes as $x$ grows, but the way it changes remains constant. Stationary Time Series data does not have any upward or downward trend or seasonal effects. Mean or variance are consistent over time. We are using a statistical method called Dickey-Fuller test to check if our time series is stationary or not. As in Figure 2.1, the original data is non-stationary as the rolling mean and standard deviation vary with time. After applying log transform and removing seasonality by differencing, we get a stationary time series data as in Figure 2.2

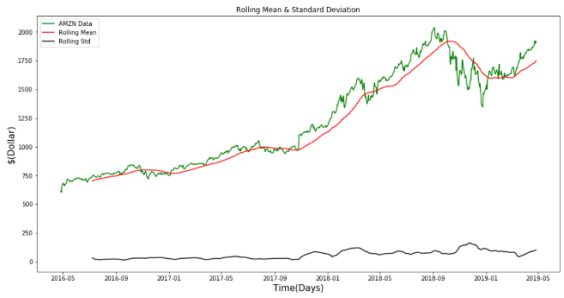


Figure 1.3: Non-Stationary Time series

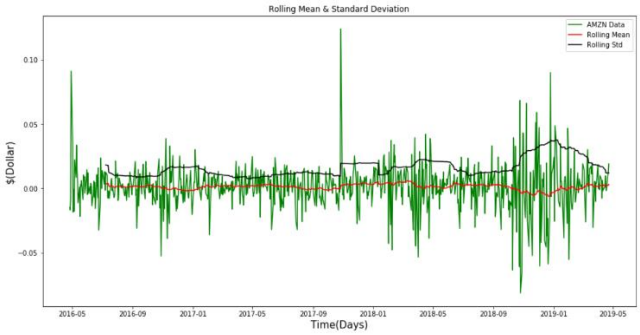


Figure 2.4: Stationary Time Series

**2.4 ARIMA Model** An autoregressive integrated moving average model is a form of regression analysis that gauges the strength of one dependent variable relative to other changing variables. The model's goal is to predict future securities or financial market moves by examining the differences between values in the series instead of through actual values.

**2.5 Hyperparameters of ARIMA Model** The components of ARIMA model are

1. Autoregression (AR) refers to a model that shows a changing variable that regresses on its own lagged, or prior, values
2. Integrated (I) represents the differencing of raw observations to allow for the time series to become stationary, i.e., data values are replaced by the difference between the data values and the previous values
3. Moving average (MA) incorporates the dependency between an observation and a

residual error from a moving average model applied to lagged observations

Each of the above component functions as a parameter with the standard notation of ARIMA(p,d,q) which is defined as
1. p: the number of lag observations in the model; also known as the lag order
2. d: the number of times that the raw observations are differenced; also known as the degree of differencing
3. q: the size of the moving average window; also known as the order of the moving average

A function is implemented to choose the best hyperparameters by evaluating the model for different combinations of (p,d,q) by specifying a range of values.

| Model(ARIMA(p,d,q)) | Accuracy |
|---|---|
| ARIMA(0,1,1) | 91.7 |
| ARIMA(0,2,1) | 91.7 |
| ARIMA(1,0,0) | 91.8 |

Figure 2.5: Accuracy of the model for combinations of (p,d,q) with least RMSE

**2.6 LSTM Model:** To implement a deep learning technique in TensorFlow, we are implementing a simple LSTM model. With a window size of 7 we are going to use the previous 7 days to predict the stock close price for today.

**2.7 Databricks Platform:** Databricks is a convenient platform to run Apache Spark. The code is deployed in Databricks Spark Cluster which has advantages of computation efficiency and handling bigdata.

**2.8 GCP Dataprocs:** We set up a storage bucket in GCP dataprocs in which our code was imported to. A cluster was created to run the code from the storage bucket. However, we were not successful in completion of GCP Dataprocs deployment.

**3. Conclusion:** The comparative results of both ARIMA and LSTM models are tabulated as in Figure 3.1. When deployed on Databricks, the computation time reduces because of the Spark clustering compared to usual Jupyter Notebook. The advantage of using Databricks lets you have a Spark environment already loaded as setting up of Spark environment on local machine is time consuming and complicated.

| Date | Actual | LSTM_Predicted | Arima_Predicted |
|---|---|---|---|
| 2019-04-22 | 1887.31 | 1865.43 | 1861.05 |
| 2019-04-23 | 1923.77 | 1870.62 | 1886.67 |
| 2019-04-24 | 1901.75 | 1877.36 | 1923.12 |
| 2019-04-25 | 1902.25 | 1890.42 | 1901.11 |
| 2019-04-26 | 1946.19 | 1904.37 | 1901.61 |

Figure 3.1 Comparison of results

**4. References:**


1. https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/
2. https://databricks.com/spark/getting-started-with-apache-spark
3. https://heartbeat.fritz.ai/a-beginners-guide-to-implementing-long-short-term-memory-networks-lstm-eb7a2ff09a27
4.https://www.statsmodels.org/stable/generated/statsmodels.tsa.stattools.adfuller.html
5. https://blog.minitab.com/blog/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit
6. https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/
7.https://spark.apache.org/docs/2.3.1/api/python/index.html
8. https://heartbeat.fritz.ai/a-beginners-guide-to-implementing-long-short-term-memory-networks-lstm-eb7a2ff09a27