VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

J Component report

Programme : B.Tech in CSE with specialization in AI and ML

Course Title : Machine Learning Essentials

Course Code : CSE1015

Slot : A1

Title: Ubiquant Stock Market Prediction

Team Members:

Uzair Alladin, 20BAI1062

Pinni Venkata Abhiram, 20BAI1132

Varsha Sharma, 20BAI1190

Aron Ritesh, 20BAI1195

Faculty: Dr. R. Rajalakshmi **Date:** 29.04.2022

CSE1015 – Machine Learning Essentials

J Component Report

A project report titled

Ubiquant Stock Market Prediction

(Data Challenge)

By

20BAI1062	Uzair Alladin
20BAI1132	Pinni Venkata Abhiram
20BAI1190	Varsha Sharma
20BAI1195	Aron Ritesh

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGENEERING
WITH
SPECIALIZATION IN ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Submitted to

Dr. R. Rajalakshmi

School of Computer Science and Engineering

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

April 2022

DECLARATION BY THE CANDIDATE

I hereby declare that the report titled "Ubiquant Stock Market Prediction" submitted by me to VIT Chennai is a record of bona-fide work undertaken by me under the supervision of **Dr. R. Rajalakshmi, Associate Professor, SCOPE, Vellore Institute of Technology, Chennai.**

Varsha

VARSHA SHARMA (20BAI1190)

Uzair

UZAIR ALLADIN (20BAI1062)

Aron Ritesh

ARON RITESH (20BAI1195)

P.V.Abhiram

P.V. ABHIRAM (20BAI1132)

Signature of the Candidates

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. R. Rajalakshmi,** School of Computer Science and Engineering for her consistent encouragement and valuable guidance offered to us throughout the course of the project work.

We are extremely grateful to **Dr. R. Ganesan, Dean,** School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our **Head of the Department** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the courses.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

BONAFIDE CERTIFICATE

Certified that this project report entitled "Ubiquant Stock Market Prediction" is a bona-fide work of **Uzair Alladin (20BAI1062), Varsha Sharma (20BAI1190), Aron Ritesh (20BAI1195)** and **Pinni Venkata Abhiram (20BAI1132)** carried out the "J"-Project work under my supervision and guidance for CSE1015 – Machine Learning Essentials.

Dr. R. Rajalakshmi

SCOPE

TABLE OF CONTENTS

ABSTRACT

The financial market is bound to fluctuate, regardless of your investment strategy. Professional investors attempt to estimate their overall profits despite this volatility. Risks and rewards vary depending on the type of investment and other factors that influence stability and volatility. Many computer-based algorithms and models for financial market trading are used to try to anticipate profits. Data science, on the other hand, could boost quantitative researchers' capacity to forecast investment returns with new tools and approaches.

Ubiquant Investment (Beijing) Co., Ltd is a well-known Chinese quantitative hedge fund. They were founded in 2012, and its quantitative financial market investment is driven by multinational skills in math and computer science, as well as cutting-edge technology. Overall, Ubiquant is dedicated to providing investors with consistent long-term returns.

The goal of this project is to create a model that predicts the rate of return on an investment. The goal would be to train and evaluate an algorithm with as much accuracy as possible using historical pricing.

INTRODUCTION

Stock market constitutes the buyers and sellers of stocks that are owned by various businesses. In order to increase their profits and plan their business strategies, these companies and corporations would like to know what could be the future value of their market shares. Thus, stock prediction comes out to be useful for such enterprises to help them design and carry out their businesses accordingly.

In our project, we have created a model that forecasts the investment's return rate for a Chinese company called Ubiquant Investment (Beijing) Co., Ltd.

The attributes in the offered dataset were extracted from real historical data from thousands of investments. A time-series based machine learning model is thus created to predict the return rate of an investment.

The various models that were used to make time series forecasting for the given dataset were DNN ensemble model, ARIMA model and the Exponential Moving Average.

The DNN ensemble model reduces the variance of predictions and generalisation error by combining predictions from numerous neural network models.

ARIMA (Autoregressive Integrated Moving Average) is a statistical analysis model that predicts future trends using time series data. The model employs the time series data's lags and lagged forecast errors to create an equation that may be used to estimate future values.

EWMA (Exponentially Weighted Moving Average) applies weights to the values of a time series. More weight is applied to more recent data points, making them more relevant for future forecasts.

LITERATURE SURVEY

"Stock Price Prediction Using the ARIMA Model"

A. A. Ariyo, A. O. Adewumi and C. K. Ayo, "Stock Price Prediction Using the ARIMA Model," 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, 2014, pp. 106-112, doi: 10.1109/UKSim.2014.67.

Introduction

It is considered that the ARIMA models are very efficient and robust machine learning models for financial stock market predictions. In case of short-term prediction, ARIMA model is better than even the most popular ANN techniques.

In this paper extensive process of building ARIMA models for short-term stock price prediction is presented. The results obtained from real-life data demonstrated the potential strength of ARIMA models to provide investors short-term prediction that could aid investment decision making process.

Proposed Model

The tool used for implementation of the model is Eviews software version 5. The dataset used is the historical daily stock prices obtained from two countries stock exchanged. The two companies whose stock prices information is taken are *Nokia* and *Zenith Bank*. It consists of the columns namely: open price, low price, high price and close price. The target variable whose value to be predicted is 'closing price'.

The following criteria are used for each stock index in order to find out the best ARIMA model out of the several experiments performed:

- Relatively small of BIC (Bayesian or Schwarz Information Criterion)
- Relatively small standard error of regression (S.E. of regression)
- Relatively high of adjusted R^2
- Q-statistics and correlogram show that there is no significant pattern left in the autocorrelation functions (ACFs) and partial autocorrelation functions (PACFs) of the residuals, it means the residual of the selected model are white noise.

The Augmented Dickey Fuller test (ADF) is performed to check if the time-series is stationary or not.

The values of the parameters autoregressive (p) and moving average (q) is found out after experimenting with different values of *p* and *q*.

Conclusion and Results

For **Nokia Stock Index**, the best model is found out to be ARIMA (2,1,0) with 5.329, 0.0033 and 3.5808 as BIC, adjusted R^2 and S.E. of Regression respectively.

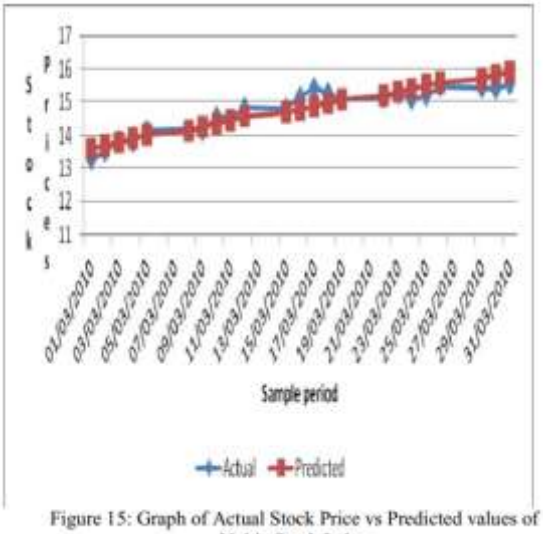


Figure 15: Graph of Actual Stock Price vs Predicted values of Nokia Stock Index

Therefore, from the plot of actual and predicted values, it can be seen that the results are pretty much satisfactory.

For **Zenith Bank Index**, the best model is found out to be ARIMA (1,0,1) with 2.3736, 0.9972 and 0.7872 as BIC, adjusted R^2 and S.E. of Regression respectively.

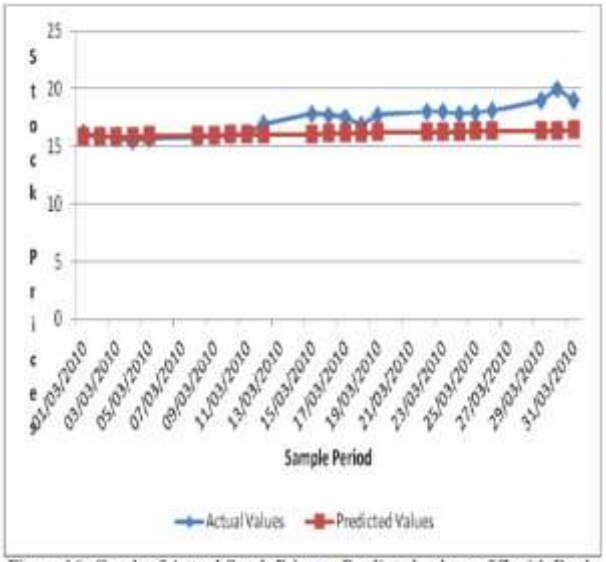


Figure 16: Graph of Actual Stock Price vs Predicted values of Zenith Bank Stock Index

Therefore, from the plot of actual and predicted values, it can be seen that the ARIMA model selected is quite impressive as there are some instances of closely related of actual and predicted values.

"Exponential Smoothing Methods for Detection of the Movement of Stock Prices"

Shaik Shahid, SK. Althaf Rahaman, "Exponential Smoothing Methods for Detection of the Movement of Stock Prices," 2020 International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Volume-8 Issue-5, January 2020, doi: 10.35940/ijrte.E6409.018520

Introduction

Exponential smoothing is a technique for smoothing time data in order to predict the near future. Exponential smoothing's fundamental principle is to forecast future values by taking a weighted average of all previous values in our time series data. Exponential smoothing is a technique for forecasting time series data based on three factors: level, trend, and seasonal component.

Based on these methods, we have three types of smoothing techniques:

- **Simple Exponential Smoothing (SES)**

  The SES method can be used to anticipate future values of time series data that does not contain a trend or seasonality.

  The forecast equation for SES is:

  Forecast = Estimate level at most recent time point

  $$F_{t+k} = L_t$$

  the kth step ahead of SES forecast is simply the most recent Estimate of level (L), at time (t) for which we need to estimate the level using the level updating equation:

  $$L_t = \alpha Y_t + 1 - \alpha\, L_{t-1}$$

  The above equation states that the algorithm is learning the new level from the newest data it is seeing.

- **Holt's Exponential Smoothing (HES)**

  Holt's Exponential Smoothing is also called Double Exponential Smoothing. The major goal of this method is to extend the SES setup to include a trend component. When there is a trend in the time series data but no seasonality, Halt's Exponential Smoothing might be used.

  The forecast equation for Holt's Exponential Smoothing is:

  Forecast = Estimated level + Trend at most recent time point

  $$F_{t+k} = L_t + KT_t$$

  Here, we have two update equations. One for level and one for trend.

  Level updating equation,

  $$L_t = \alpha Y_t + (1 - \alpha)\, Lt_{-1} + T_{t-1}$$

The above equation states that we are adjusting previous level by adding trend to it.

Trend updating equation,

$$T_t = \beta\ (L_t - L_t{-}1) + (1 - \beta)\ T_{t-1}$$

The above equation states that we are updating previous trend by using the difference between the most recent level values.

- **Holt Winter's Exponential Smoothing (HWES)**

Any Winter's Exponential Smoothing also called Holt-Winter's Exponential Smoothing and even sometimes called Triple Exponential Smoothing. This method takes the idea of Holt's method and adds a seasonal component to create the even more complex system. We assume here, that the series has a level, trend, seasonality with M seasons and noise.

The forecast equation for Winter's Exponential Smoothing is:

Forecast = Estimated level + Trend + Seasonality at most recent time point

$$F_{t+k} = L_t + KT_t + S_{t+k-M}$$

In Winter's Exponential Smoothing method, we have three smoothing constants and there are three updating equations.

Level updating equation,

$$L_t = \alpha\frac{\mathrm{Yt}}{\mathrm{St-M}} + (1 - \alpha)(L_{t-1} + T_{t-1})$$

Here, when we divide Y by S it means, we are de-seasonalizing the value of Y. In this level equation we are therefore updating the previous level, else of t-1 by adding the previous trend estimate, $T_{t-1}$and then combining with the de-seasonalized value of Y_t.

Trend updating equation,

$$T_t = \beta\ (L_t - L_{t-1}) + (1 - \beta)\ T_{t-1}$$

The above equation is similar to the one in Holt's Exponential Smoothing method.

Seasonality updating equation,

$$S_t = \gamma\frac{\mathrm{Yt}}{\mathrm{Lt}} + (1 - \gamma)\ S_{t-M}$$

Here, we can see the Y_t is divided by the level component L_t. This gives the de-trended value of Y. So, the seasonal component St-M with the de-trended value of Y_t.

Smoothing constant (γ) is controlling the speed of adjusting the seasonality

Proposed Model

Stock price data-sets containing historical stock prices of two Indian IT businesses, Tata Consultancy Services Limited (TCS) and Hindustan Computers Limited (HCL), were downloaded from Kaggle.com for this time-series data analysis task.

Date, Symbol, Series, Prev Close, Open, High, Low, Last, Close VWAP, Volume, Turnover, Trades, Deliverable Volume, and percent Deliverable are among the attributes in the data collection.

The data set is split into two sections: training data and test data.

Data cleaning, data transformation, and data splitting are all steps in preparing the data collection for use in the model (train dataset, test dataset). The data had previously been cleansed and converted. The data from the previous three years is divided into two training sets and one testing set.

Results and Conclusion

The root means square error (RMSE), mean absolute percentage error (MAPE) and mean absolute error (MAE) metrics are used to evaluate the effectiveness of the methods.

According to the results, Holt's approach of exponential smoothing performed the worst on the TCS data-set, while the simple exponential smoothing method performed the worst on the HCL data-set. On the other hand, Holt-exponential Winter's smoothing method performed the best on both firms' data sets, outperforming the other two methods.

“A Survey on Stock Market Prediction Using Machine Learning Techniques”

Polamuri, Subba & Srinivas, Kudipudi & Mohan, A.. (2020). A Survey on Stock Market Prediction Using Machine Learning Techniques. 10.1007/978-981-15-1420-3_101.

Introduction

Stock Market Prediction is an extremely difficult task, and there are various methods to achieve such a task. Due to the fluctuating nature of the stock, the stock market is uncertain in various ways, making it a complex model.

The various methods that are discussed in this research paper, along with their advantages and disadvantages, to achieve this feat are –

- Artificial Neural Network
- Hidden Markov Model
- Support Vector Machine for Stock prediction
- Time Series Linear Model
- Arima Model

Proposed Techniques

S.No.	Technique	Model Working	Advantages	Disadvantages
1.	Artificial Neural Network (ANN)	Works on the Basis of the Back-propagation algorithm. A neural network of Multilayer Perceptron is used. Consists of an Input Layer with a set of sensor nodes as input nodes. Hidden layers of computation nodes and computation nodes of output layer.	Better Performance compared to Regression. Lower Prediction Error. Great Ability to predict from large databases.	Prediction gets worse with Increased Noise Variation

2.	Hidden Markov Model	Stock Market trend analysis is based on this. Hidden sequence of states, and corresponding probability values are found for a particular observation sequence.	Gives Better optimization, and is used mainly for optimization purposes	Evaluation and Decoding
3.	Support Vector Machine	Uses a technique called the kernel trick to transform your data. Based on the transformations it finds an optimal boundary between the outputs.	Does not lose much Accuracy when applied to a sample from outside the training sample	Exaggerate minor fluctuations, decreasing predictive ability
4.	Time Series Linear Model	A stochastic method. Ideal linear model is created and data is incorporated so reflects the properties of actual data	Integrate the actual data to the ideal linear model	Traditional and Seasonal trends present in the data
5.	ARIMA Model	Identified, estimated, and diagnosed with time-series data. Generate short-term forecasts. Future Value of a variable is a linear combination of past values and past errors	Identified, estimated, and diagnosed with time-series data. Generate short-term forecasts. Future Value of a variable is a linear combination of past values and past errors	Suitable for short term predictions only

Conclusion

Therefore, we can see that each of the models has its own certain advantages and disadvantages. According to the needs of the Stock Market Prediction, it has to be deliberated as to which Method would be the most efficient and satisfying. There is a possibility that we have to combine different methods in order to reach the most efficient one.

“Review of Data Pre-processing Techniques in Data Mining”

Bhaya, Wesam. (2017). Review of Data Preprocessing Techniques in Data Mining. Journal of Engineering and Applied Sciences. 12. 4102-4107. 10.3923/jeasci.2017.4102.4107.

Introduction

The given Research Paper talks about the possible Data Pre-processing techniques that can be used in order to obtain better and more efficient results. Even though this Research Paper talk about Pre-processing in the field of Data Mining, similar techniques can be applied in the field of Machine Learning for optimized results. Raw Data usually susceptible to Missing Values, noisy data, incomplete data, inconsistent data and outlier data. So, it is important for these data to be processed before being worked upon. Data Pre-processing deals with data preparation and transformation of the dataset and seeks at the same time to make knowledge discover more efficient.

Several techniques that can be used for Data Pre-processing are

- Cleaning
- Integration
- Transformation
- Reduction

The Research Paper elaborates on these techniques and when can they be utilized.

Proposed Techniques

1. **Data Cleaning**
   Row Records may have incomplete records, noise values, outliers and inconsistent data. Data Cleaning is used to find the missing values, smooth noise data, recognize outliers and correct inconsistent data. Missing values can be filled in the following ways
   - Ignore the Tuple

- Fill the Values Manually
- Use a Global Constant to fill the Missing Values
- Use the Attribute Mean to Fill the Missing Values
- Use the Attribute mean for all samples belonging to the same class as the given tuple
- Use the most probable value to find the missing value

2. **Noise Data**

   Noise Data is a random error or variance in a measured variable. Noise Error means that there is an error data or outliers which deviates from the normal.

   Possible Techniques include

   - Binning – Smoothing stored data based on its "neighbourhood" which is the values around it. Sorted values are divided into a number of buckets or bins. They perform local smoothing
   - Regression – Fitting the data into a function. Uses the line of best fit for two variables, so that each attribute can be used to predict the other
   - Clustering – Grouping set of points into clusters according to a distance measure. Technique is used to detect outliers, since it is grouping similar points into a cluster.

3. **Data Integration**

   This technique works by combining data from multiple and various resources into one consistent data and store, like in data warehouse. These resources can have multi database, files, or data cubes. In data integration, there are a number issues like - schema integration, redundancy and object matching which are important aspects.

4. **Data Transformation**

   Includes transforming the data to suitable forms, it includes the following

   - Smoothing
   - Aggregation
   - Generalization
   - Normalization
   - Data reduction

- Data Cube aggregation
- Attribute Subset selection
- Dimensionality reduction
- Numerosity reduction

Conclusion

Real world data tend to be incomplete, inconsistent, noisy and missing. Data pre-processing, which includes Data integration, Data transformation, Data Cleaning, Data transformation and Data Reduction, is one of the important matters for both data warehousing, mining, and Machine Learning. The study explains an overview of those techniques.

PROPOSED METHODOLOGY

Scaling the features

In order to scale the features, we have used the *MinMax Scaler*. The *MinMax Scaler* transforms features by scaling each feature to a given range. The values are scaled in the range of 0 to 1.

The *MinMaxScaler( )* function is called from the python's *sklearn.preprocessing* module to perform the scaling of features.

Principal Component Analysis (or PCA)

PC Analysis is the process of calculating the principal components and using them to change the basis of the data, often simply using the first few and disregarding the rest.

PCA is used in exploratory data analysis and for making predictive models. It is commonly used for dimensionality reduction by projecting each data point onto only the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible.

In our model, we have used the Principal Component Analysis to reduce the 300 features to around 50 features.

1. Deep Neural Networks (DNN)

A neural network consists of several connected units called nodes. These are the smallest part of the neural network and act as the neurons in the human brain. When a neuron receives a signal, it triggers a process. The signal is passed from one neuron to another based on the input received. A complex network is formed that learns from feedback.

The DNN aspect of our project is an ensemble of DNN Algorithms that consist of the following:

- DNN module imported from TensorFlow
- DNN module imported from Keras
- Convolution Neural Network (CNN) {Loosely Implemented}

The core layers that are connected with each other and together form the Neural Network, in our project are:

1. Input
2. Dense
3. Convolution 1D and Convolution 2D
4. Embedding
5. LSTM

2. ARIMA Model

An **Autoregressive Integrated Moving Average,** or **ARIMA** is a statistical analysis model that uses time series data to predict future trends. The model uses lags and the lagged forecast error of the time series data in order to generate an equation that can be used to forecast future values.

An ARIMA model is characterised by three terms:

- p: this is the order of the auto-regressive (AR) term, which is the number of lags of Y to be utilized as predictors.
- q: this is the order of the moving average (MA) term, which means that the number of lagged forecast error should be used in the ARIMA model.
- d: the number of differences required to make the time series stationary

The equation for the ARIMA model is:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \dots + \beta_p Y_{t-p} \epsilon_t + \emptyset_1 \epsilon_{t-1} + \emptyset_2 \epsilon_{t-2} + \dots + \emptyset_q \epsilon_{t-q}$$

where, Y_{t-1} is the lag1 of the series. β_1 is the coefficient of lag1 and is the term of intercept that is calculated by the model.

Augmented Dickey-Fuller Test (ADF)

Finding the order of differencing 'd' in the ARIMA model

The augmented dickey-fuller test is used to check if the time series is stationary or not. It is necessary to check whether the series is stationary or not in order to decide a value of *d*.

The value of *d* will be the difference if the series is not stationary, otherwise, the value of *d* will be equal to 0.

The augmented dickey-fuller (ADF) test's null hypothesis is that the time series is not stationary. Thus, if the ADF test's p-value is less than the

significance level (0.05), then we will reject the null hypothesis and infer that the time series is definitely stationary.

Finding the order of the Auto-Regressive (AR) term, 'p'

The Partial Autocorrelation (PACF) plot is used to find the number of AR terms. When the contributions from intermediate lags are removed, Partial Autocorrelation is defined as the correlation between the series and its lag. As a result, PACF prefers to convey the series' and lag's pure correlation. As a result, we can determine whether or not that lag is required in the Auto-Regressive (AR) term.

Partial Autocorrelation of lag(k) of a series is the coefficient of that lag in the Auto-Regression Equation of Y.

$$Y_t = \alpha_0 + \alpha_1 Y_{t-1} + \alpha_2 Y_{t-2} + \alpha_3 Y_{t-3}$$

By inserting enough AR terms, any autocorrelation in a stationary series can be corrected. As a result, we can set the order of the Auto-Regressive (AR) term to the number of lags that cross the PACF Plot's significance limit.

Finding the order of the Moving Average (MA) term, 'q'

At the same time, 'q' is the order of the 'MA' (Moving Average) term, which means that the number of lagged forecast errors should be used in the ARIMA Model.

The ACF plot expresses the number of Moving Average (MA) terms needed to remove the autocorrelation in the stationary series.

3. Exponential Weighted Moving Average (or EWMA)

EWMA applies weights to the values of a time series. More weight is applied to more recent data points, making them more relevant for future forecasts.

The smoothing parameter (or learning rate) **alpha** of the exponential smoothing model will determine how much importance is given to the most recent demand observation.

$$f_t = \alpha d_{t-1} + (1 - \alpha) f_{t-1}$$

$$0 < \alpha \leq 1$$

where,

- **α** is a ratio (or a percentage) of how much importance the model will allocate to the most recent observation compared to the importance of demand history.
- **α d_{t-1}** represents the previous demand observation times the learning rate. You could say that the model attaches a certain weight (alpha) to the last demand occurrence.
- **(1- α) f_{t-1}** represents how much the model remembers from its previous forecast. Note that this is where the recursive magic happens as f_{t-1} was itself defined as partially d_{t-2} and f_{t-2}.

4. LGBM Regressor

The LGBM (Light Gradient Boosting Model) employs two techniques: GOSS (Gradient Based on Side Sampling) and EFB (Exclusive Feature Bundling). GOSS will ignore the large chunk of the data with modest gradients and just utilise the remaining data to calculate the total information gain. The data instances with large gradients actually play a bigger role in information gain computation. Despite employing a smaller dataset than other models, GOSS can produce reliable findings with a large information gain.

The EFB pairs mutually exclusive features with nothing, however it will seldom take any non-zero value at the same time in order to limit the number of features. This has an impact on the overall result for successful feature elimination without jeopardising the split point's precision.

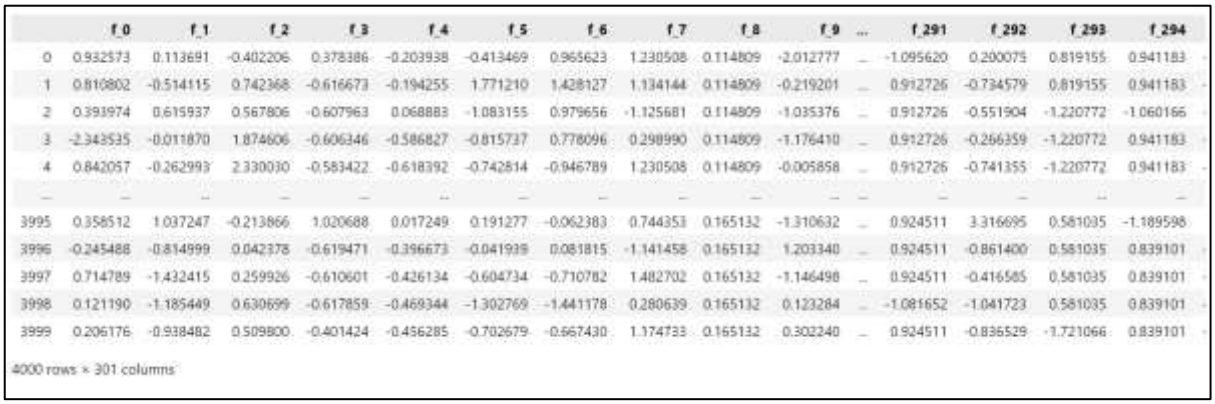
The main features of the LGBM model are as follows:

- Higher accuracy and a faster training speed.
- Low memory utilization
- Comparatively better accuracy than other boosting algorithms and handles overfitting much better while working with smaller datasets.
- Parallel Learning support.
- Compatible with both small and large datasets

RESULTS AND DISCUSSION

Principal Component Analysis (or PCA):

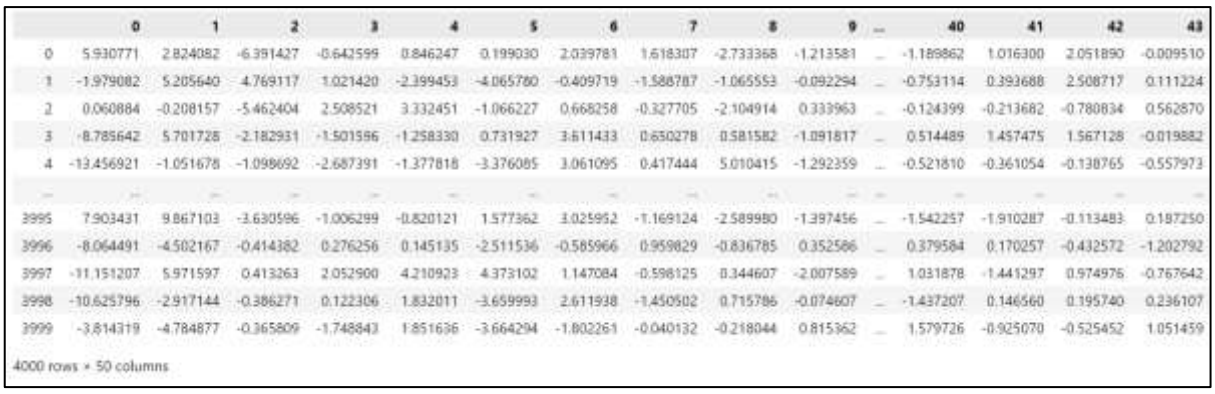
Initially, the given dataset was:



	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	...	f_291	f_292	f_293	f_294
0	0.932573	0.113691	-0.402206	0.378386	-0.203938	-0.413469	0.965623	1.230508	0.114809	-2.012777	...	-1.095620	0.200075	0.819155	0.941183
1	0.810802	-0.514115	0.742368	-0.616673	-0.194255	1.771210	1.428127	1.134144	0.114809	-0.219201	...	0.912726	-0.734579	0.819155	0.941183
2	0.393974	0.615937	0.567806	-0.607963	0.068883	-1.083155	0.979656	-1.125681	0.114809	-1.035376	...	0.912726	-0.551904	-1.220772	-1.060166
3	-2.343535	-0.011870	1.874606	-0.606346	-0.586827	-0.815737	0.778096	0.298990	0.114809	-1.176410	...	0.912726	-0.266359	-1.220772	0.941183
4	0.842057	-0.262993	2.330030	-0.583422	-0.618392	-0.742814	-0.946789	1.230508	0.114809	-0.005858	...	0.912726	-0.741355	-1.220772	0.941183
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3995	0.358512	1.037247	-0.213866	1.020688	0.017249	0.191277	-0.062383	0.744353	0.165132	-1.310632	...	0.924511	3.316695	0.581035	-1.189598
3996	-0.245488	-0.814999	0.042378	-0.619471	-0.396673	-0.041939	0.081815	-1.141458	0.165132	1.203340	...	0.924511	-0.861400	0.581035	0.839101
3997	0.714789	-1.432415	0.259926	-0.610601	-0.426134	-0.604734	-0.710782	1.482702	0.165132	-1.146498	...	0.924511	-0.416585	0.581035	0.839101
3998	0.121190	-1.185449	0.630699	-0.617859	-0.469344	-1.302769	-1.441178	0.280639	0.165132	0.123284	...	-1.081652	-1.041723	0.581035	0.839101
3999	0.206176	-0.938482	0.509800	-0.401424	-0.456285	-0.702679	-0.667430	1.174733	0.165132	0.302240	...	0.924511	-0.836529	-1.721066	0.839101

4000 rows × 301 columns

After applying PCA, the number of features were reduced to 50.
Thus, the dataset becomes:



	0	1	2	3	4	5	6	7	8	9	...	40	41	42	43
0	5.930771	2.824082	-6.391427	-0.642599	0.846247	0.199030	2.039781	1.618307	-2.733368	-1.213581	...	-1.189862	1.016300	2.051890	-0.009510
1	-1.979082	5.205640	4.769117	1.021420	-2.399453	-4.065780	-0.409719	-1.588787	-1.065553	-0.092294	...	-0.753114	0.393688	2.508717	0.111224
2	0.060884	-0.208157	-5.462404	2.508521	3.332451	-1.066227	0.668258	-0.327705	-2.104914	0.333963	...	-0.124399	-0.213682	-0.780834	0.562870
3	-8.785642	5.701728	-2.182931	-1.501596	-1.258330	0.731927	3.611433	0.650278	0.581582	-1.091817	...	0.514489	1.457475	1.567128	-0.019882
4	-13.456921	-1.051678	-1.098692	-2.687391	-1.377818	-3.376085	3.061095	0.417444	5.010415	-1.292359	...	-0.521810	-0.361054	-0.138765	-0.557973
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3995	7.903431	9.867103	-3.630596	-1.006299	-0.820121	1.577362	3.025952	-1.169124	-2.589980	-1.397456	...	-1.542257	-1.910287	-0.113483	0.187250
3996	-8.064491	-4.502167	-0.414382	0.276256	0.145135	-2.511536	-0.585966	0.959829	-0.836785	0.352586	...	0.379584	0.170257	-0.432572	-1.202792
3997	-11.151207	5.971597	0.413263	2.052900	4.210923	4.373102	1.147084	-0.598125	0.344607	-2.007589	...	1.031878	-1.441297	0.974976	-0.767642
3998	-10.625796	-2.917144	-0.386271	0.122306	1.832011	-3.659993	2.611938	-1.450502	0.715786	-0.074607	...	-1.437207	0.146560	0.195740	0.236107
3999	-3.814319	-4.784877	-0.365809	-1.748843	1.851636	-3.664294	-1.802261	-0.040132	-0.218044	0.815362	...	1.579726	-0.925070	-0.525452	1.051459

4000 rows × 50 columns

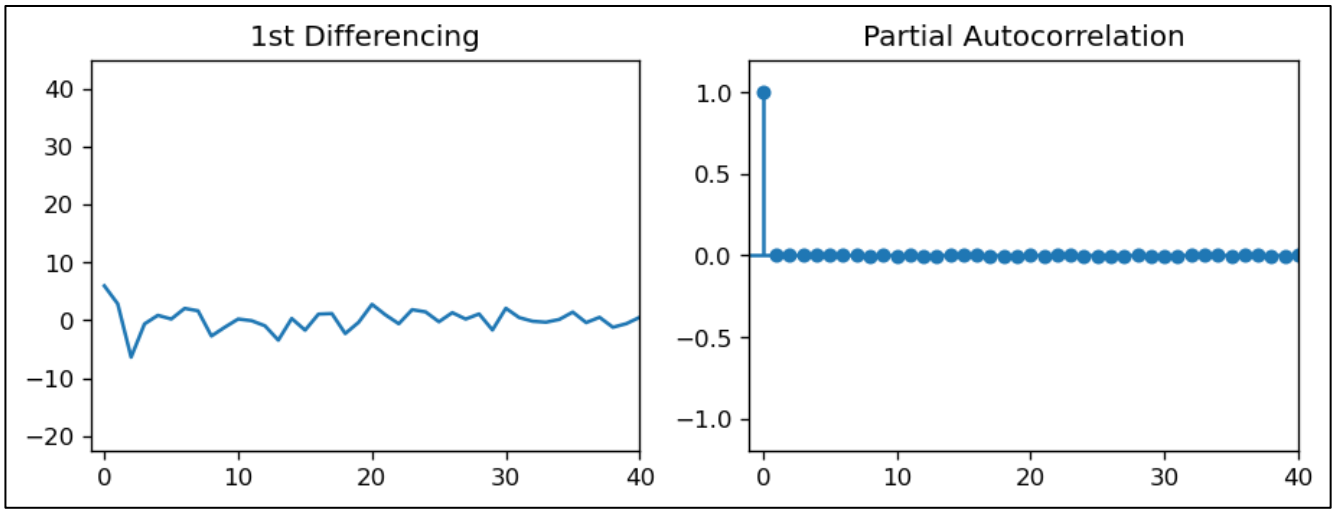
1. ARIMA Model

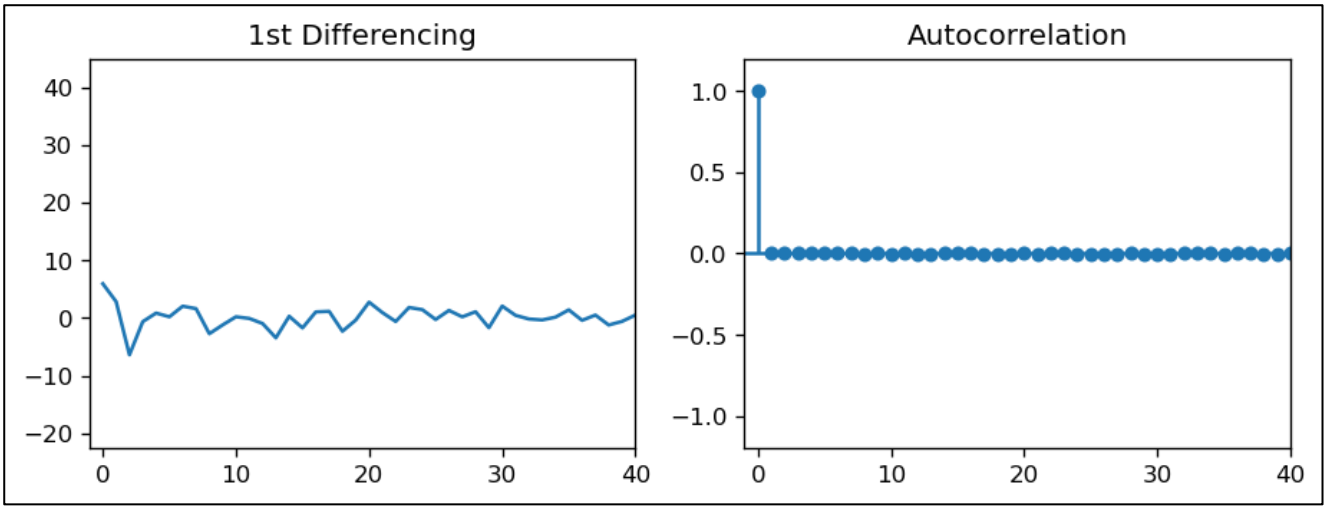
After Applying the Dicky-Fuller Test, we get the following Results:

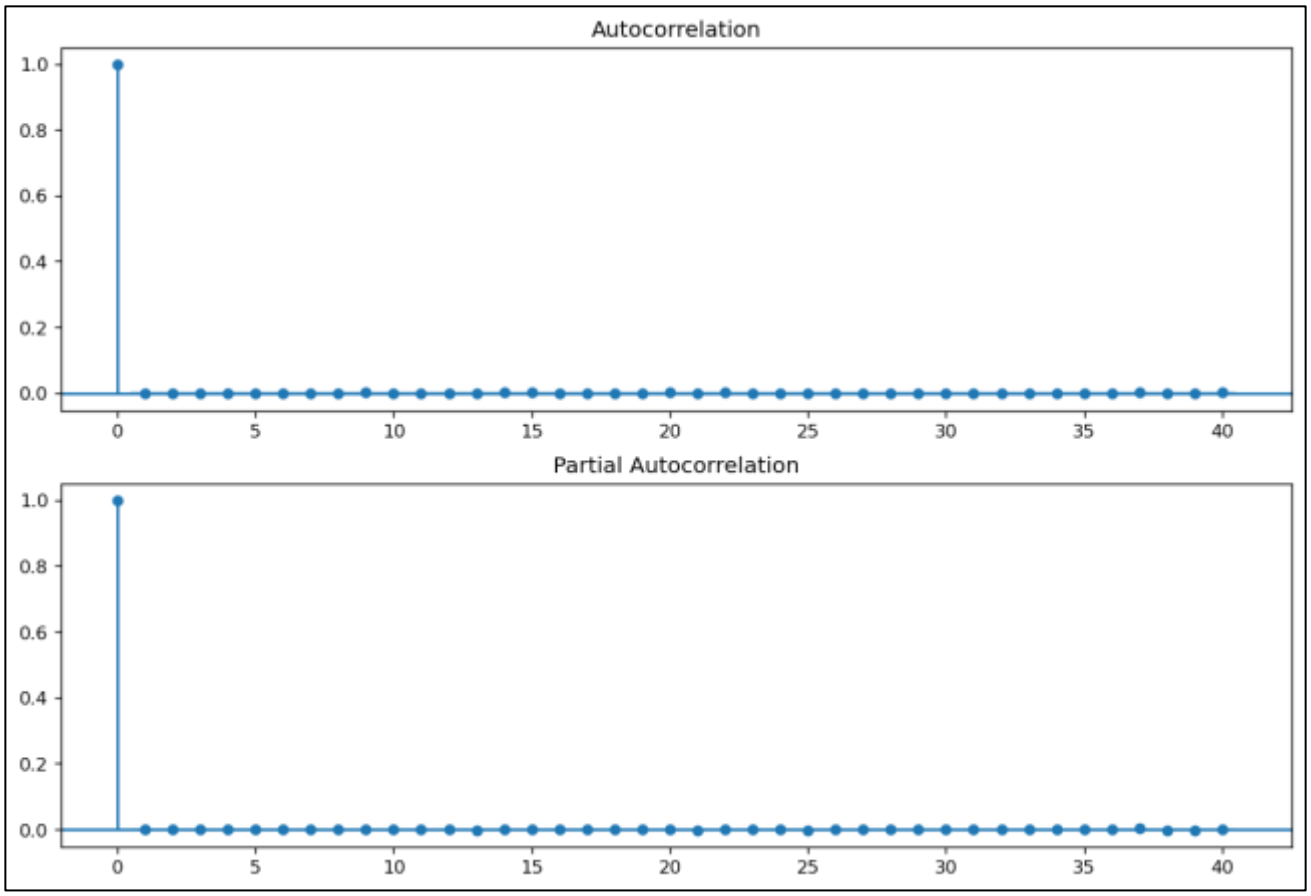
```
Augmented Dickey-Fuller Statistic: -451.578624
p-value: 0.000000
```

Since the p-value, we obtained is 0.It implies that the Null Hypothesis is Rejected and the time-series is Stationary. This means that we do not need to apply any additional processes to convert the data as it is already in the desired format.

In order to obtain the values for p and q, we use the Partial Autocorrelation and Autocorrelation plots, respectively:







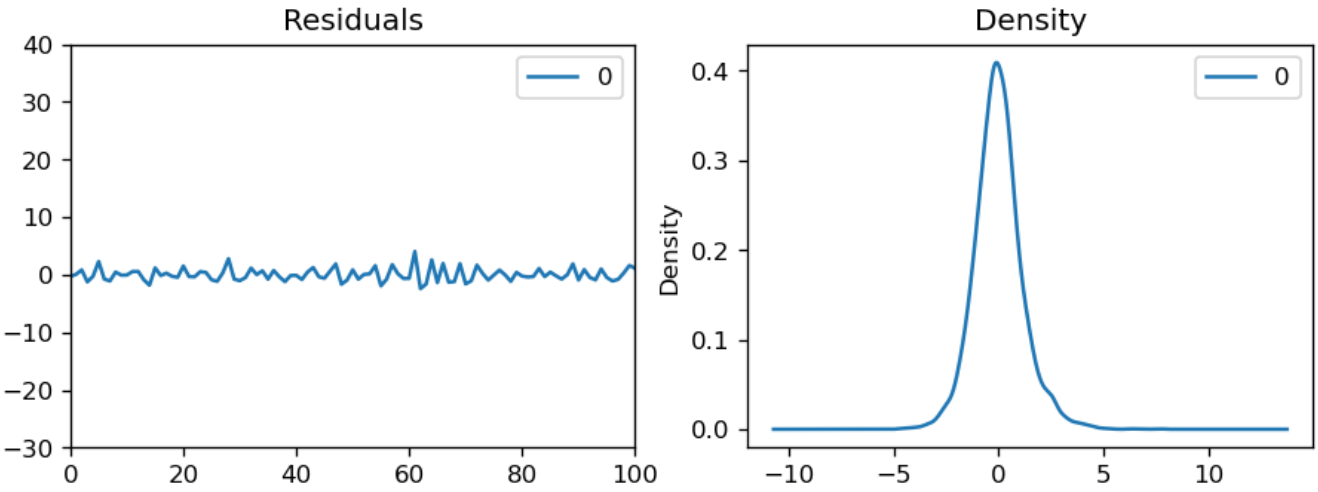
As we can observe from the plots above, the most efficient values of p and q that can be used for this Model are 1. Since the data is stationary, the d value that can be taken for this Model is 0.

The Model Summary Obtained is:

```
                               SARIMAX Results
==============================================================================
                                =======
       Dep. Variable:                           y   No. Observations:
                                  4000
Model:                 ARIMA(1, 1, 0)   Log Likelihood                -6
                                231.427
Date:                Thu, 21 Apr 2022   AIC                           12
                                466.854
Time:                        00:24:45   BIC                           12
                                479.441
Sample:                             0   HQIC                          12
                                471.316
                                 - 4000
Covariance Type:                  opg
```

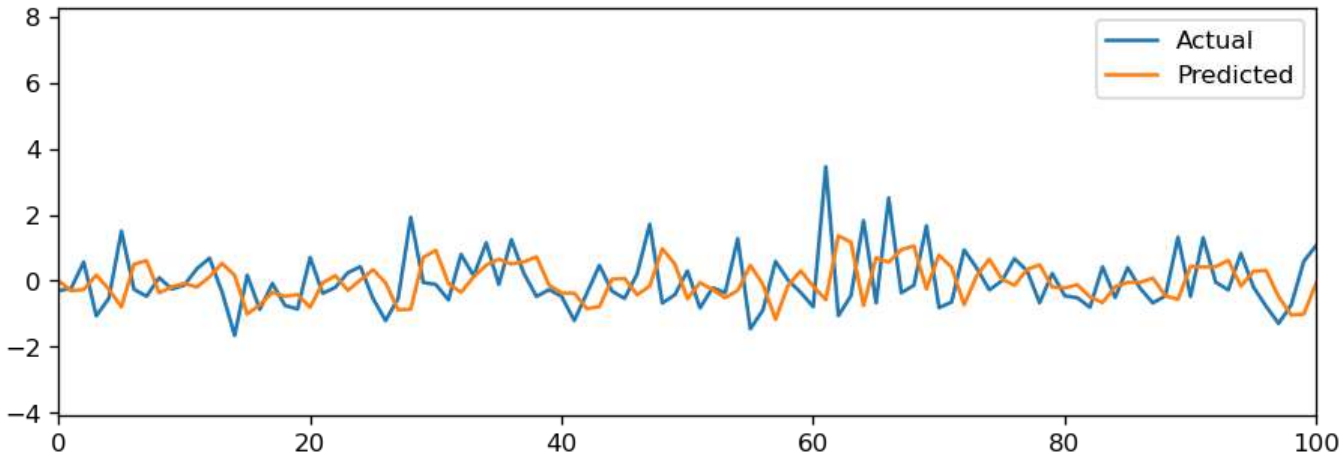
```
==============================================================================
                                 =======
     coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
                                 -------
   ar.L1         -0.4936      0.011    -45.824      0.000      -0.515
                                -0.472
   sigma2         1.3212      0.021     62.437      0.000       1.280
                                 1.363
==============================================================================
                                ============
    Ljung-Box (L1) (Q):                  115.65   Jarque-Bera (JB):
                                735.94
        Prob(Q):                                0.00   Prob(JB):
                                 0.00
          Heteroskedasticity (H):                1.02   Skew:
                                 0.45
        Prob(H) (two-sided):                    0.72   Kurtosis:
                                 4.90
==============================================================================
                                ============
```

The Residual as well as the Density Plot Obtained are:



We can see that since the mean of residuals and the Uniform Variance is around 0. Thus, they seem fine for the Model used.

The Final Predictions vs Actual Graph obtained is:



The Final Metrics we obtained for the model is:

```
Mean Absolute Error (MAE):  0.8593399464316749
Mean Squared Error (MSE):  1.3208995275858584
Root Mean Squared Error (RMSE):  1.1493039317716869
```

2. Exponential Moving Average

The Model involves the inclusion of two model parameters:

1. Time Period
2. Alpha Value

The Time Period talks about the average moving period length. This means that the Exponential Moving Average is designed to see the trend changes over a specified period of time in a specific time Frame.

The value of Alpha is calculated by the following Formula:

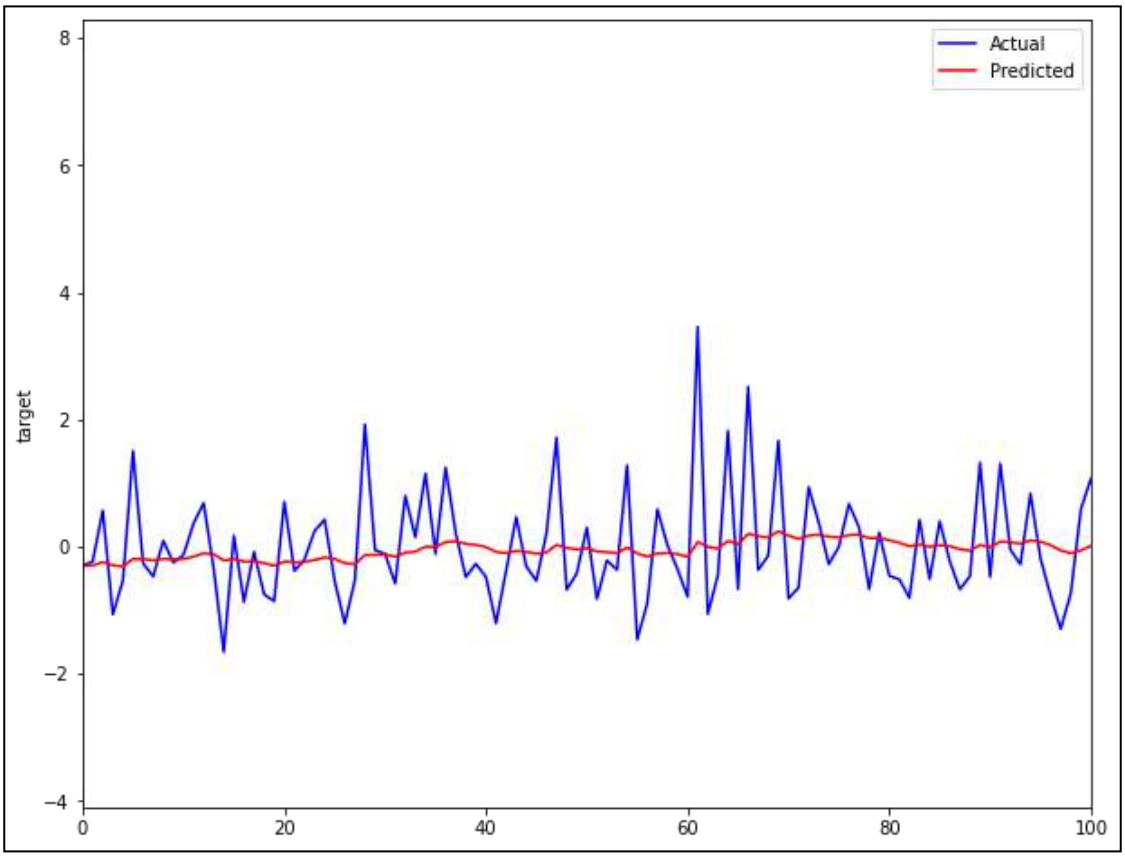
$$\alpha = \frac{2}{Time\ Period + 1}$$

The Time Period we chose was 30, and accordingly put the formula so that the value of alpha can be calculated.

We then created the Exponential Moving Average model with the specified parameters and obtained the following predictions:

```
0       -0.300875
1       -0.296369
2       -0.240552
3       -0.293728
4       -0.309096
...
3995    -0.061489
3996    -0.048541
3997    -0.100211
3998    -0.127950
3999    -0.150798
Name: target, Length: 4000, dtype: float64
```

The Final Actual vs Predicted plot obtained was:



The Final Metrics we obtained for the model is:

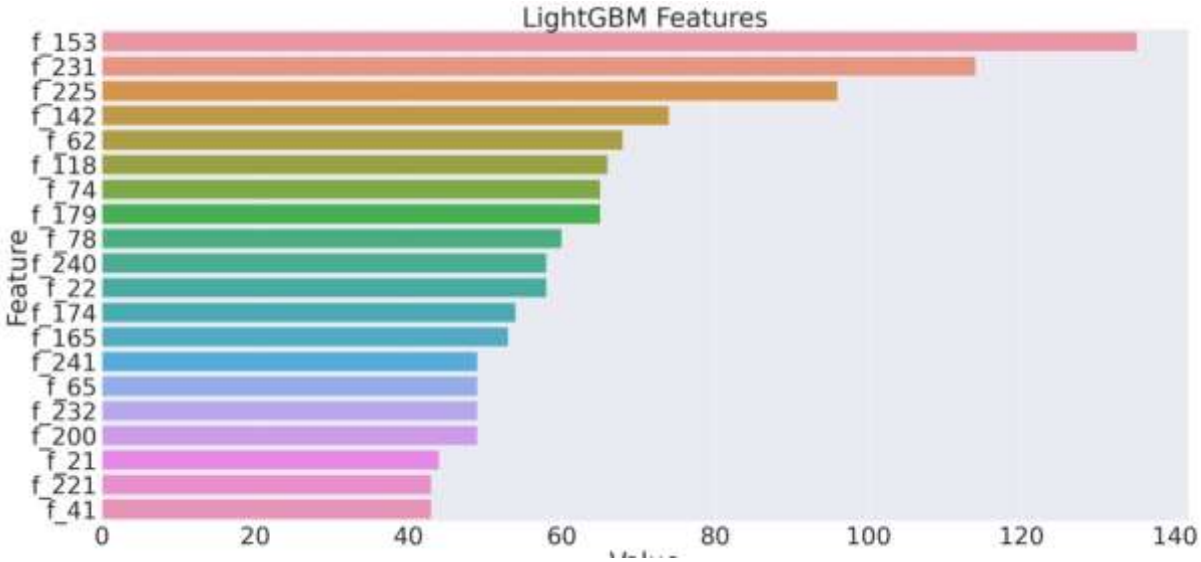
```
Mean Absolute Error (MAE):  0.6450389030746332
Mean Squared Error (MSE):  0.7976945483005927
Root Mean Squared Error (RMSE):  0.8931374744688484
```

We can see from the Above results that even though Exponential Moving Avera ge can be used for Time-Series Forecasting, it is not necessarily the most efficie nt model in terms of results for the Dataset given to us.

3. LGBM Regressor

A LGBM Regressor model is created using the *LGBMRegressor( )* function from the *lightgbm* python library. The type of boosting used is the traditional Gradient Boosting Decision Tree (or GBDT). The number of boosted trees to fit are 1400. The boosting learning rate is 0.05.

The LightGBM Features obtained are as follows:



The evaluation metrics obtained for the LGBM Regressor are:

```
Mean Absolute Error (MAE):  0.6061684525822186
Mean Squared Error (MSE):  0.846989918313624
Root Mean Squared Error (RMSE):  0.9203205519348266
```

4. DNN

We developed a model for which the test dataset was provided by the competition through the API and the metrics of the Neural Network are internally calculated using Kaggle's private dataset. Using these inherent calculations, the competition provides a score on the basis of the model submission. This Score is utilised to provide a final ranking.

The Score Obtained is: 0.1540 {Highest score is 0.186}

The Rank Obtained is: 670/2400 {2400 being the number of teams}

CONCLUSION

Evaluation Metrics

S.No.	Model	RMSE	MSE	MAE
1.	ARIMA	1.1493039317716915	1.3208995275858688	0.8593394643164451
2.	Exponential Moving Average	0.8931374744688484	0.7976945483005927	0.6450389030746332
3.	LGBM Regressor	0.9203205193482 66	0.846989918313624	0.6061684525822186

The above table describes the performance metrics of ARIMA, Exponential Moving Average and LGBM regressor which includes their RMSE, MSE and MAE values.

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit.

RMSE shows how low the error is, so the lower the better. ARIMA model scored an RMSE of 1.1493039317716915, EMA scored 0.8931374744688484 and LGBM scored 0.9203205519348266.

And so, from the values, EMA has the best RMSE score. However, ARIMA and EMA were trained on a reduced dataset while LGBM was trained on the whole. Thus, it would be invalid to conclude that EMA is the better model and this is apparent from the Actual vs Predicted graph of EMA [mentioned in results section under EMA]. LGBM has a better RMSE value, though trained on a larger dataset as compared to ARIMA. Thus, we conclude that LGBM is the better model among the three when looked at from the perspective of RMSE values.

In statistics, the mean squared error (MSE) or mean squared deviation (MSD) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors—that is, the average squared difference between the estimated values and the actual value.

The closer the MSE value is to 0, the better fit it is. However, it wouldn't be ideal to have the MSE value as 0 as it would infer that the model will struggle with new data. Thus, it would be appropriate to find a value that accommodates new data.
ARIMA model scored an MSE of 1.320899527585868, EMA scored 0.7976945483005927 and LGBM scored 0.846989918313624.
Keeping in mind the points discussed above, we conclude that the model with the values lying between the extremes (1.320899527585868 and 0.7976945483005927, i.e., 0.846989918313624) would be a suitable choice.
Thus, we conclude that LGBM is the better model among the three when looked at from the perspective of MSE values.

In statistics, mean absolute error (MAE) is a measure of errors between paired observations expressing the same phenomenon. Examples of Y versus X include comparisons of predicted versus observed, subsequent time versus initial time, and one technique of measurement versus an alternative technique of measurement.

Ideal MAE score would be closer to 0. ARIMA model scored an MAE of 0.859339946431645, EMA scored 0.6450389030746332 and LGBM scored 0.606168452582218. Thus, we conclude that LGBM is a better model among the three when looked through the perspective of MAE.

However, these evaluation metrics are relative in nature and it is wise to choose the model that works best for our current use case. And when looked at it from that perspective, even then, LGBM has a better Kaggle score of 0.133 thus, we conclude that between the three, LGBM is better. However, when looked at it from the perspective of use cases, DNN scored a better Kaggle score of 0.1540.

Thus, we conclude that DNN is better suited for this use case. And it has scored the highest in our Kaggle submissions.

REFERENCES

1. A. A. Ariyo, A. O. Adewumi and C. K. Ayo, *"Stock Price Prediction Using the ARIMA Model,"* 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, 2014, pp. 106-112, doi: 10.1109/UKSim.2014.67.
2. Bhaya, Wesam. (2017). *Review of Data Preprocessing Techniques in Data Mining.* Journal of Engineering and Applied Sciences. 12. 4102-4107. 10.3923/jeasci.2017.4102.4107.
   doi: 10.35940/ijrte.E6409.018520
3. Polamuri, Subba & Srinivas, Kudipudi & Mohan, A.. (2020). *A Survey on Stock Market Prediction Using Machine Learning Techniques.* 10.1007/978-981-15-1420-3_101.
4. Shaik Shahid, SK. Althaf Rahaman, *"Exponential Smoothing Methods for Detection of the Movement of Stock Prices,"* 2020 International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Volume-8 Issue-5, January 2020,
5. Ubiquant. (January 18, 2022). Ubiquant Market Prediction,Version 1. Retrieved January 20, 2022 from https://www.kaggle.com/c/ubiquant-market-prediction/data.

APPENDIX

Implementation / Code

.ipynb and *.html* files for the following models is uploaded in the Google Drive:

i. ARIMA
ii. DNN
iii. Exponential Moving Average
iv. LGBM Regressor

Along with this, the video demonstrating the contribution of each team member is also uploaded in the same Google Drive link.

Google Drive link:
https://drive.google.com/drive/folders/19Z56gDQRHQ9KY5EmCLMVaT5EF_KjroLA?usp=sharing