A Study of Visual and Data Analytics of Multivariate Time Series

Abhishek Vinodchandra Patel

A thesis submitted in partial fulfilment of the
requirements for the award of the degree of

MASTER OF DATA SCIENCE

CHARLES DARWIN UNIVERSITY
COLLEGE OF ENGINEERING, INFORMATION TECHNOLOGY AND ENVIRONMENT

October 2019

Declaration

I hereby declare that the work herein, now submitted as a thesis for the degree of Master of Data Science at Charles Darwin University, is the result of my own investigations, and all references to ideas and work of other researchers have been specifically acknowledged. I hereby certify that the work embodied in this thesis has not already been accepted in substance for any degree, and is not being currently submitted in candidature for any other degree.

Signature:

Date: 18 October 2019

Abstract

Keywords: *time series analysis, univariate, multivariate, LSTM, CNN, VAR, MLP, GRU, artificial neural network*

It may not be wrong to say that the world is full of connecting data and devices including devices in every smart city and towns from Industries to hospital regarding healthcare and mobile connectivity. The act of actual data time playing an important role after that analysis of time has come into reality having its own objectives. Thus, there is an issue to determine the data in real time and most of the data are in form of multivariate and univariate time series which is based on regular time interval. Time series analysis (TSA) is process of collecting the data at regular time intervals within certain time with determining trends, seasonal variances and cycle's that helps to forecast the future event. Within the Time Series Analysis, Time Series forecasting is the use of a model to predict future values based on previously observed values. Time series are widely used for non-stationary data, like economic, weather, stock price, and retail sales. Several methods and algorithms are used to determine univariate time series data like SARIMA and ARIMA whereas, majority of data are nonlinear multi variables and unstructured format so the significant scope of research is to forecast multivariate time series. Multivariate time series data contains multiple variables to reflect the real life and massive amount of data that has been created in the form of multivariate time series. Therefore, the task of analysing and forecasting multivariate time series is both challenging and important across industry and academic sectors. The purpose of the thesis is to determine the efficient algorithm which offers clear outcomes in real time. In this study, we compare the artificial neural network based models such as Long Short Term Memory (LSTM), Convolutional Neural Network (CNN), GRU (Gated Recurrent Unit), MLP (Multi-Layer perceptron), and ensemble learning model such as Random Forest and Statistical model such as Vector Auto-Regression (VAR). The proposed research claimed that artificial neural network based models outperform statistical models and the advance ensemble learning models also give significant results. Hence, the disruptive technologies like advance machine learning and neural network will able to predict multivariate time series and visual analytics helps to make effective decision to enhance the business and services for various industries like smart cities, health care and financial services.

Acknowledgements

First and foremost, I would like to thank and show my humble gratitude to my supervisor Dr Kewen Liao and Dr Bharanidharan Shanmugam. Particularly, I am very fortunate and thankful for the feedback, corrections and guidance given by my first supervisor Dr Kewen Liao. It was my supervisors support in the form of guidance and suggestion that made it possible to complete the thesis work. Your supervision and suggestions have lightened a spirit of thesis in me and led me successfully finish it. In every step of my work were also helped by my university's learning resources and I would like to thank Charles Darwin University for the learning resources such as library, weekly workshops and seminars that helped me to complete my research. It motivated and provided me the core content which is relevant to my thesis.

I would also like to show my gratitude to my family and class mates whose moral support and positive surrounding made working in thesis interesting and comfortable in so many ways. At last, I would like to thank every individual who directly and indirectly contributed in my thesis work.

Abhishek Patel
Charles Darwin University, October 2019

Table of Contents

DECLARATION I

ABSTRACT II

ACKNOWLEDGEMENTS III

LIST OF FIGURES VI

LIST OF TABLES IX

1. INTRODUCTION 1

1 LITERATURE REVIEW 4

1.1 Introduction 4
1.2 Univariate Time Series 5
1.3 Time Series Forecasting Using the Stochastic Method 10
1.4 Time Series Forecasting Using the Artificial Neural Method 12
1.5 Multivariate Time Series Models 14
1.6 Forecast Performance Measures 24
1.7 Summary of Literature Review 26

2. COMPARISON OF ADVANCE MACHINE LEARNING AND DEEP LEARNING MODELS 27

2.1 Multi-Layer Perceptrons (MLP) 27
2.2 Convolutional Neural Network (CNN) 28
2.3 Recurrent Neural Network 28
2.4 Hybrid Neural Neywork 29

3 IMPORTANCE OF DATA SETS IN MACHINE LEARNING 31

4 METHODOLOGY 32

4.1 Data Pre-Processing 32
4.2 Features Extraction 34
4.3 Data Scaling and Spitting 41
4.4 Model Building 42
3.4.1 Recurrent Neural Neyworks (RNN) *42*
3.4.2 Long Short Term Memory (LSTM) *42*
3.4.3 Convolutional Neural Network (CNN) *47*
3.4.4 Random Forest *50*
3.4.5 Gated Recurrent Unit (GRU) *51*
3.4.6 Multi-Layer Perceptron (MLP) *54*
3.4.7 Vector Auto-Regressive (VAR) *57*

5 ANALYSIS AND DISCUSSION OF RESULTS 60

5.1 Experiment and Data Analysis 60
5.1.1 Banknfty Index Dataset *60*
5.1.2 Household Electricity Consumption Dataset *60*
5.1.3 Exploratory Data Analysis of Stock and Household Electricity Data sets *60*
5.2 Factors Affecting Performance of a Model 71

5.3 Evaluation Matrix 72
5.3.1 Mean Absolute Error *73*
5.3.2 Mean Square Error *73*
5.3.3 Root Mean Square Error *73*
5.4 Results 74
5.5 Results by Month Grouping 74
5.5.1 Results of LSTM *74*
5.5.2 Results of CNN *75*
5.5.3 Results of GRU *77*
5.5.4 Results of MLP *78*
5.5.5 Results of Random Forest *80*
5.5.6 Results of VAR *80*
5.6 Results by Day Grouping 82
5.6.1 Results of LSTM *82*
5.6.2 Results of CNN *83*
5.6.3 Results of GRU *85*
5.6.4 Results of MLP *86*
5.6.5 Results of Random Forest *88*
5.6.6 Results of VAR *89*
5.7 Results of Hour Grouping 90
5.7.1 Results of LSTM *90*
5.7.2 Results of CNN *92*
5.7.3 Results of GRU *93*
5.7.4 Results of MLP *94*
5.7.5 Results of Random Forest *96*
5.7.6 Results of VAR *97*
5.8 Household Electricity Consumption Results 98
5.8.1 Results of LSTM *98*
5.8.2 Results of CNN *100*
5.8.3 Results of GRU *102*
5.8.4 Results of MLP *104*
5.8.5 Results of Random Forest *106*
5.8.6 Results of VAR *106*
5.9 Outcomes of this Research 107
6 PROBLEMS DURING RESEARCH 109
7 CONCLUSIONS & FUTURE WORK 111
REFERENCES 112

List of Figures

Fig. 1 Different types of Network 14
Fig. 2 Different types of Performance Matrices 25
Fig.3 Raw Stocks Data 33
Fig. 4 Sorted Stocks Data 34
Fig. 5 Time Features Based Stocks Data 35
Fig. 6 Stocks rate from 2012 to 2016 of the recorded dataset. 35
Fig. 7 Hour Grouped Stocks Data 36
Fig. 8 Day Grouped Stocks Data 37
Fig. 9 Month Grouped Stocks Data 38
Fig.10 Household Raw Data Set 38
Fig. 11 Household Power Consumption after Feature Extraction 39
Fig.12 Voltage values Time Based Distribution 39
Fig.13 Global Intensity values Time Based Distribution 39
Fig.14 Global Active Power values Time Based Distribution 40
Fig.15 Trend and seasonality of stocks data 40
Fig. 16 Trend and seasonality of Household data 41
Fig. 17 LSTM Architecture 43
Fig. 18 LSTM Model Building 44
Fig.19 Graphical Representation of Activation Functions 46
Fig.20 Descaling Predicted Value 46
Fig.21 Convolution Neural Network Architecture 47
Fig.22 Average and Max Pooling 48
Fig. 23 CNN Model Building 49
Fig. 24 Random Forest Model Building 51
Fig. 25 GRU Architecture 52
Fig. 26 GRU Model Building 53
Fig.27 Artificial Neural Network 54
Fig. 28 Model chart of Multi-Layer Perceptron 56
Fig. 29 MLP Model Building 57
Fig. 30 VAR Model Building 59
Fig.31 Stocks Time Series Distribution by Hours 61
Fig.32 Box Plot (Stocks Time Series Distribution by Hours) 62
Fig.33 Histogram and Curve Distribution 63
Fig.34 Stocks Trend Line by Hours 63
Fig.35 Stocks Time Series Distribution by Days 64
Fig.36 Box Plot (Stocks Time Series Distribution by Days) 65
Fig.37 Histogram and Curve Distribution 66
Fig.38 Stocks Trend Line by Days 66
Fig.39 Stocks Time Series Distribution by Months 67
Fig.40 Box Plot (Stocks Time Series Distribution by Hours) 68
Fig.41 Histogram and Curve Distribution 69

Fig.42 Stocks Trend Line by Months 69
Fig.43 Household Electricity Power Consumption Ditsribution 71
Fig.44 LSTM Mean Absolute Error 74
Fig.45 LSTM Mean Square Error 75
Fig.46 LSTM Predicted vs True Stock Values 75
Fig.47 CNN Mean Absolute Error 76
Fig.48 CNN Mean Square Error 76
Fig.49 CNN Predicted vs True Stock Values 77
Fig.50 GRU Mean Absolute Error 77
Fig.51 GRU Mean Square Error 78
Fig.52 GRU Predicted vs True Stock Values 78
Fig.53 MLP Mean Absolute Error 79
Fig.54 MLP Mean Square Error 79
Fig.55 MLP Predicted vs True Stock Values 80
Fig.56 RF Predicted vs True Stock Values 80
Fig.57 VAR Predicted vs True Stock Values 81
Fig.58 LSTM Mean Absolute Error 82
Fig.59 LSTM Mean Square Error 83
Fig.60 LSTM Predicted vs True Stock Values 83
Fig.61 CNN Mean Absolute Error 84
Fig.62 CNN Mean Square Error 84
Fig.63 CNN Predicted vs True Stock Values 85
Fig.64 GRU Mean Absolute Error 85
Fig.65 GRU Mean Square Error 86
Fig.66 GRU Predicted vs True Stock Values 86
Fig.67 MLP Mean Absolute Error 87
Fig.68 MLP Mean Square Error 87
Fig.69 MLP Predicted vs True Stock Values 88
Fig.70 RF Predicted vs True Stock Values 88
Fig.71 VAR Predicted vs True Stock Values 89
Fig.72 LSTM Mean Absolute Error 91
Fig.72 LSTM Mean Square Error 91
Fig.73 LSTM Predicted vs True Stock Values 91
Fig.74 CNN Mean Absolute Error 92
Fig.75 CNN Mean Square Error 92
Fig.76 CNN Predicted vs True Stock Values 93
Fig.77 GRU Mean Absolute Error 93
Fig.78 GRU Mean Square Error 94
Fig.79 GRU Predicted vs True Stock Values 94
Fig.80 MLP Mean Absolute Error 95
Fig.81 MLP Mean Square Error 95
Fig.82 MLP Predicted vs True Stock Values 96
Fig.83 RF Predicted vs True Stock Values 96
Fig.84 VAR Predicted vs True Stock Values 97
Fig.85 LSTM Mean Absolute Error 99
Fig.86 LSTM Mean Square Error 99
Fig.87 LSTM Predicted vs True Stock Values 100
Fig.88 CNN Mean Absolute Error 101
Fig.89 CNN Mean Square Error 101

Fig.90 CNN Predicted vs True Stock Values 102
Fig.91 GRU Mean Absolute Error 103
Fig.92 GRU Mean Square Error 103
Fig.93 GRU Predicted vs True Stock Values 104
Fig.94 MLP Mean Absolute Error 105
Fig.95 MLP Mean Square Error 105
Fig.96 MLP Predicted vs True Stock Values 106
Fig.97 RF Predicted vs True Stock Values 106
Fig.98 VAR Predicted vs True Stock Values 107

List of Tables

Table 1: Pros and Cons of State of Art Models 30
Table:2. Month Grouping Results 81
Table:3. Day Grouping Results 89
Table:4. Hour Grouping Results 97
Table:5. Household Power Consumption Results 107

1. Introduction

It is irrefutable that world is connecting with billions of IoT devices like smart city sensor, industrial sensors, healthcare devices and mobile devices. A massive amount of data has been created from the IoT paradigm and tends to generate more. Hence, the real time data analysis act as the major role of data analytics and because of time series analysis came into existence. Time series analysis attracted great amount of study so far. One reason is that most of the industry sectors generate massive time series data like stock market, daily transactions, weather monitoring etc [1]. These data are gradually recognised to be extremely valuable for future forecasting and decision making in a regular time interval. In real world, many things happen at the same period therefore it means there are many variables involved at any time point. TSA is the statistical technique which controls trend analysis and time series data. Series data indicates that data will be in the series of time intervals or periods.

To tackle complex and non-stationary problem with time series in the practical life and production multivariate time series model can be used. The multivariate time series can able to solve the random factors of time series and use some factors which affect the development. It can enhance the accuracy of forecasting and prediction reliability. The multivariate time series offer various advantages to the business people and helps business grow effectively in the sales and revenues. TSA helps business to manage the stationery, non-stationary and complex problem of time series. It helps business offer effective decision making with effective intention of decision makers and act as initiative to decision makers. In addition to this, the time series forecasting helps business predict corporate earnings.

Both qualitative and quantitative methods are used for profit forecasting. Quantitative prediction comprises of factor analysis prediction and time series analysis prediction. This time series approach has applied to companies and industries which predict future sales trends and history of data like energy, medicine, electricity, goods and others. Defensive and stable companies prefer exponential smoothing method to predict. Factor analysis forecast has applied to companies and industries like tourism, materials and building, both internal and external affects the sales. They concentrate on innovations focused industries and innovative companies like information technology industry. Sales are considered as

dependent variables whereas factors involved in the industries are independent variables. By using these variables, it is significant to establish relationship between earning and factors using multivariate time series. The profit of company predicts to know production and determining effective business plan and the factors which affects the market environments and macroeconomic environment and other factors such as investment in research and development, management strategy and market researching. The profitability of the company and composition factors can be determined by using MTSA [1]. It can develop multi regression time series model over the simple statistical model to forecast company's profitability and make different departments a reasonable investment.

Time series is found in various domains like stock market analysis, medical, IoT, economic forecasting, quality control and big data systems and utility studies. Series like heart rhythm are considered as univariate and obtained from sensors are multivariate. Time series data are collected from the applications on thousands of big data system at different Fortune 500 companies. Time series forecasting methods generate the forecasts depend upon the historical values. It can be used in the business situation where forecasts are needed. These are preferred for production planning, sales, marketing and finance etc. These applications have simplicity and some factors need to be highlighted. The time series methods are effective in predicting short term forecasts. It focuses on adequate past data and the data should be high quality. This time series are useful for stable situations. The substantial fluctuations have been occurred and the conditions are associated with extreme change and provides poor results.

Time series analysis is process of collecting data at a regular interval within certain time with determining trends, seasonal variances and cycle's helps to forecast future event. Data will be observed outcome which seems to be measurable. The data should be evaluated at constant intervals to determine the patterns which create cycles, seasonal variances and trends. It is essential to notice that past performance do not predict future performance or growth. If the customer purchases 200 units last year, it does not mean they still need the same this year. There are two types of time series data has been generated in the industry which are univariate time series and multivariate time series data [2]. Therefore, in the master thesis, I am going to research on a MTSA which is challenging and at the same time interesting and practical. Nowadays, every organisation across all industry generates more big data and majority data are in multiple variables which are dependent on time. Hence, it is necessary to analyse MTSA and highly used in non-linear relationship. Therefore, advance

machine learning algorithm is used to predict multivariate time series data. In this paper, I have used all the state-of-art research algorithms which are currently using in the industry to solve multivariate time series data such as Long Short Term Memory (LSTM), Convolutional Neural Network (CNN), GRU (Gated Recurrent Unit), MLP (Multi-Layer perceptron), Random Forest and Vector Auto-Regression (VAR).

The project aims to analyse and forecast multivariate time series which can ultimately help make better decision. The scope of the research is to evaluate Multivariate Time Series Forecasting (MTSF) and determine different models or algorithms for MTSF. In addition to this, the study compares all the algorithms and proposes an optimized solution model for the analysing multivariate time series and visual analytics of different time series datasets. The study will collect and pre-process two different datasets for multivariate time series forecasting and define a concrete problem for MTSF. It offers visual analytics of different time series datasets and apply python and other machine learning and deep learning packages to get desired analysis and forecasting results.

1 Literature Review

1.1 Introduction

There are several models proposed for the time series forecasting. In this study, the previous works of models for time series forecasting have been highlighted. Industries such as finance, manufacturing, retail or e-commerce has time series data and data has fluctuated based on seasonal demand and global economy. Huge amount of data has already been generated from the IoT paradigm and it will continue generating more on the fly. Therefore, real time data analysis has become a vital part of data analytics and due to that time series analysis comes in to a picture. A significant aspect of time series problem is forecasting. This time series forecasting helps business by improving their future product demand, customer demand, sales demand and as well helps government to predict weather, air pollution or traffic forecast.

Time series are used in finance for making a strong decision in the financial department across the globe. Forecasting of data using time series is an important measure and is used for improving financial decision. The two factors which are significant for prediction in time series is non-stationary and noisy data. In numerical experiments, dataset related to economic time is stock market, oil, banking sector, information technology, and petroleum. Prediction of data set in time series is tested for its accuracy by standard deviation and root means squared error. This helps in indicating the usefulness of the proposed method, along with its application. Time series has numerical data in particular order, which are a succession of each other. The time series is used in making investment decisions as it helps in tracking the price of a security. Tracking of security is done for a short term, which involves monitoring on an hourly basis, and in the long run, it is done by monitoring the security price on the month's last days, which is tracked over the years.

There is a perfect classification and it draws the attention to the time series forecasting. This classification is shown by the artificial neural network (ANNs). Regarding time series forecasting, neural neutral work has more characteristic in compare to Statistic process. Regarding time series forecasting, there is a review of neural network (NN) has been defined

by Zhang in 2017. He said in his review that nonlinear time series forecasting has been used in this NN models (Markham at el. 2000). Moreover artificial neural network (ANN) model is the best forecasting tool working properly regarding the huge amount of applications but in the end it has to stop because it has the limitations like the techniques of black box. In AR model (Manezes at el 2006) has revealed about the structure of the recurrent neural network for establish the multi challenging forecasting of time series data analysis. In nonlinear time series forecasting the local model like artificial neural network (ANNs) and feed forward network (FNNs) can be very efficient because it can deliver the accurate result. However, there is difficult to forecast time series and it is the most meaningful aspects of time series. It helps in examine the value of the business field and the essential market related to some industry. According to Chen (1997) in short term time process of prediction can be very easy with the use of Jenkins box. It is one of the best methods to deliver a considerable and perfect result regarding time series data analysis. After Chen (1997) now it is Oztruk (2018). According to him after the use of ARMA model ARIMA model is best model to predict the market value. It is the most challenging issues in time series data that to predict the actual value of the market.

1.2 Univariate Time Series

Univariate time series has two variables namely data time and field. If person needs to forecast the specific weather field such as average temperature tomorrow, they need to use temperature of previous dates and predict an effective model for tomorrow. Multivariate time series have more than two variables. In this case, if we forecast the average temperature for tomorrow, then they need to consider all the factors like rainfall, humidity, solar radiation and atmospheric pressures. These factors are associated with temperature.

Holt Winter exponential smoothing model will be used to predict seasonal time series data. Banerjee (2014) used two models such as multiplicative and additive seasonal model. For instance, if the sales of any company increase every year in particular time frame then this type of seasonality called multiplicative and if the sales of company increase in particular time frame by any factor then it called additive seasonality.

Banerjee (2014) stated Exponential smoothing method use in specifically seasonal time series data set and in this classical method target value much depends on recent past variables than the older observations. Single exponential smoothing use for short term prediction while double exponential smoothing use for upward or downward trend time series data. If time series data has patterns of trend and seasonal then we can use triple exponential smoothing method. While performing the Holt-winter exponential smoothing method, we should test two methods such as non-adaptive test which is used first two year data to building model and establish the parameters and in adaptive test first it established the parameters using the first two years data. Kalekar (2004) concluded that adaptive method works better when it considers large amount of historical data (Kalekar 2004).

Followed by Holt exponential model, a statistical method called Theta model is useful for the univariate time series forecasting which is highly based on the second differences of the time series data. Thomakos & Nikolopoulos (2015) used this theta model to illustrate the time series data in an efficient manner. This moderation happens through Θ coefficient which is put in application directly into second difference of time series T. Therefore, if Θ is greater than one then second difference is increased and if Θ is less than one then second difference is decreased. After that modified new time series produced which is called Theta-lines and finally it will combine the method to forecast time series data. The data set has been collected from FRED database which contains data with different time frame such as quarterly, monthly, and weekly with the different parameters like GDP (gross domestic product), personal consumption and foreign exchange rate. The researchers want to predict one value in a long-term and short-term manner (Kalekar 2004).

Thomakos & Nikolopoulos (2015) stated theta model perfume better than naïve model for both univariate and bivariate data set. However, theta model performs better for bivariate data set as compare to univariate data set (Thomakos & Nikolopoulos 2015). While performing theta model, less volatility affected to forecast but length of series and frequency observation does not affect.

Chen (1997) found that Box-Jenkins method works very well for short-term prediction and its transfer function approach consider as one of the robust methods for accurate result. However, every time series data has forecasting error variables and without counting them

the Box-Jenkins model does not give accurate result. Chen used ARAM model for obtaining accurate results. Therefore, to break that limitation author proposed novel approach called adaptive ARAM (Auto Regressive Moving Average) model which can be created for the short-term prediction of power system. This adaptive model first used minimum mean square error (MMSE) theory to determine error learning coefficients and after that upgrade forecast with those error and coefficients which is based on one step forecast method. With this adaptive approach, ARAM model can able to deal with uncommon system condition. Researcher applied this adaptive ARAM model and compared to Box-Jenkins method. Thus, the result shows that adaptive ARAM model outperform Box-Jenkins method for twenty hours and one week ahead forecasting.

Followed by ARAM model, ARIMA model has been used by Oztruk (2018). Predicting stock market price is one of the challenging problems in the space of data science. It will help the stakeholders and investors to review the present situation of the companies and it will assist them to analyse right time to invest money. In this paper, they collected data of sensex (Bombay Stock Exchange) which is aggregation of top 30 Indian stocks from 2007 to 2012. They tried to predict closing stock price of sensex so therefore first they checked whether data is suitable for this purpose. So that, researchers used Durbin-Watson test which gives us vivid impression about the data set which helps to identify auto-correlation between variables. To predict the future stock price, they used probabilistic model ARIMA model which stands for Auto Regressive Integrated Moving Average (Banerjee 2014). To get the desire output, they checked Auto Correlation function (ACF) and Partial Auto Correlation Function (PACF) and then verified whether time series is stationary or not.

ARIMA has emerging variables of the interest and used past values or observations. MA implicates that linear integration of error terms where the values occurred at different times in past. Integrated values represent that data values can be replaced by the difference between present and previous values. The objective of the model is to make a model fit the data and this model is based on the accuracy through time series. Vector auto regression model acquires the interdependencies over different time series in data. It is generalization of univariate model. Long short-term memory (LSTM) model is a form of recurrent neural network used to make sequence dependence. It can handle large architectures at training session.

Renewal energy is one of the essential aspects for Turkey as it dependent on the fossil fuel. Therefore, it is necessary for Turkey to start producing clean energy to minimise the fossil fuel. Hence, they collected renewal energy consumption data from OECD and while oil, gas and coal data extracted from the Turkish government department. Total energy consumption of per capita data is achieved from World Bank and multiplied with population to get total energy consumption. With the help of ARIMA model (autoregressive integrated moving average), they tried to predict total energy consumption for the next 25 years.

In ARIMA (p, d, q) model, 'p' represents autoregressive process for AR parameter, 'd' represents order of difference to get stationary series for I parameter and 'q' represents the moving average order for MA parameter. Hence, autoregressive settle regressive variables of their prior terms. I parameter is generally use the non-stationary time series. Therefore, if time series is stationery then value of d becomes zero and for first difference stationary the value of d becomes one. In the study of Ozturk & Ozturk (2018), it is highly addressed that moving Average express that the observations are linearly dependent of its present and past values. After applying the ARIMA model, the result shows that energy consumption rate will be increased by 1.64% annual rate and it gave good accuracy as well.

People of Africa livelihood mainly rely on the production of Coffee. Coffee is the most valuable commodity in the world. Harris et al (2012) wanted to study the behaviour of coffee production and want to predict that next five years production would be increased or decreased. They took data from Ghana cocoa board from 1990 to 2010 and use ARIMA (autoregressive integrated moving average) algorithm to predict the future which is based on Box-Jenkins method. ARIMA model is an integration of Autoregressive Model and Moving Average Model. Before applying ARIMA model, first they checked whether TSA is stationary or not after that they checked three estimation such as Identification, estimation and diagnostic checking. To apply best model, they checked significance level which is 5%. As we can see that, coffee production will be decreased for the next five year (Harris et al. 2012).

Like ARIMA model, the ARIMA auto model has proposed by Jain and Mallick. Jain and Mallick (2017) found that weather prediction is a challenging task as it has noise, outliers

and unclear structure. In this paper, they applied automated ARIMA (autoregressive integrated moving average) and Automated ETS (exponential smoothing) model are used to forecast different parameters and compare them with estimation of MAE (Moving Absolute Error), MASE (Moving Absolute Scaled Error), MAPE (Moving Absolute Percentage Error) and RMSE (Root Mean Square Error). The result shows that auto ARIMA model outperform auto ETS model. However, these models perform accordingly different parameters and condition for instance auto ARIMA is good for predicting non-linear and unclear structure TSA data while auto ETS model is good for seasonal time series forecasting (Jain & Mallick 2017).

We should consider seasonality while tourism demand forecasting and it affect directly on tourism business and local community as well. However, if we could plan it better and make better prediction for future tourism business then it will be beneficial for the local people, enterprises and the government. With the better prediction, companies and government could decide good strategy which can be helpful for various people including local community and tourist as well. Hence, seasonality is vital for the tourism demand forecasting. Therefore, Seasonal Auto Regressive Model is very popular to predict seasonal demand while significant popular model for seasonal time series data is Structural Time Series Data (STSM) which can be decomposed time series into seasonal, trend and irregular component. Chen et al (2019) proposed Seasonally Restacked Multi series STSM which is restacked time series into quarter from the original quarterly time series component which is in the form of vector. Each component extract from the time series which gives strong justification for the multivariate time series method. In this way, the seasonality should not be modelled separately which keep away potential misspecification and remove the source of inaccurate forecast. The data collected from visitor arrival statistics which is published by Hong Kong Tourism Board and it include time series data of tourism demand from top 20 countries. The results show that restacked structural time series model outperform other classical model and STSM gives very good result in term of long-term prediction (Chen et al. 2019).

To get the intelligence transportation system we should need short-term traffic flow prediction which is essential for ITS. With the accurate and timely traffic data we could control transportation system. Since, the univariate time series has some limitation to get accurate result we should use hybrid model which has integration of statistical analysis and

computational intelligence approaches. Zhang et al (2018) used wavelet de-noising technique to deal with the noise information so that time series forecasting helps to analyse time series variables and periodic features of traffic flow. After that, auto-regressive model is used external input which is initiated to fit traffic flow with occupancy as exogenous variables. Thus, WSARIMAX is built to forecast traffic flow and which is outperform SARIMA model in accuracy. Zhang et al (2018) shows that WSARIMAX and SARIMAX gave better result as compare to traditional SARIAM method.

There are many problems associated with the ITS such as pollution control, congested traffic, road safety and so on. In the past, many several models used to solve same problem. Many used different algorithms such as historical average method, time series analysis, K-nearest neighbour, support vector regression and artificial neural network algorithms (Zhang et al. 2018). But there is some imitation to get desire result from these algorithms. Zhang et al (2018) implanted hybrid model to solve this problem because many factors involve with this forecasting and univariate time series method cannot give better result with the missing values therefore multivariate time series hybrid model gave good result as compare to previous models. Hybrid model is a combination of one or more models which is good for linear and non-linear features of traffic flow and it improves accuracy of forecasting of traffic flow. Apart from this, multivariate time series consist of some external variables which are more accurately predicting compare to univariate time series analysis. The quality of data is vital for prediction nevertheless the current hybrid model SARIMAX and SARIMAX is performed better with this issue (Zhang et al. 2018).

1.3 Time Series Forecasting Using the Stochastic Method

Autoregressive models:

This model is mainly used for predicting future behaviour based on previous or past behaviour. Whenever there is a correlation found between the values in a time series and the values succeed them the autoregressive model is used for time series forecasting. Only the previous or past data is used for modelling the behaviour and that is why it is called ‘autoregressive'. The process follows the linear regression of the current series of data against the past values present in the same series. The model depends on the sums of past values.

Moving average models:

The Moving average model is another approach used for modelling univariate time series. The main purpose of this model is that it specifies that the output variable is dependent on the current and previous values of an imperfectly predictable term.

Autoregressive moving average models (AMRA):

Box and Jenkins (1970) discovered a more effective and systematic method fitting and identifying a combination of the AR and MA process. The process was originally investigated not by them but by yule.

There are two major components of AMRA; they are autoregressive component (the sum of previous values) and moving average component (the sum of previous errors). The AMRA model is stationary according to the stationary condition of both AR and MA processes if the root of its polynomial is zero. The most important AMRA model' assumption is stated as the mentioned time series is stationary. Box and Jenkins recommended differentiating the time series in order to achieve the stationarily if the series are not stationary. The acronym 'I' in the AMRA model stands for integration or integrated. It can be also shown that a stationary process can be termed or represented as MA(∞) model. The box and Jenkins model can be extended for including the seasonal moving average and seasonal autoregressive terms if the related series is seasonal. Long series is needed to fit an AMRA model more effectively.

According to CHATFIELD (2013), it was recommended to use a minimum of 50 observations, while some other authors also recommended using at least 100 observations. The AMRA model is also important as it can reduce the number of parameters for contrasting the use of pure MA and AR process. This indicates that the use of the AMRA process is more effective than the use of AR or MA process individually. The AMRA process is always preferred by the forecasters for time series forecasting. The three stages which are used to build the AMRA model are the identification of models, fitting or estimation and validation.

According to Zhang (2003) during the past decades, the autoregressive moving average (ARIMA) is proved to be one of the most popular models for time series forecasting. Some reports show that the ARIMA model could be more promising in the future and can be used as an alternative to traditional methods. ARIMA and ANN are compared with some mixed conclusions in forecasting performance. Here the methodology combines both ANN and ARIMA models. It is proposed to take the advantages of both ARIMA and ANN models in nonlinear and linear modelling. The experiment shows a positive result of the combined use

of both the models as it proved to be a more effective way of improving the forecasting accuracy.

The use of a univariate forecasting method is only effective if the skill of the analyst is limited if there is a large number of series for forecasting and if the forecast should be made of explanatory variables required by the multivariate method. The difference between model and method is that a model is a mathematical representation whereas methods are the formulas or rules used in forecasting. According to CHATFIELD, forecasting methods should be more focused. For evaluating a forecast the analyst defines a loss function, where ‘e' denotes a forecast error. e is also valued as (observed value – forecast). L(e) is the loss function, the loss function specifies the "loss" which is associated with the forecasting error of size e and it also has two properties: L(0) = 0 and L(e) is a continuous function. L (e) increases with the total value of e.

The mean square error shows or implies some quadratic loss function. According to CHATFIELD, the minimum mean square error forecast is the best forecast. Some problems also arise so the analyst must know the spectrum or the functions of the exact correlation. The past data should be collected in an infinite amount by the analyst. CHATFIELD referred to concentrate on mean square error for finding the best point forecast. However, in order to access the uncertainty of the future, it might be convenient to look for the interval forecast but not point forecast. Furthermore, a distinction could be made between the single-period forecast. Forecasting the sum of values over a sequence of periods could be used as an alternative. The cumulative forecast or cumulative sum of forecast errors is the measurement of the entire forecast error. Reviews of dynamic forecasting define that after the first period the forecast must be collected for the periods by using the previous left-hand variable values of the forecast.

1.4 Time Series Forecasting Using the Artificial Neural Method

Artificial neural networks (ANNs):

The main objective of the ANNs was the construction of a model that can be used for the mimicry of intelligence of the human brain into a machine. ANNs try and works in recognizing the patterns and regularities available or stored in the input data, and provides general results on the basis of the previous knowledge. Some of the important features of ANNs are as follows:

ANN is self-adaptive in nature and it inherently nonlinear feature brings more accuracy in modelling complex data patterns. According to Hornik and Stinchcombe, artificial neural networks are universal function approximates.

Multi-layer perceptrons (MLPs) are one of the most popular and widely used ANNs for forecasting problems. A single-layered FNN (feed-forward network) is used by multi-layer perceptrons. They are three networks of layers that have been characterized in this model is hidden, input and output layer.

Time lagged neural network (TLNN) also known as time-delay neural networks (TDNN), its main purpose is the modelling of the context in each layer of the network and classification of the patterns with respect to shift-invariance.

Here the input nodes are known as successful observations of the related time series. Seasonal artificial neural networks (SANN). This structure is accepted by HAMZACEBI (2007) for the improvement of the performance of the forecasting of ANNs for the data related to the seasonal time series. Any kind of pre-processing of raw data is not required by the seasonal artificial neural networks. SANN can learn the seasonal pattern available in the series without any removal. According to HAMZACEBI the SANN is having a great time series forecasting abilities. The seasonal parameter of this model ‘s' is used for determining the number of input neurons as well as the number of output neurons. All these features and abilities make the model simple and good for implementation and understanding.

Above mainly three important networks have been discussed (FNN, TLNN, and SANN) which are widely used for forecasting problems. The desired network model should always produce small reasonable errors within the sample data and also out of the sample data. As a result, immense care is must needed for the selection of the hidden neurons and the number of inputs. There is no availability of such theoretical guidance which could help in the selection of required parameters. Network parameters are another problem that results in the overtraining of data.

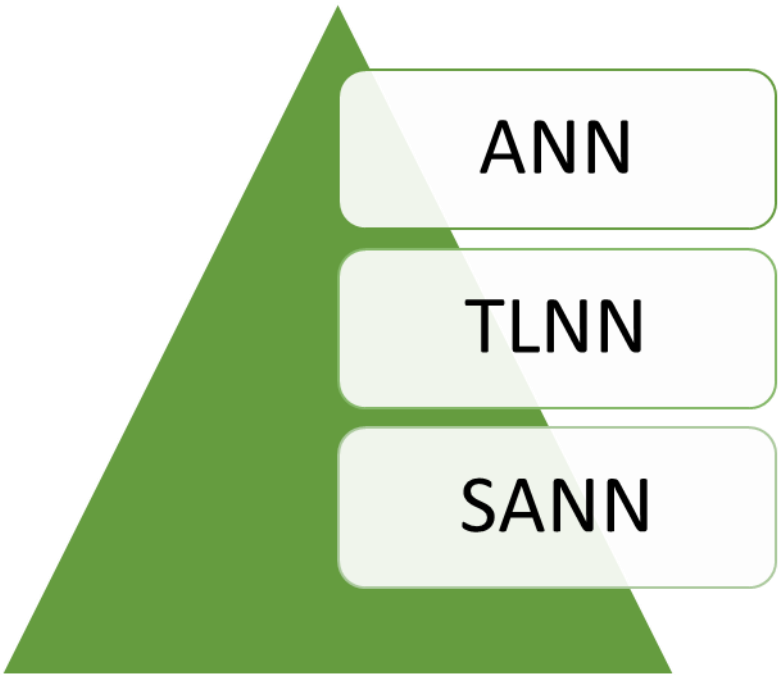


Fig. 1 Different types of Network

1.5 Multivariate Time Series Models

In the multivariate time series more than one-time dependent variable is contained in the dataset as it considers concurrent multiples of time series which deals with the dependent data. Time is an important factor that decides the ups and downs of a business. It is important to study and understand past related historical data for increasing consumer spending. The multivariate time series model depends on some previous values and also on other variables.

There are several models that can be used in multivariate time series forecasting.
According to Thomakos & Nikolopoulos (2015) copula model gives a flexible representation in the distribution of multivariate and the dependent structure is also accounted for by the copula model. In this model, the distribution of the marginal probability is uniform for each variable. Copula models are used for describing the dependence between random variables.

It is used in quantitative finance for minimizing risks. The model is more suitable for multiple time series and non-linear variables. When the time series are continuous and stationary copula works well by checking the seasonality, trend, and auto-correlation within the time series.

Jian et al. (2012) states that the auto-regression model is a time series model that uses past observations and its steps as input in the regression equation for the prediction of values for the next time steps. The AR model is mainly used to describe and define the time-varying process of economics, nature, etc. The auto-regression model shows that the output variables depend on its own past values and on an imperfectly predictable term. The infantile model of this section is very suitable in it but it is more suitable in the relation of some based variations of time series. Multivariate time series helps to calculate the exact production of an industry and it also helps to make the decision regarding the returns of the stock market. R vine is related to COPAR and with the help of this Researchers examine the influence rate of statistic solution.

After identifying the significance of hybrid model, the COPAR model has been studied. Multivariate time series forecasting is essential for economics and finance domain. Vector Autoregressive model is better suitable for the multivariate time series forecasting but it is suitable for only linear relation and symmetric dependent variables. Brechmann (2015) found novel copula-based model which is suitable for non-linear symmetric variables and multiple time series dependencies. It will help to find accurate the industrial production, interest rate and stock market returns. Researchers want to predict the effect in inflation rate using COPAR model which is based on R-vine structure and statistics inference. They took data from US economic website which is from December 1956 to December 2000 and it has four parameter industrial production rate, annual interest rate, real returns of S&P 500 and compounding annual rate. On these observations, they want to predict inflation rate will be affected in the future or not. After applying the COPAR model, he found that inflation rate will not get affected immediately for industrial rate and interest rate but it will receive immediate result for stock market return (Brechmann 2015).

In this study of Brechmann (2015), stock investment is one of the debatable questions for the people with diverse background. People are not easily do investment in this volatile market where we do not know how market will react in the future. Stock market is a

backbone of a capital market and it is very different from the traditional bank market. Without stock market we cannot imagine capital market and therefore stock market which is operated by different stock exchange is the heart of any country's economic growth.
A stock exchange is a system to sell and buy stocks of the company through various platforms and it is connected to the company's annual growth. Someone make money and other could loss as well and it is the way how business works. For an individual it is hard to predict stock prices so that people generally study past values of stock price and after analysing they decide to invest but it is not the mathematical way to get accurate result so that TSA is the significant way to predict stock prices in a statistical and scientific way.

To analyse dependent variables with a certain tools and techniques the process is known as time series. In a time-series, all the dependent variables are measured with a regular interval with repeatedly time interval. In addition to that, time series data classify into two category univariate time series which is impacted by the single observation while multivariate time series which is impacted by multiple factors and variables (Tsantekidis et al. 2017). Therefore, in many cases univariate time series analysis occurs some limitation while multivariate time series analysis gives us an accurate result, multivariate time series analysis with multiple interrelation variables can predict future stock prices of different banks in Nigeria.

From the literature review, we can say Tsantekidis et al (2017) used several techniques to use analyse multivariate time series forecasting stocks prices such as regression and data mining for the exploratory time series financial data, other used vector autoregression to predict future stock prices, and principle component analysis (PCA) for the volatile market. Apart from that, many other used non-linear tools to find the relationship between in artificial and real-time financial data while other used Auto-regressive Integrated Moving Average (ARIMA) and Long-short term memory (LSTM) to predict stocks price with deep neural network. Thus, they used Vector Autoregressive model which is based on vector of the component series of a correlation structure and it is found to be useful to predict future stocks prices of Nigerian banks (Tsantekidis et al. 2017).

Xie et al (2017) wants to forecast an energy consumption of China with the autoregressive model. Here, the sample size is a small and researchers used the auto regressive model with

the group method of data handling (GDHM) which is use for small population and proposed GDHM based auto regressive (GAR) model. This model manages all the process by itself such as determining the optimal complexity, finding the auto regressive order, and determining the parameter. The researchers analysed three different time series such as total energy consumption, total gas consumption and total oil consumption. The authors also compare GAR model with the other models such as ARIMA model, neural network, support vector regression and GM model. But GAR model outperforms the rest models.
The data has been collected from China's statistical yearbook and it contained oil, gas and energy consumption data from 1978 to 2013 (Xie et al. 2017). GAR model predicted that China's energy consumption oil consumption and gas consumption rate will be increased by following 6.99%, 6.84% and 31.44%. GAR model works well with the linear time series data set while prediction will be difficult with non-linear time series data set.

Transport industry now moving towards from fossil fuel to electric vehicles. China is the leading country in the adoption of electric vehicles. Automobile prediction has become vital among practitioners so that they use different techniques such as support vector machine and linear regression. For time series forecasting, we should consider different method as EV data ser include different parameters such as GDP (gross domestic product), interest rate, unemployment rate, CPI (consumer price index), gas price. Apart from that, charge point, revenue tax, subsidise amount, emission rate considers while forecasting. Here, researchers found that univariate time series forecasting model outperform MTSA prediction for short term period (Zhang et al. 2017). While other believes that there is not significant difference notice between those two models.

Zhang et al (2017) use SSA (singular spectrum analysis) model for univariate time series forecasting which gives better result in comparison conventional univariate time series model for specifically financial analysis. Because SSA does not affect by linearity, seasonality or normality. Zhang et al (2017) used VAR model for multivariate time series forecasting which is an integration of auto-regressive model and Box-Jenkins model.

SSA model works on four parameters such as transformation, decomposition, groping, and reconstruction. While VAR model works on stationary time series data and then ganger causality test explained about relationship and ultimately VAR model build on recursive

method. The data collected from energy savings and new energy vehicles yearbooks from January 2011 to December 2015. They used SSA model and VAR model on the following battery electric vehicles and plug-in electric vehicles. The result shows that VAR model outperform SSA model in terms of accuracy and results suggest that EV demand will be increased in the future. However, producer and consumer price, fuel and vehicle price and consumer confidence could be affected for prediction. Nevertheless, result shows positive future for EV in China (Zhang et al. 2017).

Lora et al (2003) applied KNN statistical algorithm to predict 24 hours load forecasting problem with in electricity domain in Spanish which is time series problem. They also used conventional dynamic regression method to solve this problem by solving least square problem in the parameter estimation. KNN is a classification problem used for determining the similarity of the individuals of a population. Each member of the population has grouped by similar pattern or characteristics. KNN algorithm consider training set as a model itself which characterised by number of neighbours and Euclidean distance. While dynamic regression model used to predict hourly electricity demand which is based on linear regression (Lora et al. 2003). The results show that dynamic regression model applied to those parameters which is obtained by solving least square problem and KNN algorithm applied to predict Spanish short-term electricity load forecasting problem which has analysed six-month period data. After comparing both algorithms, it is clearly seen that KNN classification algorithm perform better to predict load forecasting problem as compare to traditional regression algorithm (Lora et al. 2003).

Montori et al (2018) stated there are massive amount of data created from the last several years and with the IoT technology mare data will be generate. Because of these data will be create in a real-time we should have some techniques or algorithm so that it will classify information into different sectors. Therefore, Montori et al (2018) talked about two different algorithm CBOS and TKSE and they claimed that the result from this algorithm is much better than the traditional classification algorithm.

Due to the billions of IOT device connection, more data has been produced and it is also available on internet to access for people. And there are various platforms which is provide this kind of open data which is powerful IOT application for different domain such as healthcare, observing environment, smart cities, defence, and so on (Montori 2018). But

these all IOT open data is not correctly classify within their metadata and there is lack of availability of accurate metadata. Internet of things is the process of interconnecting the computing devices, digital machine and mechanical machines, objects or people which are rendered with unique identifiers. This system can able to transfer the data over the network by replacing human to computer or human to human interaction. Many industries are using IoT to access effectively and enhance decision making process. Hence, multivariate time series uses IoT devices to make effective decisions in the companies. Therefore, this paper is trying to solve the open IOT data-stream classification issue which is not very well discus in the literature (Montori 2018). So that, this paper focus on specifying streaming data into their relatively Meta data. Therefore, they are using two algorithms A novel Class-wise Bag of Summaries (CBOS) and A novel Top-k Sequential Ensemble (TKSE) approach which is highly depends upon generic data mining and machine learning. With the CBOS approach, it went successful to classify the IOT stream data with numerical characteristics and it performed well as compare to traditional TSC approach while classifiers like decision tree and random forest gave good results with bag-of-summaries method in numerical features of IoT data stream (Montori 2018). The second TKSE model also performed well for partially available text data that implies as IOT data stream and the numerical features of the IOT data stream. However, we can improve our model by improving annotation quality with a specific feature engineering technique (Montori 2018).

Multivariate time series forecasting is an essential task for various domain including solar plan energy output, electricity consumption, transportation etc. Lai et al (2018) discuss about neural network-based model including RNN (recurrent neural network) and CNN (convolutional neural network) which is very effective as compare to state-of-art autoregressive models. Multivariate time series data has created enormously from different sector such as daily traffic data, solar plant energy data, smart cities, stock market prices, temperature of cities, just to name a few. Multivariate time series usually faces challenges in research which mainly on how to detect and leverage the observation among various factors. Precisely, real-world application is often combination of Short term and Long-term repeating patterns, daily, weekly, monthly. To get an accurate result, we should focus on both long-term and short-term observations for instance the output of solar energy farm based on various factors such as long-term factors like days vs nights, summer vs winter etc., while short-term factors like cloud movement, wind direction etc. However, in this type

of case autoregressive model cannot work efficiently (Lai et al. 2018). Therefore, the researchers claimed that deep neural network is very well worked for this kind of multivariate time series data. In addition to that, in RNN (recurrent neural network) two methods works very well namely LSTM and GRU (Gated Recurrent Unit).

Lai et al (2018) used deep learning approach for forecasting multivariate time series data namely, LSTnetwork (LSTNet). It is a combination of a convolutional layer to identify the local dependency patterns within multi-dimensional input variables and the recurrent layer to identify long term dependencies. Moreover, LSTNet model build on top of autoregressive model with a non-linear neural network part which makes non-linear deep learning model become more robust for the multivariate time series forecasting (Lai et al. 2018).

Stock market price prediction is very useful for the mutual fund and venture capital companies for their future investment. It is an important and challenging financial problem for the entire practitioner. Nowadays, artificial neural network has got attention to predict stock market price and Guresen at al. (2011) found that ANN works better with MLP (multi-layer perception), Dynamic Artificial Neural Network (DAN2), and hybrid model which based on autoregressive conditional heteorscedasticity (GRACH). Here, researchers compared these models on two points: Mean square error (MSE) and Mean absolute deviate (MAD) and they took NASDAQ stock exchange index from 7 October 2007 to 26 June, 2009 (Guresen et al. 2011).

MLP (multilayer perception) is one of widely used neural network in the artificial neural network topology which is based on backpropagation algorithm. Dynamic artificial neural network compressed with the classic ANN and autoregressive conditional heteroscedasticity (ARCH) current variance error term which is depend on the past variance error term. The result shows that the MLP (multilayer perception) with ANN outperform both DAN2 model and GARCH model. Simple MLP with ANN works better and gives very practical output (Guresen et al. 2011).

Aizenberg et al. (2016) applied Multilayer Neural Network with Multi-Valued Neurons (MLMVN) for long term time series predicting which is based on derivative free back propagation learning algorithm. Here, researchers predict behaviour of oilfield asset in Mexico gulf and they showed that MLMVN efficiently apply for the univariate and one step multivariate time series forecasting and specifically use for pattern recognition.

MLMVN is combination of Multi-Valued neurons (MVN) and complex-valued weights which is different from the classical feed forward neural network. Neural network weights, input-output neurons and activation function helps to improve single neural network and reduce processing time therefore ultimately it improves the neural network performance. Here, the results show significant improvement when compare to other methods. MLMVN works well for both univariate time series and one step multivariate time series forecasting for the next 15 years (Aizenberg et al. 2016). MLMVN model used regression and classification approach to predict the results with the help of local variables and which gives better output as compare to conventional method (Aizenberg et al. 2016).

Corchado at al. (1998) applied unsupervised learning model to predict financial index. Authors claimed that unsupervised learning model outperform other multilayer perception neural network model, statistical ARIMA model and Radial Based Function Network (RBF). The researchers took data set of Dow-Jones Index from 30th November of 1988 to 15th of May 1985. Corchado at al. (1998) used novel negative feedback artificial neural network for predicting and determined the integrating functions of time series. In this unsupervised learning approach, the authors used Finite Impulse Response Hebbian Model which is used negative neural feedback. In this method, input data fed into left side and after some time it passed into further neurons and the actual values of input are a function of the determined time series. Here, the result shows that FIR unsupervised ANN model opt perform the other ARIAM, MLP and RBF models. And FIR model is faster to train as compare to other models (Corchado et al. 1998).

Intelligent transportation system has become important part of any smart city project and now we are moving into the world of big data where we should collect those sensor data and analyse it faster pace. Existing traffic flow prediction model has some limitation as it does not capable of analyse real-time data. Therefore, authors applied the deep learning method stacked autoencoders (SAEs) traffic flow prediction. Researchers claimed that here they used first time deep learning approach which used stacked auto encoders understand the characteristics of genetic traffic flow and it is trained layer wised in a certain way (Lv et al. 2015).

The SAE model has converted the stacking auto encoders into the deep neural network environment. In this process, first input layer passed into second layer and output works as input for the next layer. Apart from this, author also used logistic regression on top of neural network layer for supervised prediction. The data collected from Caltrans Performance Measurement System (PeMS) database which has collected every 30 second from 15000 different sensors. The model trained by BP method in the supervised way with gradient based optimisation technique (Lv et al. 2015). They also used the unsupervised learning algorithm to pertain deep network and after that he enhanced the process to make model parameters to improve model's performance to predict better traffic flow. They also compare this method to classical SVM (support vector machine) algorithm but SAE model gave impressive results.

Uber used state of art algorithm to improve their ecosystem and which is benefited to their drivers by reducing wait time. These algorithms useful for especially in holiday seasons or any sports event to find anomaly detection, resources allocation and budget maintenance.

In this paper, Laptev at al (2017) applied LSTM-RNN architecture to predict accurate complete trip in a specific time-period such as festivals or sports event. There are several obstacles have become building blocks to get desire output such as population growth, weather or marketing change which could affect while forecasting.

The data collected form riders and driver's mobile application which is stored on Uber database. Apart from this, Laptev at al (2017) consider different parameters such as holidays, city level information, weather information and current traffic flow. To predict, they used novel architecture to provide single model for time series forecasting and initially features extracted automatically with some ensemble techniques. Then final vector output data works as input for LSTM model to forecast. After that authors add external features by auto encoders method. They used AWS and GPU to train those time series data set due to it large data set (Laptev et al. 2017).

The result shows 2% to 18% prediction accuracy as compare to statistical model. The researchers claimed that there are three main reasons to use deep learning technique to forecast TSA. First factor is the number of time series data, second will be length of time series data and third is correlation among different time series data. Therefore, if we find those types of characteristics in our time series data set then we should use deep neural

network approach otherwise statistical model such as VAR or ARIAM works best for it (Laptev et al. 2017).

Wenbin & Mugen (2017) wants to predict wind power forecasting which is useful for production planning for wind farm and to guide the dispatching the grid. So that, the researchers used K-means clustering with bagging of neural network model to predict short term wind power forecasting. Neural network gives better results as compare to other classical model for short time period. Neural network works perfectly for non-linear relationship data between historical data and forecasting power (Wenbin & Mugan 2017). Neural network used historical data as input data like speed and temperature to evaluate wind power. The whole neural network process is optimised by back propagation model. While K-means clustering is helpful to determine the wind speed and classify into different categories.

The result shows that accuracy of wind power forecasting is improved as compare to other models and the authors obtained mean absolute error (MAE) and root mean square error (RMSE) less value which indicate that we could improve model accuracy. Thus, in this approach, Wenbin & Mugen (2017) used K-means clustering to identify the wind speed and historical data and used three -layer neural network and neural network variable are achieved by the algorithm to minimize complexity (Wenbin & Mugan 2017). Bagging algorithm is used to improve stability and accuracy of back propagation neural network (BPNN). Therefore, researchers compare three models first forecasting with neural network, second forecasting with back propagation neural network with clustering and third forecasting with bagging back propagation with clustering. So that, Bagging BPNN with K-means clustering gives better results as compare to other models.

Generalized Autoregressive Conditionally Heteroskedastic Models (GARCH): AR application in time series variance is referred as ARCH, and when the same thing is applied to ARMA model it is referred as GARCH (Davis *et al.* 2018). The variance of squared errors or residuals is referred to as the AR model. The Formula for GARCH model is:

$$\sigma^2(t) = a*\sigma^2(t-1) + b*e^2(t-1) + w$$

The model is regarded as unstable when (a+b) is less than 1. The heteroskedastic process has substantial evidence through successive lags decay. The importance of lags in PACF and ACF indicates that MA and AR components are needed in the model (Dias, Vermunt & Ramos, 2015). The parameters which are true fall in intervals of respective confidence. The GARCH model is used in the financial time series. The process involves different steps which are stated below:

- Repeat the combination used in ARIMA models so that the best fit can be implemented in time series.
- The orders of the GARCH model are picked up according to ARIMA by considering the AIC. The lowest AIC is selected to be used in GARCH.
- GARCH model is fitted in time series.
- Autocorrelation is achieved by square residual examination and model residual

GARCH and ARIMA model are used on an everyday basis for finding best fit and which can be further used for T days. The combination of these two models is used for making predictions related to return the next day (Gong *et al*. 2016). The predictions are used for making purchase decision of stocks if the predictions are identified as positive stocks are purchased, and in case negative predictions are made, stocks are shorted at the end of day.

1.6 Forecast Performance Measures

From the above techniques, we have learned many popular and useful techniques of the "time series forecasting". Next is the implementation. It helps in the application of the methods to get the output in a general forecast. To calculate a time series two types of data are needed to be calculated one is test set and another is time set. There is also a sub-part of training set which is called Validation Set. Sometimes this is done by formalizing the data. One such technique is known as "Box-Cox Transformation". Once it is constructed it is used for generating the forecasts. The observations are therefore kept for generating the forecasts later on. It is also used to see how the exact value is coming as an output.

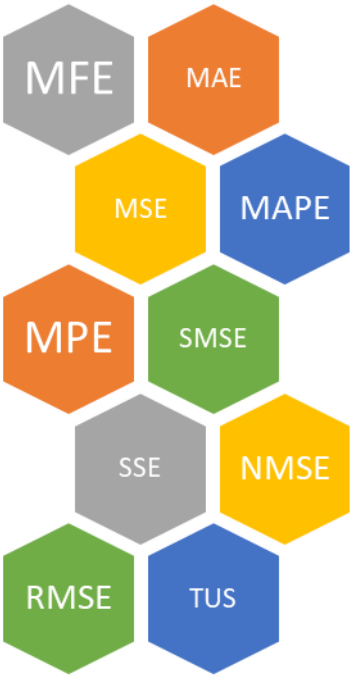


Fig. 2 Different types of Performance Matrices

Proper care must be taken in forecasting a specific series or a particular model. Therefore many of the measures are proposed to estimate the accuracy of the forecast. It also helps in the comparison of the different types of the model. It is also known as "performance metrics". (Botchkarev, 2018). There are various types of measures such as:

- "Mean forecast error" (MFE)
- Thief's U statistics
- "Mean absolute error" (MAE)
- "Mean Squared Error" (MSE)
- "Mean Absolute percentage error" (MAPE)
- "Mean Percentage Error" (MPE)
- "Signed Mean squared Error" (SMSE)
- "Sum of Squared Error" (SSE)
- Normalized Mean Square Error (NMSE)
- "Root Mean Squared error" (RMSE)

1.7 Summary of Literature Review

Time series are used in finance for analyzing the data set of a particular sequence in a specific interval of time to make decisions related to finance. In the implementation of different models of time series analyst find it challenging to select the appropriate models. The data which are available are not accurate, and results which are achieved are not perfect. This makes the predictions unreliable. Financial forecasting must be careful as decisions regarding the investment are taken based on results of time series. The data which are taken for time series are based on an assumption which does not help the analyst in making proper predictions. The uncertainty of the future and hypothesis together make the result of time series challenging to rely upon and make financial decisions (lcs.poli.usp.br, 2019).

It has given the proper statement about the univariate time series and the other multivariate time series here in this side discussed the role of univariate time series model that how the univariate time series model helps and plays the role in analysing the stationary. Univariate time series is a term that is an observation that is equal for the interval of time in the time series forecasting. On the other side, the Multivariate time series has stated that how the multivariate time series is dependable on past values and business. On the above giving statements, there are some points relating to Time series forecasting using artificial neural methods has been given. In this section, it is said about the alternative techniques of the time series forecasting which are suggested to use. This approach has gained a lot of popularity in the last previous years. Here in this part, (ANN) artificial neural network has been analysed as that the objective of the ANNs is to the construction of a model that can be utilized for human beings for their intelligence which is transferred a human brain into a machine. After this forecast performance measure has been described. Forecast performance is to help the application in the way in which it may get source output to a common forecast. It helps in the application of the methods to get the output in a general forecast. It is used to calculate the time series data one is called test set and the other one is time set. All the description about time series forecasting has been given in this project which is related to their points and models.

2. Comparison of advance machine learning and deep learning models

Deep learning serves significant for the enterprises and government to enhance the services and business used for unstructured data. The artificial neural network application is based on neuro science. Researchers proposed several models for the modification to existing neural network model. There are some models used in real time for time series forecasting.

- Recurrent neural network
- Convolution neural network
- Multilayer Perceptrons
- Hybrid neural network

These network classes offer flexibility and proved to be reliable and useful in various problems. It has various subtypes to makes them different framings of different dataset and prediction problems.

2.1 Multi-Layer Perceptrons (MLP)

MLP is the classical type of neural network which has one or more layers of neurons. Data is fed into input layer and output layer offer predictions and other hidden layers render abstraction. Output layer is also called as visible layer used for the classification prediction problems in which inputs are assigned to label. It is used regression prediction problems in which real value is used to predict set of inputs. Data will be given in a tabular format and can be in form of csv or spreadsheet file.

Classification and regression addiction problems are reliable and flexible to map from input to output. It is applicable for different types of data. Pixel of the image and word document will be reduced to one long row of data. For time series, the prediction problem is reduced to long row of data. If the data is on a tabular dataset like document, image and time series and implicated to prefer MLP on problem. The outcomes are used as baseline for comparing to other models which may use for adding value. Rolling forecast approach is used and known as walk forward model validation. Each time step will be given one at time. This

model is used to predict a time step and actual expected value will be used and forecasted on next time step. It has real world scenario where the observation available on each month and used in forecasting. Test dataset will be gathered and error score used to compute the skill of model. Root mean squared errors is used to solve large errors and resulted in the score which predicts data (Hontoria et al. 2002). Multilayer perceptrons prefers text data, time series data, image data and other forms of data

2.2 Convolutional Neural Network (CNN)

This model is used to map image data to the output variable. It is so effective for the prediction problems includes image data like input. Advantage of CNN is to create internal representation of two-dimensional images. It makes model to scale in different structures which seems to be significant while using images. CNN use for image data problem, regression prediction problem and classification prediction problems. CNN has spatial relationship which works with data. However, the input is two dimensional or matrix it can be reduced to one dimensional. It tends to create internal representation of single dimensional sequence. It allows using different types of data. There is association between words in documents. This ordered relationship makes a time series. It is created for image dataset but CNN can attain state of art found on sentiment analysis and document classification (Tsantekidis et al. 2017). CNN use time series data, sequence input data and text data.

2.3 Recurrent Neural Network

Recurrent neural network is used to control the sequence prediction problems. These problems will be exist in different forms and define different types of output and input. Example of sequence prediction problems are as follows

- One to many: input is mapped to sequence with various steps as output
- Many to many: sequence of various steps as input mapped to sequence with various steps as output.
- Many to one: sequence of various steps as input mapped to prediction

Many –to- many problems are described as sequence to sequence. Long short-term memory network is one of the most significant recurrent neural networks as it resolves training problem of recurrent network and used in different types of applications. Recurrent neural network and long-short term memory are the significant applications while working with sequence of paragraphs and words known as natural language processing. It consists of sequence of spoken language and text represented as the time series. These are used as generative model which needs sequence output and applications like creating handwriting (Zhang & Xiao 2000). RNN use speech data, text data, regression and classification prediction problems and generative models. These networks are not used for tabular datasets as we can see in spreadsheet and csv file. It is not used for image data input. RNN use image, time series and tabular data. LSTM and RNN have analysed the time series forecasting problems as the results become poor. Auto regression methods and linear methods works well. Multilayer perceptrons can perform long short-term memory on same data. We use this long short- term memory to access each input of three times steps and used dense layer to manage summary of input sequence. It uses effective adam version of stochastic gradient descent and enhance mean square error function. After defining the model, they will be used training data and make a prediction (Kong et al. 2017).

2.4 Hybrid Neural Neywork

CNN and RNN model are used together in this hybrid neural network model. These networks are used as layers which has more MLP layer. These are called hybrid type of neural network. A significant work emerges from the combination of different types of network into hybrid models. If the model prefers stack of layers with CNN, then it expects LSTM as output. This model can read image inputs like video and produce prediction. It is also known as CNN LSTM. These network types will be used in a specific architecture to enhance new capabilities like reusable image recognition and prefer CNN and MLP networks which can be added to new LSTM model. This encoder and decoder LSTM networks used to make input and output sequence of different lengths. A significant advantage of this model is that model can support long input sequences and read as blocks by CNN model. We describe a univariate problem as sequence of integers and fit into model and forecast next value in sequence. By having hybrid model, the researchers can classify the sample into different subsequence. CNN model will use sub sequence and LSTM will interpret from

subsequence. We divide each sample into subsequence of two times. CNN model is used to define and expect two times steps with single feature. CNN model is integrated in the time distributed wrapper layers as it applies to every sub sequence. LSTM interpret the results and model prefers stochastic gradient descent and enhances mean square error and it works well for the multivariate time series forecasting (Lin et al. 2017).

Model	Pros	Cons
Linear regression	Control various time series, features and components High interpretability	Effective assumptions and sensitive to outliers
ARIMA	Reliable confidence intervals and high interpretability	Need more data and effective assumptions and limitations. Difficult to automate.
Exponential smoothing	Control variable trend, level and seasonality components Automated optimization	Narrow confidence intervals Sensitive to outliers
Neural model network	Low assumptions and restrictions. Manage difficult nonlinear patterns and high predictive power Automated easily	Complex to derive confidence intervals for prediction Need more data Low interpretability

Table 1: Pros and Cons of State of Art Models

3 Importance of Data Sets in Machine Learning

The basic step in a project on machine learning is organizing data to use in training the machine how to recognize patterns. Creating a data set that is not simple since its quality determines the quality of the training. The following are the three categories of dataset.

- Validation data set. It's a subset that gives a measure of how the predictive model observes the standards of quality desired.
- Training data set. A subset that forms the predictive model.
- Test data set. A subset that governs the future performance of the predictive model. The predictive model that aligns closely with the test set is less likely to be overfitted.

4 Methodology

4.1 Data Pre-Processing

In this section, we have performed multivariate time series forecasting and analysis. We took the data set of bank nifty index from Kaggle dataset and explore it with different statistical and deep learning models. In this paper, we predicted close price of index and then compared with the test data set. Here, six different models such as LSTM (Long-Short Term Memory) which is based on RNN (Recurrent Neural Network), CNN (Convolutional Neural Network), GRU(Gated Recurrent Unit), MLP (Multi Perceptron), Random Forest and VAR (Vector Auto-regressive) are used.

This second data set was attained from UCI machine learning repository website and is about Household Electricity Consumption. Electricity usage data is growing since the adoption of smart meters and new technologies of generating electricity. This data introduces more power related variables that can be used in modelling electricity consumption and even forecasting future consumption

Here six different models are developed and then compared to identify how model works and which approach would be better. In the end, results are evaluated by the visualisation and mathematical measurement.

Acquired data is in raw form, and it cannot be used directly for training time series machine learning models. The first step is to filter features of the acquired data set and process it to remove uneven distribution of data points. Below is the figure of raw data set:

	index	date	time	open	high	low	close
0	BANKNIFTY	20121203	09:16	12125.70	12161.70	12125.70	12160.95
1	BANKNIFTY	20121203	09:17	12161.75	12164.80	12130.40	12130.40
2	BANKNIFTY	20121203	09:18	12126.85	12156.10	12126.85	12156.10
3	BANKNIFTY	20121203	09:19	12157.25	12164.75	12151.60	12164.20
4	BANKNIFTY	20121203	09:20	12162.80	12162.80	12148.20	12151.15
5	BANKNIFTY	20121203	09:21	12152.95	12159.90	12151.90	12158.15
6	BANKNIFTY	20121203	09:22	12163.35	12165.65	12146.05	12146.05
7	BANKNIFTY	20121203	09:23	12147.00	12153.65	12143.70	12152.95
8	BANKNIFTY	20121203	09:24	12151.10	12153.40	12149.70	12152.55
9	BANKNIFTY	20121203	09:25	12152.10	12152.10	12143.20	12143.20
10	BANKNIFTY	20121203	09:26	12141.05	12142.20	12136.95	12138.65
11	BANKNIFTY	20121203	09:27	12140.15	12140.15	12133.30	12136.80
12	BANKNIFTY	20121203	09:28	12135.30	12135.30	12132.90	12134.00
13	BANKNIFTY	20121203	09:29	12131.75	12154.10	12131.75	12154.10
14	BANKNIFTY	20121203	09:30	12154.15	12154.55	12149.00	12154.55
15	BANKNIFTY	20121203	09:31	12156.20	12161.50	12155.60	12161.50

Fig.3 Raw Stocks Data

The data is not sorted on the basis of time. Therefore, it is necessary to sort data in ascending order on the basis of time. The results of sorted time based data are shown below:

	open	high	low	close	date
0	11276.55	11276.55	11265.80	11268.90	20121119
1	11267.95	11267.95	11258.85	11260.05	20121119
2	11259.45	11259.45	11243.90	11243.90	20121119
3	11246.50	11246.60	11237.90	11238.90	20121119
4	11241.15	11247.20	11239.30	11245.30	20121119
5	11248.35	11250.40	11247.50	11248.75	20121119
6	11243.10	11249.85	11242.20	11249.85	20121119
7	11249.65	11249.65	11230.20	11237.70	20121119
8	11239.60	11250.35	11239.60	11250.15	20121119
9	11250.00	11263.40	11250.00	11259.15	20121119
10	11261.25	11263.25	11256.90	11263.25	20121119
11	11264.40	11267.55	11262.65	11263.45	20121119
12	11263.50	11265.85	11259.60	11265.85	20121119
13	11265.35	11279.80	11265.35	11279.80	20121119
14	11280.15	11283.80	11275.80	11282.75	20121119
15	11284.55	11301.80	11284.55	11299.45	20121119

Fig. 4 Sorted Stocks Data

4.2 Features Extraction

The data is recorded on the basis of time, and to perform time series modeling, it is necessary to have time stamp features which include year, month, day and hours. These features are extracted from the acquired data which are used to aggregate data on the basis of different time based groupings. The data set obtained after features extraction are given below:

	open	high	low	Day	Month	Year	Hour	Minute	close	date
0	11276.55	11276.55	11265.80	1	11	2012	9	16	11268.90	20121119
1	11267.95	11267.95	11258.85	1	11	2012	9	17	11260.05	20121119
2	11259.45	11259.45	11243.90	1	11	2012	9	18	11243.90	20121119
3	11246.50	11246.60	11237.90	1	11	2012	9	19	11238.90	20121119
4	11241.15	11247.20	11239.30	1	11	2012	9	20	11245.30	20121119
5	11248.35	11250.40	11247.50	1	11	2012	9	21	11248.75	20121119
6	11243.10	11249.85	11242.20	1	11	2012	9	22	11249.85	20121119
7	11249.65	11249.65	11230.20	1	11	2012	9	23	11237.70	20121119
8	11239.60	11250.35	11239.60	1	11	2012	9	24	11250.15	20121119
9	11250.00	11263.40	11250.00	1	11	2012	9	25	11259.15	20121119
10	11261.25	11263.25	11256.90	1	11	2012	9	26	11263.25	20121119
11	11264.40	11267.55	11262.65	1	11	2012	9	27	11263.45	20121119
12	11263.50	11265.85	11259.60	1	11	2012	9	28	11265.85	20121119
13	11265.35	11279.80	11265.35	1	11	2012	9	29	11279.80	20121119
14	11280.15	11283.80	11275.80	1	11	2012	9	30	11282.75	20121119
15	11284.55	11301.80	11284.55	1	11	2012	9	31	11299.45	20121119

Fig. 5 Time Features Based Stocks Data

The distribution of data points on the basis of time features is shown below:

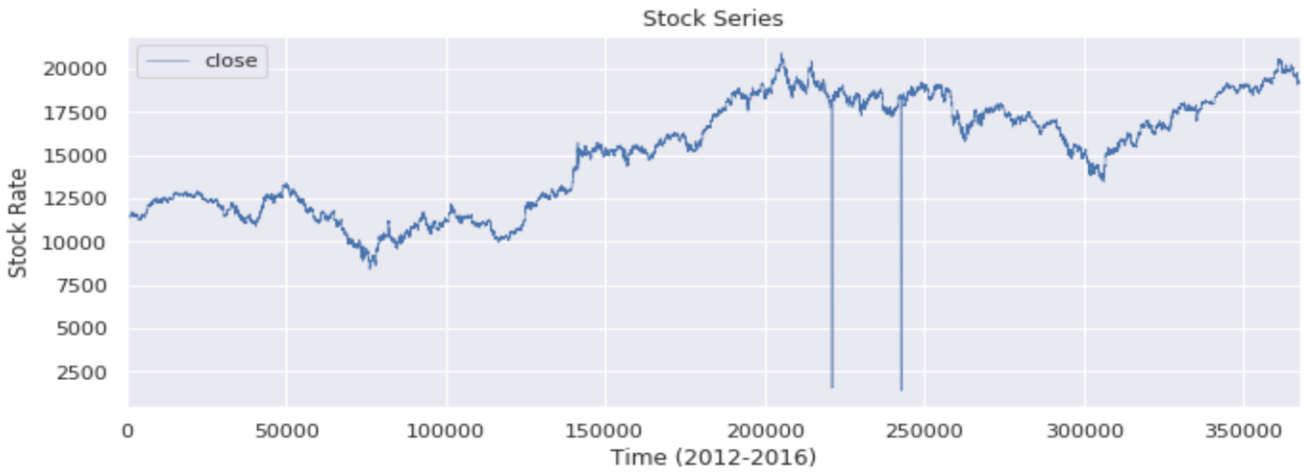


Fig. 6 Stocks rate from 2012 to 2016 of the recorded dataset.

Stock Data Aggregation by Hours:

Aggregation by hour is done by extracting hours stock values for each day. There are 7 working hours in given data and each week has 2 holidays. It makes approx. 22 days in a month. We have 5 years of data and data for first ten months (of 2012) is missing. Years = 5

Total samples for 5 years = 5*12*22*7 = 9240 samples of hours

Hence total hour aggregate should be 9240, but acquired samples are approx. 6200. This is because some data records are missing in the given data.

	open	high	low	Day	Month	Year	Hour	close
0	11276.55	11320.65	11230.20	1	11	2012	9	11301.85
1	11303.30	11305.85	11255.25	1	11	2012	10	11267.00
2	11266.50	11344.90	11256.10	1	11	2012	11	11328.50
3	11324.40	11344.55	11317.35	1	11	2012	12	11327.45
4	11332.80	11349.40	11303.20	1	11	2012	13	11325.15
5	11324.70	11333.45	11298.80	1	11	2012	14	11324.25
6	11324.15	11326.20	11299.10	1	11	2012	15	11306.25
7	11442.85	11446.05	11398.35	2	11	2012	9	11436.70
8	11438.20	11440.05	11406.40	2	11	2012	10	11422.75
9	11421.95	11429.25	11394.15	2	11	2012	11	11426.65

Fig. 7 Hour Grouped Stocks Data

Stock Data Aggregation by Days:

Same strategy is used as done for says as of hours. The only difference is, instead of hours, day unique values are extracted for each month of each year.

Total Days for 5 years = 5*12*22 = 1320 samples of days

The total days' samples should be 1320 but extracted day aggregate is 933 samples due to missing values.

	open	high	low	Day	Month	Year	close
0	11276.55	11349.40	11230.20	1	11	2012	11306.25
1	11442.85	11477.55	11394.15	2	11	2012	11465.05
2	11442.20	11489.30	11401.30	5	11	2012	11475.20
3	11439.45	11568.45	11429.05	6	11	2012	11560.05
4	11548.50	11719.10	11519.60	7	11	2012	11658.70
5	11543.70	11637.95	11533.30	8	11	2012	11618.90
6	11611.10	11663.90	11427.20	9	11	2012	11485.80
7	11501.95	11608.80	11462.30	12	11	2012	11608.80
8	11476.85	11530.25	11419.05	15	11	2012	11523.65
9	11495.10	11537.15	11306.05	16	11	2012	11328.05
10	11346.95	11378.25	11256.40	19	11	2012	11316.85

Fig. 8 Day Grouped Stocks Data

Stock Data Aggregation by Months:

For month stock values are extracted of each year.

Total Months for 5 years = 5*12 = 60 samples of days

The total month samples should be 60 but extracted month aggregate is 47 samples due to missing values.

	open	high	low	Month	Year	close
0	11276.55	12175.30	11230.20	11	2012	12153.10
1	12125.70	12571.60	12053.45	12	2012	12474.25
2	12547.15	12960.65	12509.45	1	2013	12702.75
3	12719.85	12743.20	11401.20	2	2013	11563.95
4	11544.70	12241.30	11048.40	3	2013	11401.70
5	11412.80	12755.80	10888.75	4	2013	12551.95
6	12552.85	13414.30	12188.30	5	2013	12458.55
7	12467.50	12520.55	11035.90	6	2013	11599.65
8	11607.25	11811.90	9863.05	7	2013	9982.05
9	10122.65	10219.60	8366.75	8	2013	9066.95

Fig. 9 Month Grouped Stocks Data

Household Electricity Consumption Data:

For household data, preprocessing of raw data is also carried out in the same way as we did for stocks data. But in household data, data records are missing which needs to be removed. The first step before extracting time features is to remove all those samples of household power consumption records having any missing value for any feature in the sample. After removing those household records, time features are extracted and unnecessary features are dropped. The data set before and after features extraction is shown below:

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3
0	16/12/2006	17:24:00	4.216	0.418	234.840	18.400	0.000	1.000	17.0
1	16/12/2006	17:25:00	5.360	0.436	233.630	23.000	0.000	1.000	16.0
2	16/12/2006	17:26:00	5.374	0.498	233.290	23.000	0.000	2.000	17.0
3	16/12/2006	17:27:00	5.388	0.502	233.740	23.000	0.000	1.000	17.0
4	16/12/2006	17:28:00	3.666	0.528	235.680	15.800	0.000	1.000	17.0

Fig.10 Household Raw Data Set

	Voltage	Global_intensity	Day	Month	Year	Hour	Minute	Global_active_power
0	234.84	18.4	5	12	2006	17	24	4.216
1	233.63	23.0	5	12	2006	17	25	5.360
2	233.29	23.0	5	12	2006	17	26	5.374
3	233.74	23.0	5	12	2006	17	27	5.388
4	235.68	15.8	5	12	2006	17	28	3.666

Fig. 11 Household Power Consumption after Feature Extraction

The distribution of data point on the basis of time records is shown below for voltage and global intensity and global active power consumption.

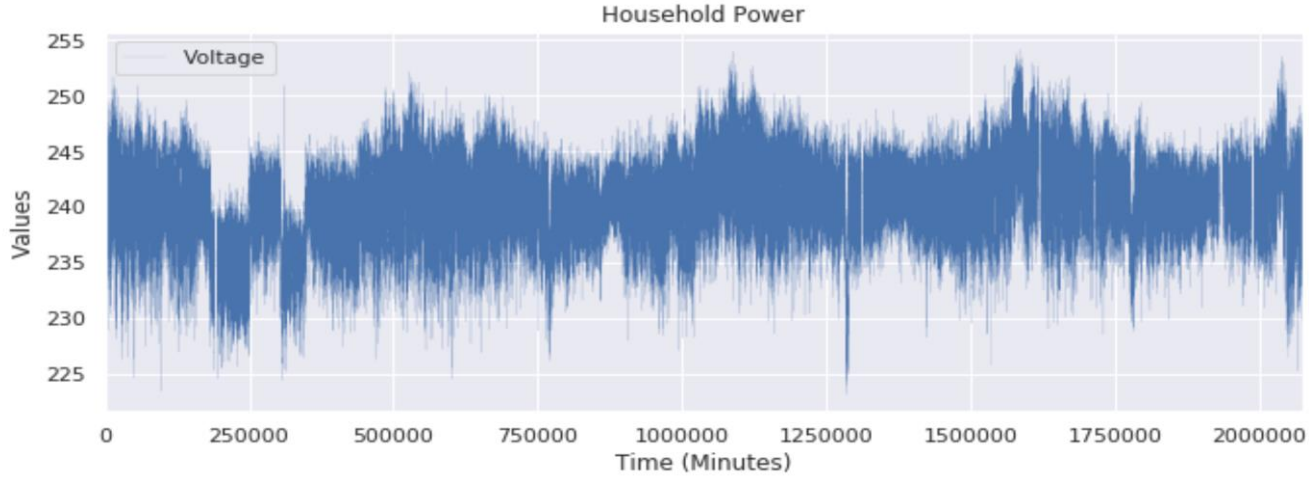


Fig.12 Voltage values Time Based Distribution

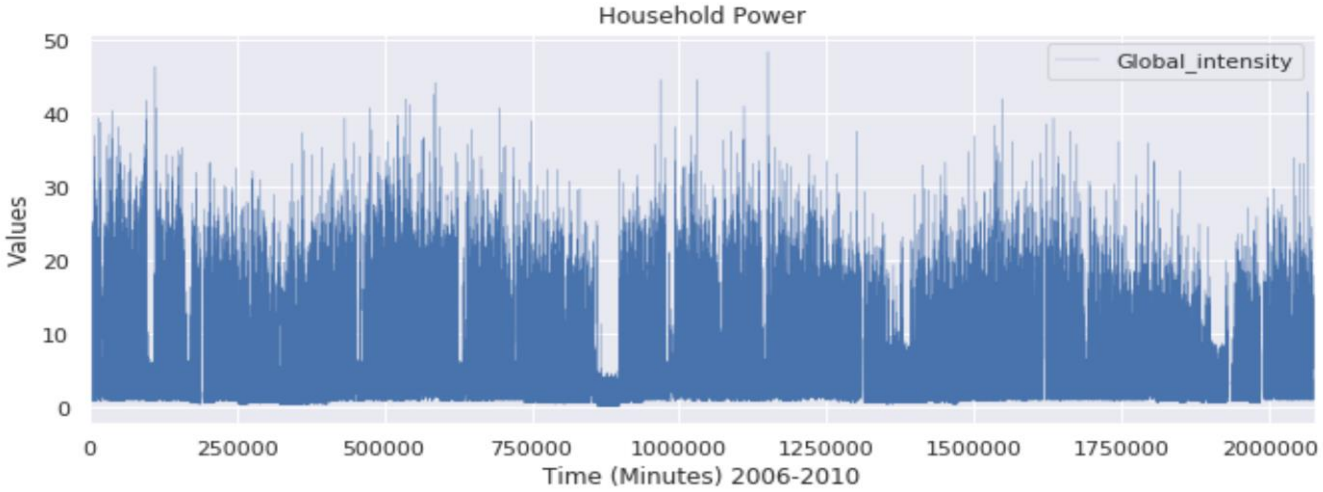


Fig.13 Global Intensity values Time Based Distribution

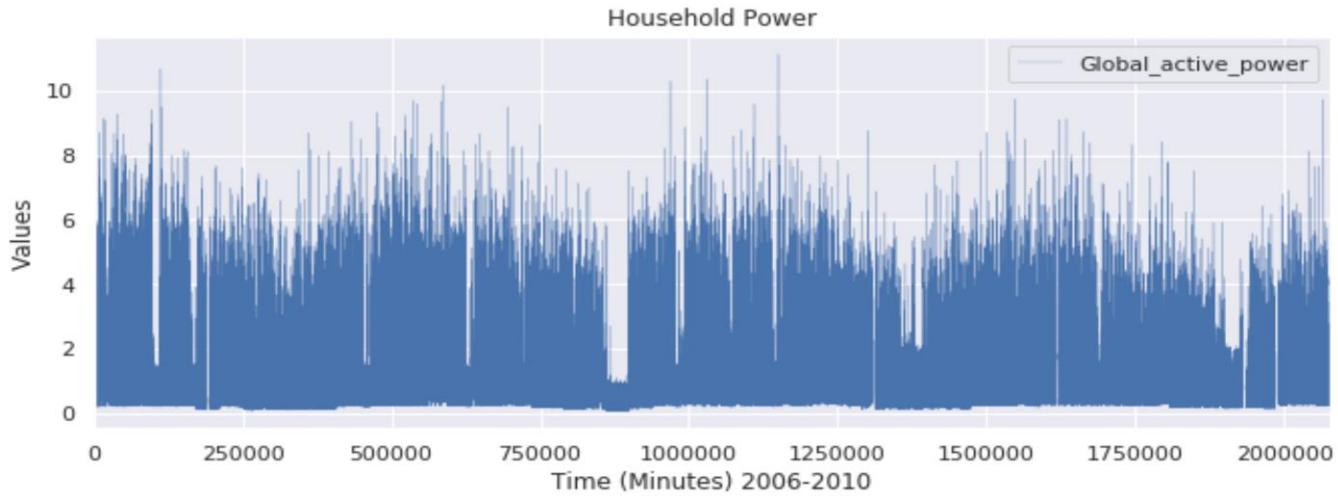


Fig.14 Global Active Power values Time Based Distribution

For time series there are some components in which a time series is broken down. The purpose of this is to observe the consistency and recurrence of values in time series data. These components include trend, seasonality, noise and stationarity. Trend shows the increasing or decreasing values in the time series data. Seasonality is the repeating short term cycle in the time series and noise is the random variation in the series. Stationary time series has constant statistical parameters such as mean, variance, auto correlation. Most statistical forecasting methods are based on the assumption that the series can be rendered approximately stationary. Time series having no trend or seasonality is stationary having cycles with no fixed length. A stationary time series have no predictable patterns in long term.

The trend and seasonality of stocks data is shown below:

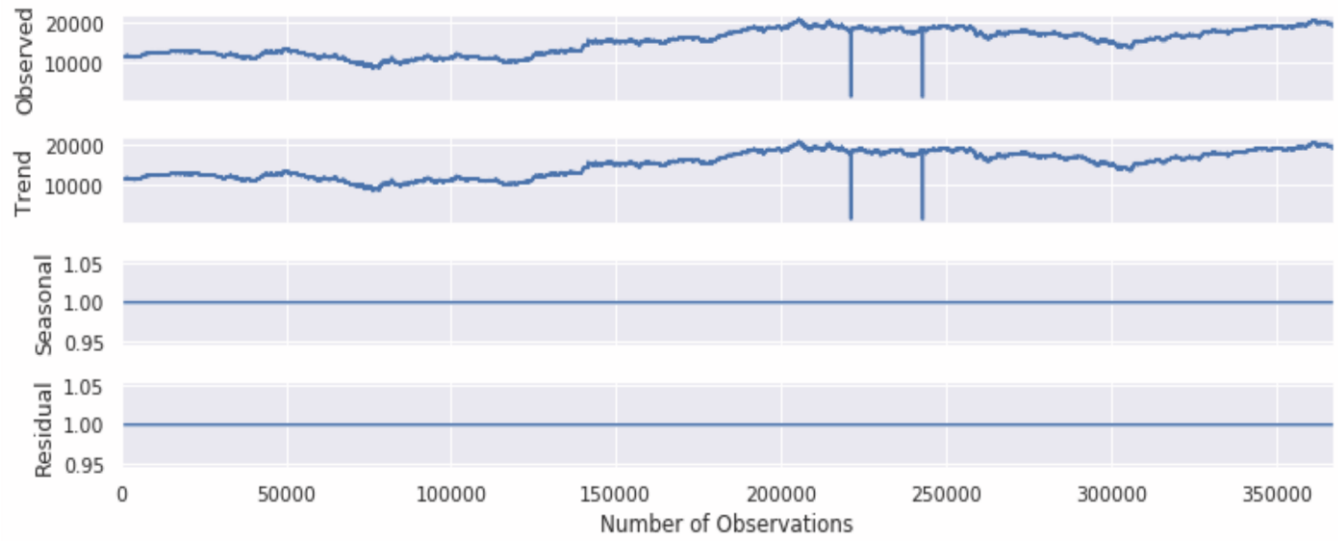


Fig.15 Trend and seasonality of stocks data

The trend and seasonality of household power consumption is shown below:

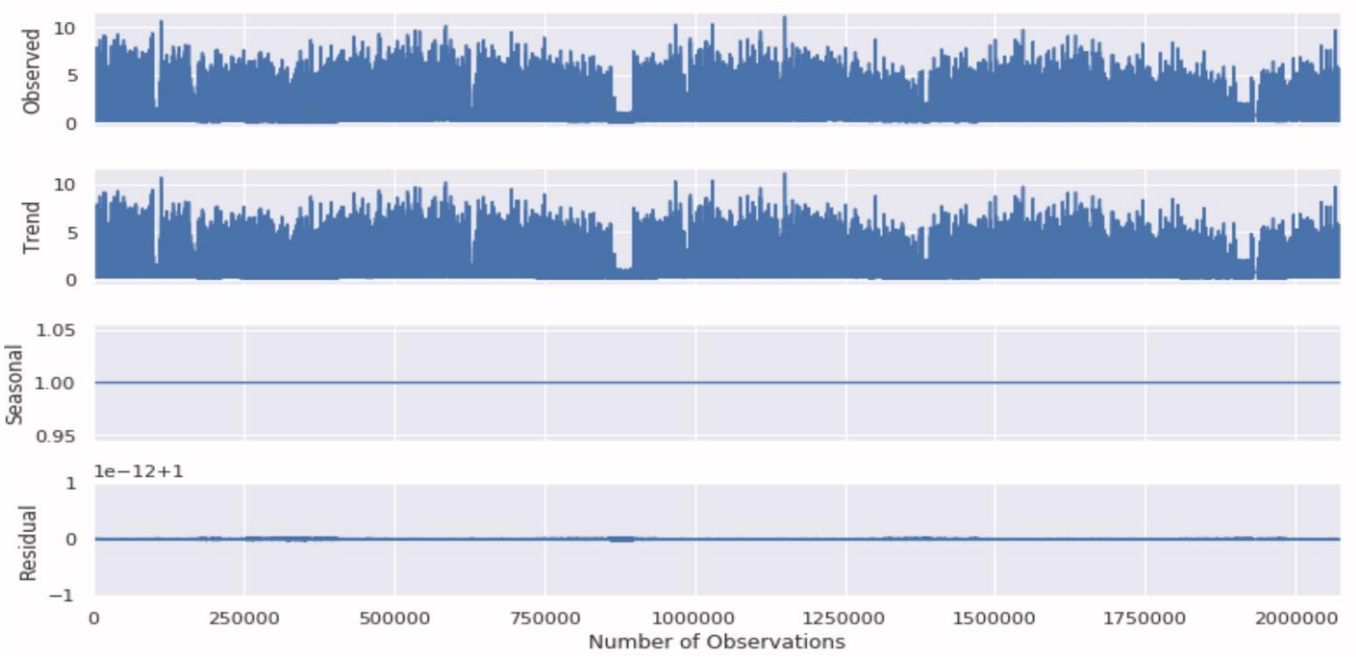


Fig. 16 Trend and seasonality of Household data

4.3 Data Scaling and Spitting

Examples in a training dataset provides inputs and outputs for deep learning neural networks to map as they learn. Estimates of error are then responded to by initializing the weight of the model to insignificant arbitrary values and an optimization algorithm is used to update them. It is also very crucial to scale inputs and outputs used in training the model. This is because of use of the error between the right values and the predicted values and use of small weights. If the inputs are not scaled, the learning process can be slow and unstable and not scaling the output makes the gradients to explode resulting in failure of the process. Standardization and normalization methods are used in data preparation before training a neural network model to rescale inputs and outputs (Kotsiantis et al. 2006).

In this proposed research, we have scaled the data for all deep learning models such as LSTM (long short term memory), CNN (convolutional neural network), MLP (multi perceptron) and GRU (gated recurrent unit) whereas for Random Forest and VAR (vector auto-regressive) data has been used without normalisation.

For LSTM, CNN, MLP, and GRU data is scaled between 0 and 1. The resultant scaled data is then split into train and test set by taking 20 percent of scaled data for test set randomly and shuffling train test data points.

4.4 Model Building

3.4.1 Recurrent Neural Neyworks (RNN)

Recurrent neural networks have cycles that input previous time step's activations to stimulate the present time step's predictions. The networks have long-term internal states that can hold contextual information, used to store the activations. Through this the RNN is able to exploit a relative window that changes animatedly over the input sequence history. A new dimension is introduced to the function being approximated by addition of sequence so that the network not only maps inputs and outputs but also learns a function for mapping inputs over time to the output. The output is not only dependent on the inputs but also the recent context in input sequence due to internal sequence. Time series tasks that cannot be solved by use of fixed time windows can be unravelled by long short-term memory (LSTM). RNN can also learn and extract temporal dependence from data. That is the network can observe a sequence of inputs and learn the relevance of a previous observation on making a prediction. LSTM networks can model complex multivariate sequences accurately due to their capability to learn Long short-term correlations when provided with a sequence. The recurrent neural networks make it possible to learn contextual information and temporal dependencies from the input data. This depends on whether the inputs are in an input sequence or are fixed. If they are in an input sequence, they can be transformed into an output sequence in a flexible way while considering contextual information (Zaremba et al. 2014).

3.4.2 Long Short Term Memory (LSTM)

The long-short term memory comprises of layers of neurons just like an MLP, but in some way it is different to the classical MLP. For it to perform a prediction, the input data has to be proliferated across the network. The LSTM formulates the outputs in the same way as the RNN, where neuron activations from previous time step are fed into the current time step. The LSTM is however different from other RNN in that its unique formulation avoids

problems that make the other RNN not be able to scale and train. This technique is popular due to this difference and its remarkable outputs.

Here LSTM model has been applied and LSTM layer has added in neural network model for this purpose.

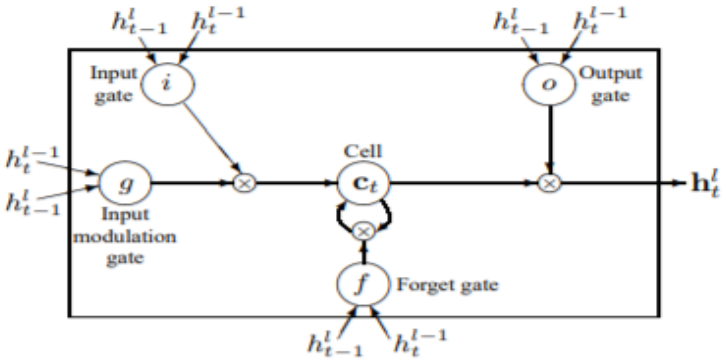


Fig. 17 LSTM Architecture

Several neural networks are connected together by data so that the neural network can accept a time series. Each of the connected neural networks, is responsible for a single time step. The data is fed to the entire window or a context of the neural network instead of a single time step. A recurrent network requires data that has periodic patterns and a record of the historic data is required but it is impossible due to exploding and vanishing gradient. The LSTM introduces a memory unit called the cell to solve this issue. It comprises of layers that look for distinct features in the data and use them in building higher-order features. This is opposed to the hand engineered systems and corresponds more deep learning systems which automatically learns (Zaremba et al. 2014).

LSTM

In [26]:
```
from keras.layers import LSTM
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, BatchNormalization, Activation
```

```
Using TensorFlow backend.
```

In [27]:
```
lstm = Sequential()
lstm.add(LSTM(20, input_shape=(X_train_df1.shape[1], X_train_df1.shape[2])))
lstm.add(Dense(2, activation='sigmoid'))
lstm.add(Dense(y_train_df1.reshape(-1,1).shape[1]))

lstm.compile(loss='mae', optimizer='adam', metrics=['mean_squared_error'])
```

```
WARNING:tensorflow:From C:\Users\lenovo\Anaconda3\lib\site-packages\tensorflow\python\framework\op_def_library.py:263: colocate
_with (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
```

In [28]:
```
lstm.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 20)                1920
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 42
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 3
=================================================================
Total params: 1,965
Trainable params: 1,965
Non-trainable params: 0
_________________________________________________________________
```

In [29]:
```
lstm.fit(X_train_df1, y_train_df1, epochs =20, verbose=1, batch_size=8,
          validation_data=(X_test_df1,y_test_df1), shuffle=True)
```

```
WARNING:tensorflow:From C:\Users\lenovo\Anaconda3\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from te
nsorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 746 samples, validate on 187 samples
Epoch 1/20
746/746 [==============================] - 2s 3ms/step - loss: 0.3348 - mean_squared_error: 0.1658 - val_loss: 0.2171 - val_m
ean_squared_error: 0.0624
Epoch 2/20
```

Fig. 18 LSTM Model Building

Different packages are used for developing deep learning models. Tensorflow developed by google, pytorch developed by facebook and theanos are very common. Keras is a high end API which uses tensorflow or theano as backend. For this research work Keras is used with tensorflow as backend. There are two ways to build Keras models. One is sequential and the other is functional. In sequential method, layers are added one after the other in a sequence. In functional method layers are flexible and can be interconnected to each other and multiple input layers can be included in the model, which is not applicable in sequential method. Functional method is used when the model is very complex, but in our case sequential method is used. Modules from Keras package are imported to develop model. The first layer is input layer with number of nodes equal to number of features of the data set. Then LSTM layer is added with number of nodes equal to 20. The input layer parameters and LSTM layer are merged in a single layer. Dense_1 is the hidden layer with 2 nodes. The last layer Dense_2 is the output layer having single node. The number of nodes and hidden layers are hyperparameter which are selected based on performance of the model on the given data set. The model is then compiled with evaluation metrics and optimization function to measure and improve the performance of the model. Mean absolute error and

mean square errors are selected to evaluate the performance, and adaptive momentum estimation is used to improve the performance of the model by optimizing the learning rate for each node. After that the model is trained over training set to obtain the final model. The trained set is used to train the LSTM model by running multiple iterations on the model. The test set is used to evaluate this model, to see how it will perform for unseen data.

Adaptive Moment Estimation (Adam):

Adam is an adaptive learning rate optimization algorithm to optimize learning rate to improve weights of all layer. In Adam learning rate is maintained for each weight of network, and separately adapted as learning fold. The formulation of Adam is given below.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

In above equations m and v are moving averages, g is gradient and beta is decay rate. After this second moving averages are computed.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$
$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t.$$

The second moving average results are fed in above equation to compute the final results. (Kingma & Ba 2014)

Activation Functions:

There are different activation functions which are chosen according to the output values. Sigmoid function is used where output is binary (0 or 1), and for regression where output values are between 0 and 1. Tangent hyperbolic function is used where output is between -1 and 1.

Sigmoid function is given by:

$$g(x) = a = 1 / 1 + e^{-z}$$

Tangent hyperbolic function is given by:

$$a = \tanh(z)$$

Rectified Linear Unit:

Rectified linear unit (relu) generates very good experimental results. The ReLU function is 0 for negative values, but grows linearly for positive values. It is given by equation:

$$a = \max(0, x)$$

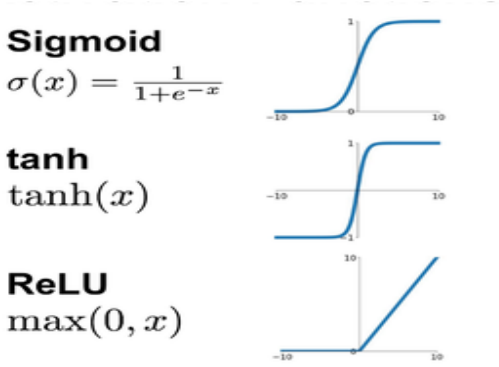


Fig.19 Graphical Representation of Activation Functions

(Nwankpa et al. 2018)

After training the model is ready for prediction. Here one thing is necessary, that input data is scaled, so the predicted value from the model is scaled value which needs to be inverse scaling to get actual output value. In figure 8 it is shown that predicted value is 0.64. Inverse scaling script is added to reverse this predicted value scaling which is 16236 and the true value is 16723.

```
# Output value is scaled. To get actual value undo scaled value of output
print('Scaled Value Predicted: %.2f' %predict[2])
print('Actual Predicted Value: %.2f'%out_scaler.inverse_transform([predict[2]]))
print('True Value: %.2f' %out_scaler.inverse_transform([[y_test_df1[2]]]))
```

```
Scaled Value Predicted: 0.64
Actual Predicted Value: 16236.89
True Value: 16723.40
```

Fig.20 Descaling Predicted Value

3.4.3 Convolutional Neural Network (CNN)

CNN model is mainly for RGB images but it can also be used for 1 dimensional data which is stock and household electricity consumption data sets. In convolution neural networks (CNN) the given data goes through different layers to get output. The input data is first convolved by applying convolution filter in the convolution layer. The output of convolution layer is passed to activation function layer to produce non linearity. Then the results are passed to pooling layers. These layers find a number of features in the data and progressively construct higher-order features. This corresponds directly to the ongoing theme in deep learning by which features are automatically learned as opposed to traditionally hand engineered. After this flattening layer is used to convert the results from previous layer from matrix to vector, which is required so output layer predicts class for these results. Before passing flattening layer results to output layer, activation function layer is applied, commonly softmax for categorical classification. (Patterson & Gibson 2017)

This is the general architecture of CNN which is also shown below:

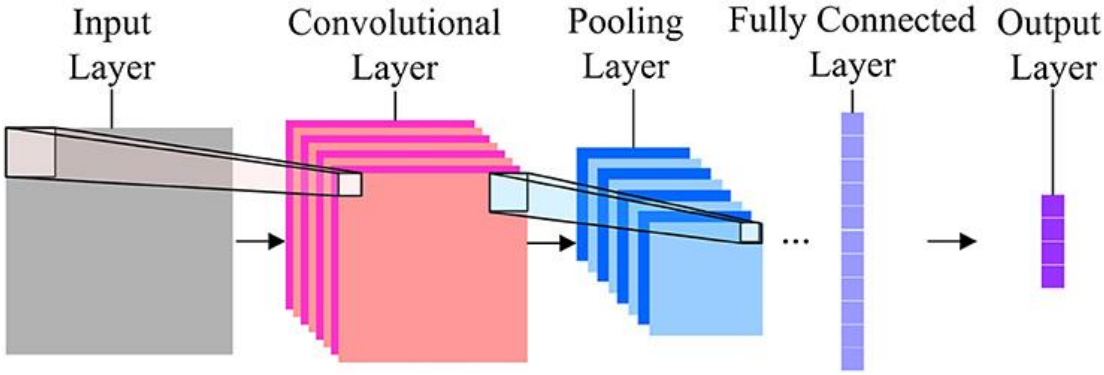


Fig.21 Convolution Neural Network Architecture

Pooling Layers:

Pooling layer is used to reduce the dimensionality of your feature 'tensor' in CNN. Max pooling layer returns the feature value which is the highest among them, this means that it returns the pixel values that cause edges (Patterson & Gibson 2017).

Average Pooling:

In average pooling, average value is selected from filter window values, and new set of features is obtained from this. (Patterson & Gibson 2017)

Max Pooling:

In max pooling, maximum value is selected from filter window values, and new set of features is obtained from this. Average and max pooling are shown below. (Rawat & Wang 2017)

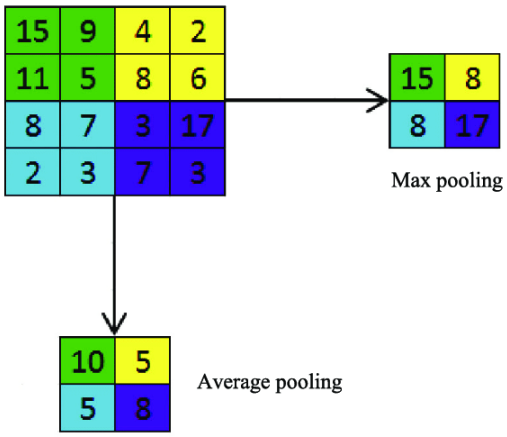


Fig.22 Average and Max Pooling

Code for CNN is given below:

Convolution Neural Network

```
In [27]: import keras
         from keras.models import Sequential
         from keras.layers import Dense, Dropout, BatchNormalization, Activation, Conv1D, Flatten, MaxPooling1D

Using TensorFlow backend.
```

```
In [28]: model = Sequential()
         model.add(Conv1D(filters=8, kernel_size=3, activation='relu', padding = 'same',
                          input_shape=(X_train_df1.shape[1],X_train_df1.shape[2])))
         model.add(MaxPooling1D(pool_size=2, padding='same'))
         model.add(Flatten())
         model.add(Dense(4))
         model.add(Activation('relu'))
         #model.add(Dense(4))
         #model.add(Activation('relu'))
         model.add(Dense(y_train_df1.reshape(-1,1).shape[1]))

         model.compile(loss='mae', optimizer='adam', metrics=['mean_squared_error'])

WARNING:tensorflow:From C:\Users\lenovo\Anaconda3\lib\site-packages\tensorflow\python\framework\op_def_library.py:263: colocate
_with (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
```

```
In [29]: model.summary()

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_1 (Conv1D)            (None, 1, 8)              152
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 1, 8)              0
_________________________________________________________________
flatten_1 (Flatten)          (None, 8)                 0
_________________________________________________________________
dense_1 (Dense)              (None, 4)                 36
_________________________________________________________________
activation_1 (Activation)    (None, 4)                 0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 5
=================================================================
Total params: 193
Trainable params: 193
Non-trainable params: 0
_________________________________________________________________
```

```
In [30]: model.fit(X_train_df1, y_train_df1, epochs =18, verbose=1, batch_size=8,
                   validation_data=(X_test_df1,y_test_df1), shuffle=True)

WARNING:tensorflow:From C:\Users\lenovo\Anaconda3\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from te
nsorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 746 samples, validate on 187 samples
Epoch 1/18
746/746 [==============================] - 2s 2ms/step - loss: 0.4367 - mean_squared_error: 0.2559 - val_loss: 0.2382 - val_m
ean_squared_error: 0.0889
Epoch 2/18
746/746 [==============================] - 0s 263us/step - loss: 0.1532 - mean_squared_error: 0.0352 - val_loss: 0.1261 - val
_mean_squared_error: 0.0224
Epoch 3/18
746/746 [==============================] - 0s 284us/step - loss: 0.1105 - mean_squared_error: 0.0185 - val_loss: 0.1002 - val
_mean_squared_error: 0.0144
Epoch 4/18
746/746 [==============================] - 0s 246us/step - loss: 0.0844 - mean_squared_error: 0.0116 - val_loss: 0.0739 - val
 mean squared error: 0.0091
```

Fig. 23 CNN Model Building

Convolution neural networks (CNN) are used for complex data such as images. Its architecture is for one dimensional data, two dimensional data and three dimensional data. Three dimensional data is the tensor data set which are of rgb images. One dimensional and two dimensional architectures are used for vectors and matrix form data sets. The convolution architecture used in this research is one dimensional. Keras API is used to develop CNN model and the method selected to develop CNN model is sequential. The input layer parameters are merged in convolution layer. The number of filters selected are 8 and the size of these filters selected is 3x3. After convolution process the size of input data matrix is changed or convolved, to keep it constant padding is added to keep the size of output same as input. After convolution layer pooling layer is added, and the pooling layer

selected is max pooling. The size of pooling window is 3x3. The matrix obtained from max pooling operation is converted into vector by adding flattening layer. Dense_1 is the hidden layer with four nodes and Dense_2 is the output layer. The activation function selected for this model is rectified linear unit. The model is then compiled with Adaptive Momentum Estimation (adam) optimizer and mean absolute error and mean square error for loss evaluation.

In our model, the input data is fed in 1 dimensional convolution layer which is added with 3 nodes, then max pooling layer is added. The 2d result of pooling layer is transformed to vector by adding flattening layer. Then a hidden layer with 64 nodes is added followed by activation layer. After that another hidden layer (Dense_5) is added with 4 nodes followed by activation layer. Dense_6 is the output layer.

The activation function used between layers is RELU. The filter size is 8x8 and padding is selected 'same', which means the size of input matrix is not recovered by adding zero padding after convolution process. The nodes in the convolution layers are 3. The optimization function for learning rate is ADAM, and loss functions are mean absolute error and mean square error.

3.4.4 Random Forest

The Random Forest is a technique that uses Bootstrap Aggregation and multiple decision trees to perform regression and classification tasks. The Bootstrap Aggregation involves using a different sample of data on each decision tree and replacement is used to sample. Instead of relying on a single decision tree, the output is derived from many decision trees. A neural network uses the concept and functionality of animal neuron with interconnected nodes and simple processing elements. The network's processing power is stored in the weights, inter-unit connection strengths, which are attained by adaptation to a set of training patterns.

This is an ensemble learning strategy that produces numerous regression trees (CART) and totals their outcomes. The model works on examples of the time series seasonal cycles which rearranges the determining issue particularly when a time series shows heteroscedasticity, nonstationarity, multiple seasonal cycles and trend. The primary benefits of the model are its capacity to speculation, worked in cross-validation and low affectability

to parameter values. (Dudek 2015). The below image has shown how the Random Forest model is developed.

Random Forest

```
In [18]: from sklearn.ensemble import RandomForestRegressor

In [19]: rf = RandomForestRegressor(verbose=1)

In [20]: rf.fit(x_train_df1, y_train_df1)

/home/kirmani/anaconda3/envs/env1/lib/python3.6/site-packages/sklearn/ensemble/forest.py:245: FutureWarning: The default value
of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done  10 out of  10 | elapsed:    0.0s finished

Out[20]: RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
                               max_features='auto', max_leaf_nodes=None,
                               min_impurity_decrease=0.0, min_impurity_split=None,
                               min_samples_leaf=1, min_samples_split=2,
                               min_weight_fraction_leaf=0.0, n_estimators=10,
                               n_jobs=None, oob_score=False, random_state=None,
                               verbose=1, warm_start=False)
```

Fig. 24 Random Forest Model Building

To develop random forest model, scikit learn package is used. Our data set is of regression type, therefore, random forest regression general architecture is imported from the package. Then the training set obtained from validation step is used to train the random forest model. The trained model is then used to predict the test set output. The predicted output is evaluated by measuring error between true value and predicted value. If the error is small the model is finalized, otherwise, the parameters of the model or the data set are tuned. The evaluation metrics used are mean square error, mean absolute error, and root mean square error. The train data from processed data explained above is fitted in random forest regression algorithm to build random forest model. RF Model is then evaluated by evaluating it on test set.

3.4.5 Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU) was introduced by Cho, et al. In 2014. The purpose of GRU is to solve vanishing gradient problem and exploding gradient problem in recurrent neural network. It was developed as an improvement to LSTM architecture. It is observed that its performance is mostly equal or better than LSTM. It has fewer parameters and operations as compared to LSTM. Like classic recurrent neural network it has a single hidden state, which is the grouping of hidden and cell states as in LSTM. In GRU two gates are present for cell. The first one is update gate z, in which input and LSTM gates are combined. The update gate makes decision based on the input data to discard unnecessary information, and add relative

information in its place. When new information is added, both input and forget gates combination ensures that cell will forget the previous information. (Wang et al. 2018) Mathematically, update gate can be represented as:

$z_t = \sigma (W_z x_t + U_z h_{t-1})$

The second gate is called reset gate, denoted by r_t. The reset gate uses the previous cell state h_{t-1} and the input given to decide how much of the previous state to pass through:

$r_t = \tanh (Wx_t + U (r_t * h_{t-1}))$

The next step is candidate state, h`t:

$h`_t = \tanh (Wx_t + U (r_t * h_{t-1}))$

The output of GRU is denoted by h_t, which is obtained at time t due to linear intepolation between previous output h_{t-1} and the candidate output $h`_t$:

$h_t = (1 – z_t) * h_{t-1} + z_t * h`_t$

The GRU cell is shown as:

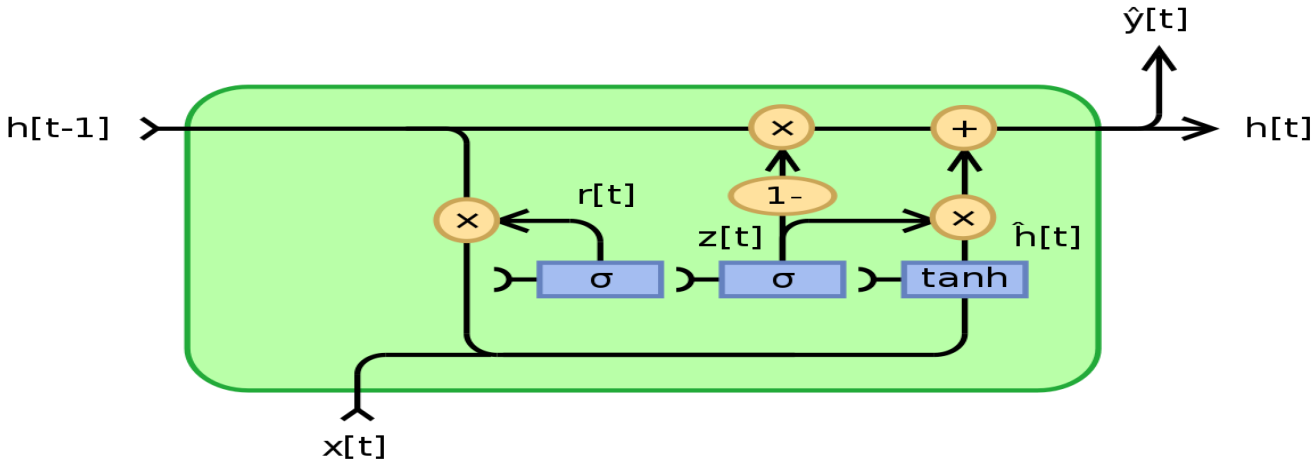


Fig. 25 GRU Architecture

(Wang et al. 2018)

The below image is the snippet of GRU modelling.

Gated Recurrent Unit

```
In [20]: from keras.layers import LSTM, GRU
         import keras
         from keras.models import Sequential
         from keras.layers import Dense, Dropout, BatchNormalization, Activation

         Using TensorFlow backend.

In [21]: gru = Sequential()
         gru.add(GRU(20, input_shape=(X_train_df1.shape[1], X_train_df1.shape[2])))
         gru.add(Dense(2, activation='sigmoid'))
         gru.add(Dense(y_train_df1.reshape(-1,1).shape[1]))

         gru.compile(loss='mae', optimizer='adam', metrics=['mean_squared_error'])

In [22]: gru.summary()

         _________________________________________________________________
         Layer (type)                 Output Shape              Param #
         =================================================================
         gru_1 (GRU)                  (None, 20)                1620
         _________________________________________________________________
         dense_1 (Dense)              (None, 2)                 42
         _________________________________________________________________
         dense_2 (Dense)              (None, 1)                 3
         =================================================================
         Total params: 1,665
         Trainable params: 1,665
         Non-trainable params: 0
         _________________________________________________________________

In [23]: gru.fit(X_train_df1, y_train_df1, epochs =18, verbose=1, batch_size=8,
                  validation_data=(X_test_df1,y_test_df1), shuffle=True)

         Train on 746 samples, validate on 187 samples
         Epoch 1/18
         746/746 [==============================] - 4s 5ms/step - loss: 0.8671 - mean_squared_error: 0.8368 - val_loss: 0.5874 - val_m
         ean_squared_error: 0.3860
         Epoch 2/18
         746/746 [==============================] - 1s 1ms/step - loss: 0.3628 - mean_squared_error: 0.1805 - val_loss: 0.2413 - val_m
         ean_squared_error: 0.0817
         Epoch 3/18
         746/746 [==============================] - 1s 1ms/step - loss: 0.2135 - mean_squared_error: 0.0624 - val_loss: 0.1965 - val_m
         ean_squared_error: 0.0510
         Epoch 4/18
         746/746 [==============================] - 1s 1ms/step - loss: 0.1881 - mean_squared_error: 0.0460 - val_loss: 0.1697 - val_m
         ean_squared_error: 0.0387
         Epoch 5/18
```

Fig. 26 GRU Model Building

For developing GRU deep learning model, keras API has been used. Sequential method is used for developing this model. The input layer parameters are merged in to gated recurrent unit layer (gru_1). Number of units in GRU layers are twenty. After that hidden layer is added with two nodes, denoted by Dense_1. The last layer Dense_2 is output layer having single node. The model is then compiled with evaluation metrics to measure loss between true and predicted values. Adaptive momentum estimation is used to optimize learning rates of the model. Number of epochs is the number of iterations a model goes through to reduce loss error and optimize weights at each node by improving learning rate. Again the values of these parameters are selected by observing the train and test set results, and are modified accordingly. If the performance of model is not improving, then more layers are added and the model becomes deeper and deeper. In our case the model is simple due to few layers and the efficiency achieved is maximum.

Gated recurrent unit model is developed by adding GRU layer in neural network model. The GRU layer is inserted after input layer and input is fed to this GRU layer. The evaluation metric includes mean square error to measure error between predicted and true value.

3.4.6 Multi-Layer Perceptron (MLP)

Multi-layer perceptron is basically artificial neural network with multiple layers in its network. Artificial neural network (ANN) is inspired by human brain system. A neural network is an interconnected assembly of straightforward process parts, units or nodes, whose practicality relies on the animal vegetative neuron. The processing ability of the network is stored in the inter-unit connection strengths, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns. (Patterson & Gibson 2017)

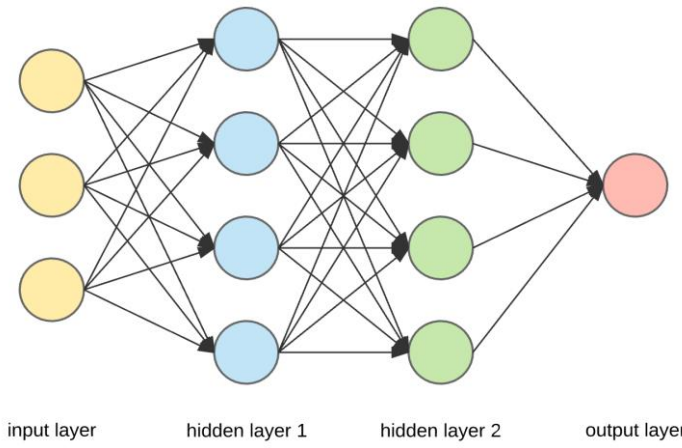


Fig.27 Artificial Neural Network

ANN connects an input layer with an output layer. The vectorial function can be written as:

$$y = f(x)$$

$$f(x) = z = \sum wx + b$$

where : w = weights, x = feauture/input, b = bias

A node is connected with input channels, each of them characterized by a synaptic weight w. The input is split into its components and they are multiplied by the corresponding weight and summed. Bias is added to the sum. The resulting sum is filtered by an activation function, and an output is produced.

The complete process in which input is fed to hidden layer and after applying activation function output is generated is called forward propagation. The predicted output result is compared to true output value and error is calculated. Error is propagated backwards to the layers and weights are updated of each node to minimize the error. Multi-layer neural network is also called multi-layer perceptron. Multi-layer perceptron becomes deep neural network if number of layers increases or network becomes deeper. The thumb rule used is when number of layers are more than 4, then it will be termed as deep neural network. Our multi-layer perceptron has 6 input features. 4 layers are added in the network with first layer called input layer having nodes equal to number of input features, second and third layers have 4 nodes and last layer is output layer having only 1 node. Number of nodes, hidden layers, activation functions and optimizers are hyper parameters which are selected according to optimum output results of the network by hit and trial method. (Bell 2014)

To calculate total parameters we use the equation:

$$y = \sum wx + b$$

$$y_l = (6*4 + 4*4 + 4*1) + (4+4+4+1)$$

$$y_l = 53$$

In one iteration feed forward is applied and then backpropagation is applied to reduce error by updating the weights. The model flow chart of multi-layer perceptron is shown as:

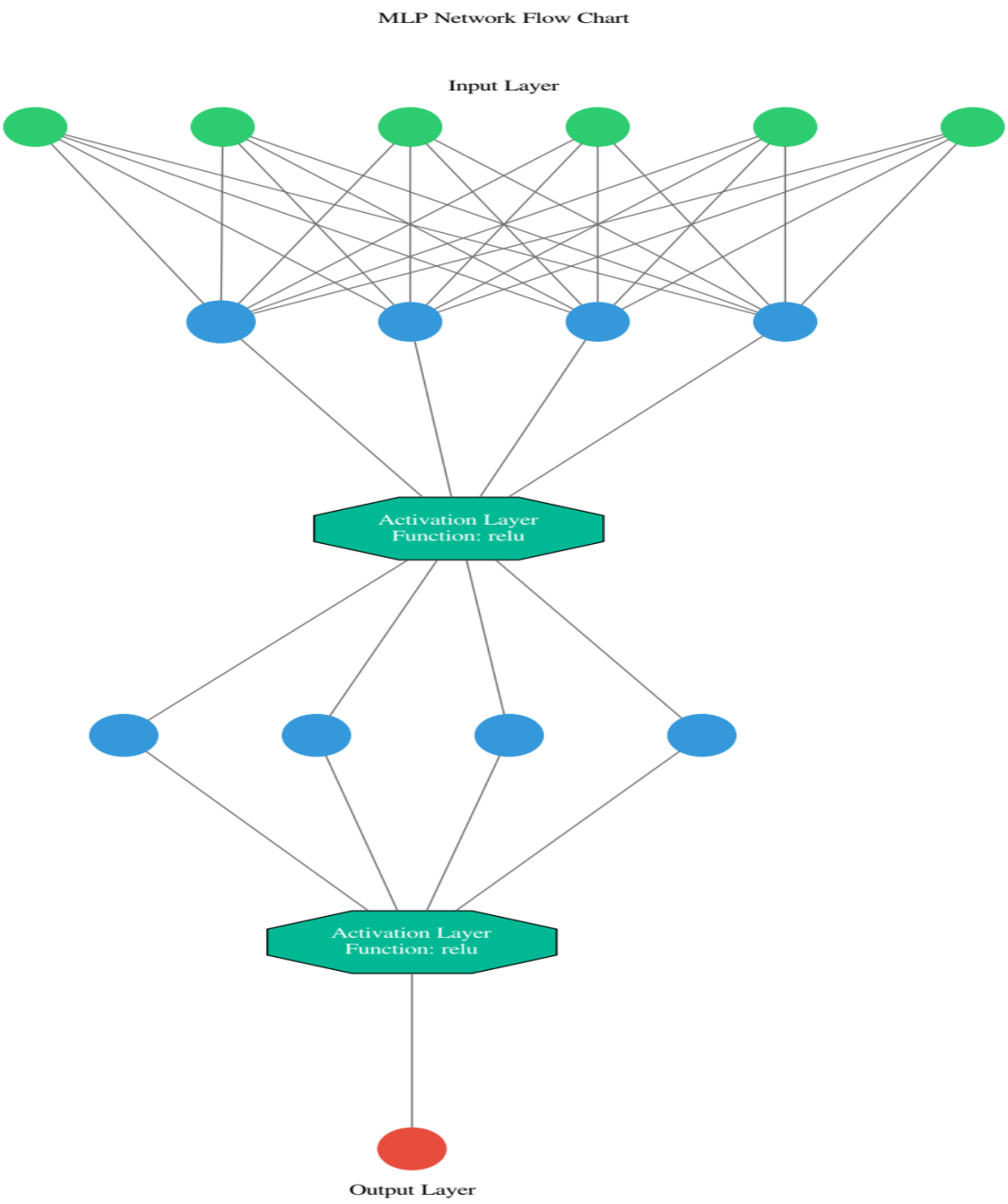


Fig. 28 Model chart of Multi-Layer Perceptron

The below image is showing implementation of MLP.

MLP (Multi Perceptron)

```
In [24]: import keras
         from keras.models import Sequential
         from keras.layers import Dense, Dropout, BatchNormalization, Activation

         Using TensorFlow backend.

In [26]: model = Sequential()
         model.add(Dense(4, input_shape= (x_train_df1.shape[1],)))
         model.add(Activation('relu'))
         model.add(Dense(4))
         model.add(Activation('relu'))
         model.add(Dense(y_train_df1.reshape(-1,1).shape[1]))

         model.compile(loss='mae', optimizer='adam', metrics=['mean_squared_error'])

In [27]: model.fit(x_train_df1, y_train_df1, epochs =20, verbose=1, batch_size=8,
                   validation_data=(x_test_df1,y_test_df1), shuffle=True)

         Train on 746 samples, validate on 187 samples
         Epoch 1/20
         746/746 [==============================] - 2s 3ms/step - loss: 0.3269 - mean_squared_error: 0.1484 - val_loss: 0.2869 - val_m
         ean_squared_error: 0.1226
         Epoch 2/20
         746/746 [==============================] - 1s 806us/step - loss: 0.2563 - mean_squared_error: 0.0988 - val_loss: 0.2473 - val
         _mean_squared_error: 0.0907
         Epoch 3/20
         746/746 [==============================] - 1s 810us/step - loss: 0.2237 - mean_squared_error: 0.0764 - val_loss: 0.2170 - val
         _mean_squared_error: 0.0726
         Epoch 4/20
         746/746 [==============================] - 1s 872us/step - loss: 0.1988 - mean_squared_error: 0.0618 - val_loss: 0.1916 - val
         _mean_squared_error: 0.0584
         Epoch 5/20
         746/746 [==============================] - 1s 832us/step - loss: 0.1757 - mean_squared_error: 0.0481 - val_loss: 0.1704 - val
         _mean_squared_error: 0.0469
         Epoch 6/20
         746/746 [==============================] - 1s 941us/step - loss: 0.1549 - mean_squared_error: 0.0386 - val_loss: 0.1489 - val
         _mean_squared_error: 0.0363
```

Fig. 29 MLP Model Building

Keras API is employed to build multi-layer perceptron model. Sequential method is used for building multi-layer perceptron model. The input layer parameters are merged in the Dense_1 hidden layer. The number of nodes are four and the activation function employed is rectified linear unit. After that second hidden layer is added denoted by Dense_2. Number of nodes chosen are again four and activation function is relu. The last layer Dense_3 is the output layer. The model learning rate and evaluation metrics are then configured by compiling method. The optimizer for learning rate improvement is Adaptive Moment Estimation (adam) and model is evaluated by mean absolute error and mean square error. The model is then trained over training set and by measuring the loss metrics the model hyperparameters are tuned.

3.4.7 Vector Auto-Regressive (VAR)

A VAR model can recognize and make use of interconnections between numerous variables. Every variable in this model is a linear function of its previous values and other variables' previous values. VAR is a very simple model to implement and is important in the forecasting of better results and also helps properly describe the active behavior of data.

Auto regression in a statistical model in which future values are predicted based on past data records. In multiple regression models, a linear combination relation is developed among features of the data set. But in auto regression randomness between past data is observed and relation is developed on the basis of these lags randomness. Vector auto regression is the extension of uni-variate auto regression and is useful when more than one time series are needed to be predicted in the same data set.

We've seen that an autoregressive moving average ARMA (p, q) model is described by the following:

$$y_t = c + \emptyset_1 y_{t-1} + \emptyset_2 y_{t-2} + \ldots\ldots + \emptyset_p y_{t-p} + \emptyset_1 \in_{t-1} + \emptyset_2 \in_{t-2} + \ldots\ldots + \emptyset_p \in_{t-q} + \in_t$$

A *K*-dimensional VAR model of order (*p*, *q*) considers each variable y_t in the system.

For example, the system of equations for a 2-dimensional VAR (1,1) model is:

$$y_1, t = c_1 + \emptyset_{11}, 1y_1, t-1 + \emptyset_{12}, 1y_2, t\text{-}1 + \emptyset_{11}, 1 \in_1, t\text{-}1 + \emptyset_{12}, 1 \in_2, t\text{-}1 + \in_1, t$$

$$y_2, t = c_2 + \emptyset_{21}, 1y_1, t-1 + \emptyset_{22}, 1y_2, t\text{-}1 + \emptyset_{21}, 1 \in_1, t\text{-}1 + \emptyset_{22}, 1 \in_2, t\text{-}1 + \in_2, t$$

where the coefficient $\emptyset_{ii}$, *i* captures the influence of the lth lag of error $\in_i$ on itself, the coefficient $\emptyset_{ij}$, *l* captures the influence of the lth lag of error $\in_j$ on $\in_i$ and $\in_1$,t and $\in_2$,t is residual white noise.

The VAR equation is given below:

$$y_{1,t} = c_1 + \phi_{11,1} y_{1,t-1} + \phi_{12,1} y_{2,t-1} + e_{1,t}$$
$$y_{2,t} = c_2 + \phi_{21,1} y_{1,t-1} + \phi_{22,1} y_{2,t-1} + e_{2,t},$$

where, e_1 and e_2 are processes that maybe correlated.

Φ is the coefficient

y_1 and y_2 are the variables

(Zivot & Wang 2007).

The below image has shown that how VAR model is built.

Vector Auto Regression

In [22]:
```
# training model
var = VAR(endog=x_train_df1)
var_fit = var.fit()
```

In [23]:
```
# predicting test set
prediction = var_fit.forecast(var_fit.y, steps=len(x_test_df1))
```

```
/home/kirmani/anaconda3/envs/env1/lib/python3.6/site-packages/statsmodels/base/wrapper.py:35: FutureWarning: y is a deprecated
alias for endog, will be removed in version 0.11.0
  obj = getattr(results, attr)
```

In [24]:
```
# SAMPPLE test prediction
print('Predicted: %.2f' %prediction[0,6])
print('True Value: %.2f' %y_test_df1[0])
```

```
Predicted: 17012.51
True Value: 16926.90
```

Fig. 30 VAR Model Building

Auto regression in a statistical model in which future values are predicted based on past data records. In multiple regression models, a linear combination relation is developed among features of the data set. But in auto regression randomness between past data is observed and relation is developed on the basis of these lags randomness. Vector auto regression is the extension of uni-variate auto regression and is useful when more than one time series are needed to be predicted in the same data set. To develop vector auto regression model statsmodel package is used. The number of observations and number of variables are defined to build the model. The train set variables are fitted in the model with the number of observations. This trained model is then used to predict the output of test set. Predicted values error is measured by evaluation metrics and on the basis of results it is decided whether to deploy the model for real time forecasting or not.

The given data set of stocks and household are fed in vector auto-regression without scaling and both models are trained. The results are evaluated by evaluation metrics and by comparison plot between true and predicted values. The results obtained are not good because the model has only single variable output and vector auto regression is for multiple variables.

5 Analysis and Discussion of Results

5.1 Experiment and Data Analysis

In this experiment, we have used two different data sets. One is about financial data sets which is Indian bank nifty index data set and other is house hold electricity consumption data set.

5.1.1 Banknfty Index Dataset

The free float methodology is applied in calculation of the Bank Nifty. The free-float method calculates the market Bank Nifty by putting together the calculated market capitalization of underlying companies. This method does not include the locked-in shares held by the government or promoters and instead only considers the shares that are actively trading in calculating market capitalization. This is different from the full-market capitalization methods which puts into consideration all the shares. The calculation is done by multiplying the number of active banking shares with equity's price. The free-float market capitalization method gives an insight on the movement of stocks in the market. Its effectiveness in computing Nifty Index is brought about by the thorough consideration of corporate actions and other important changes and not affect the index.

5.1.2 Household Electricity Consumption Dataset

Household dataset is about electricity consumption of different domestic users and it has 2075259 instances and 8 attributes such as global active power, global reactive power, voltage, sub metering 1, sub metering 2, sub metering 3 , date and time.

5.1.3 Exploratory Data Analysis of Stock and Household Electricity Data sets

Here we have performed initial exploratory data analysis of both stock and house hold consumption data sets. First I have performed EDA on stock data sets which is aggregated by hours,

Time series distribution by hours:

Heat Map

Fig.31 Stocks Time Series Distribution by Hours

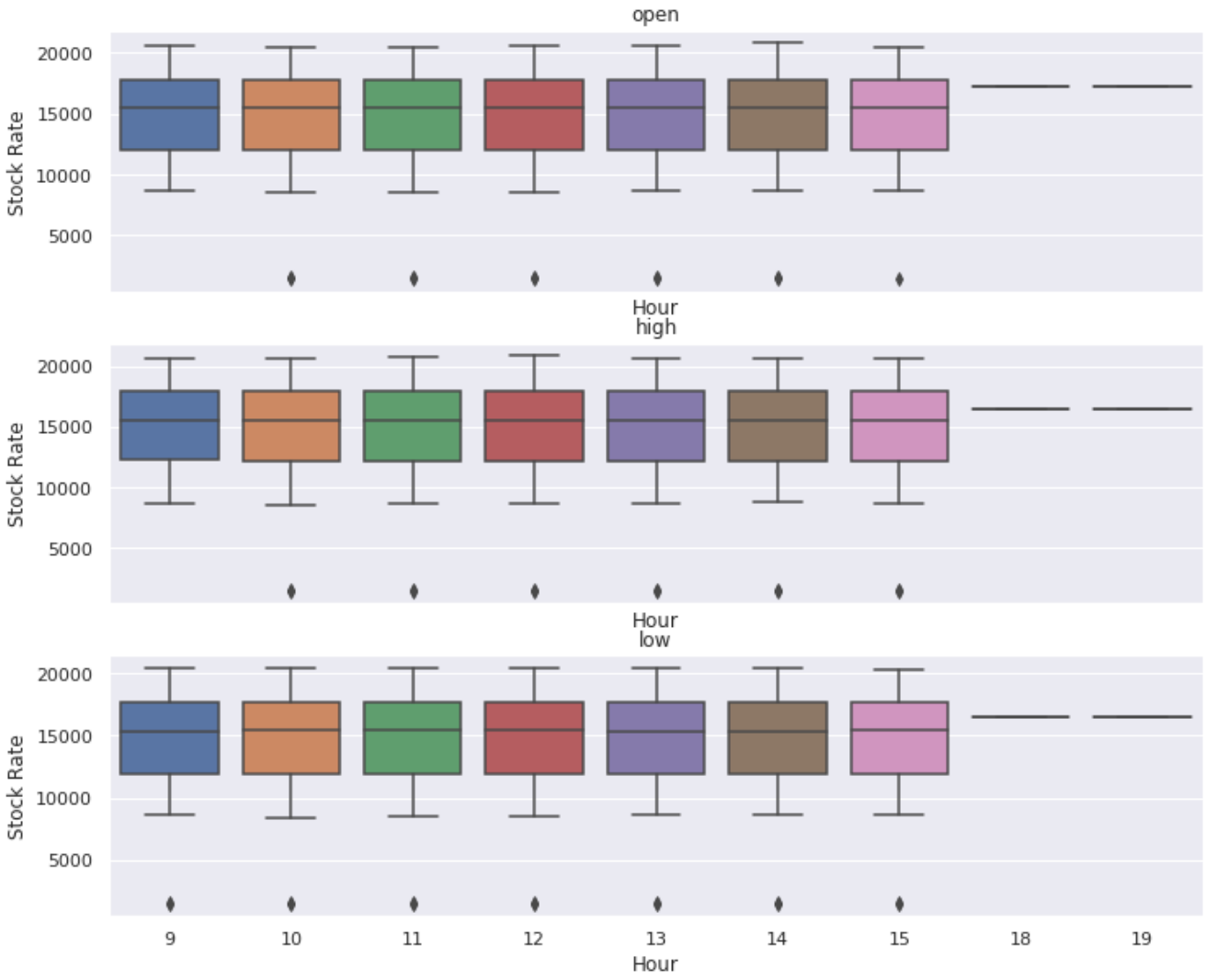


Fig.32 Box Plot (Stocks Time Series Distribution by Hours)

Histogram and Curve Distribution

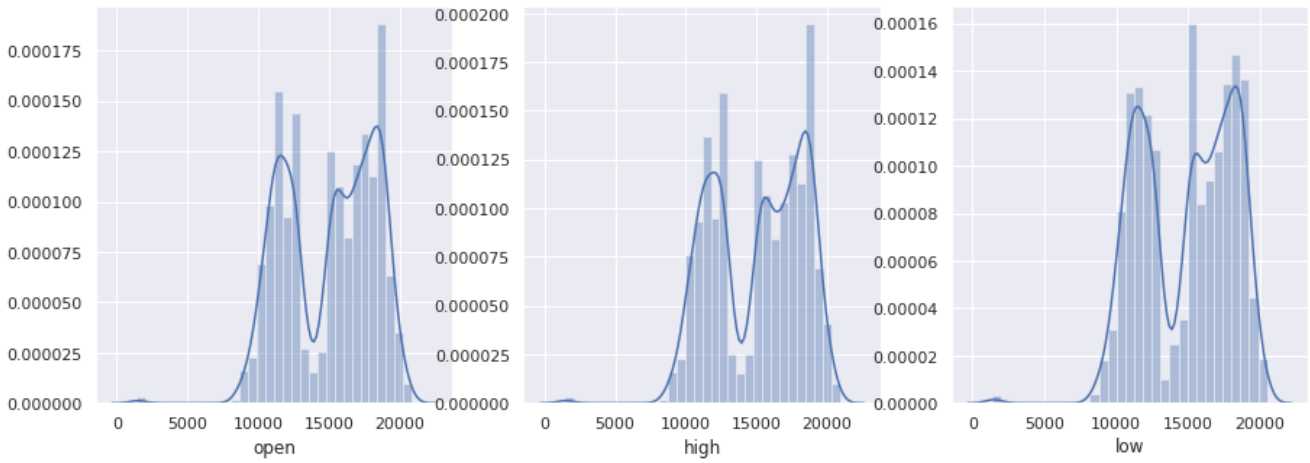


Fig.33 Histogram and Curve Distribution

Trend line by hours

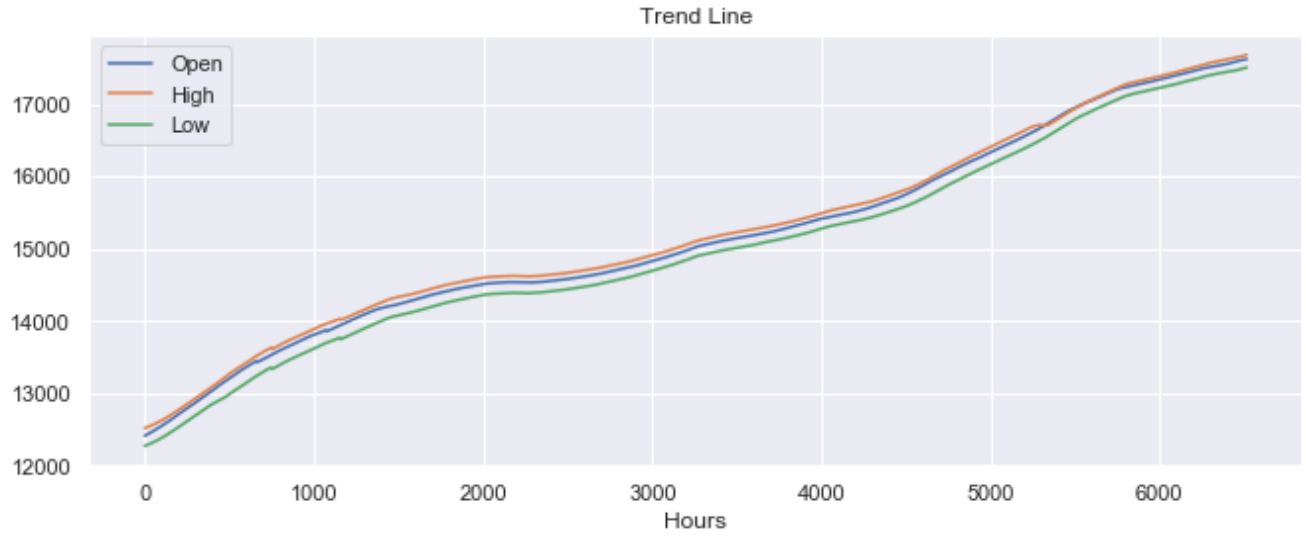
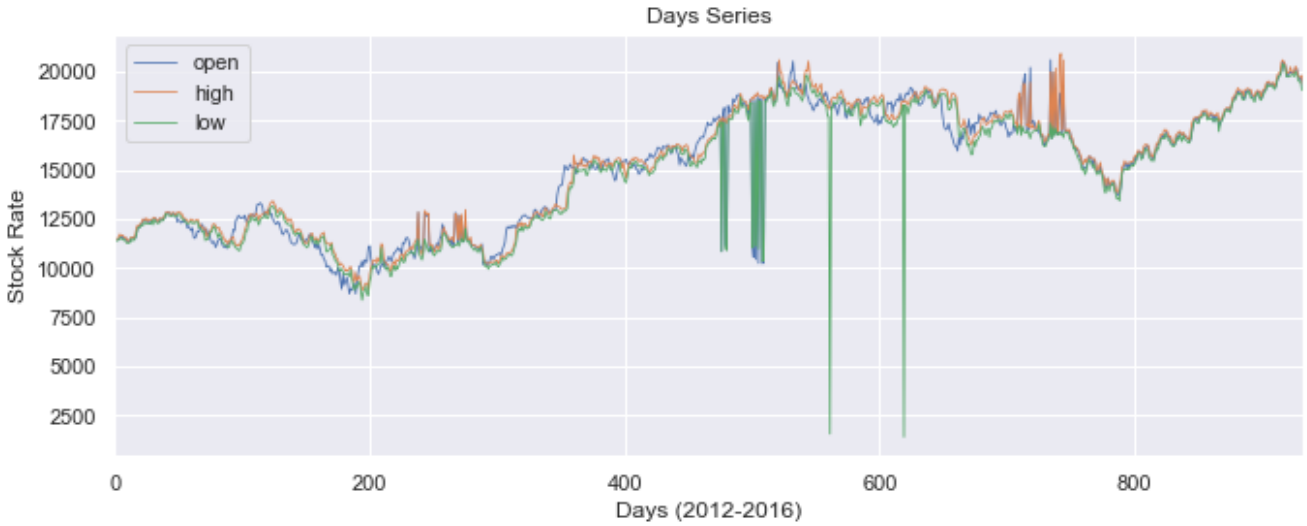


Fig.34 Stocks Trend Line by Hours

Then EDA is performed on stock data sets which is aggregated by days,



Time series distribution by Days

Fig.35 Stocks Time Series Distribution by Days

Box Plots

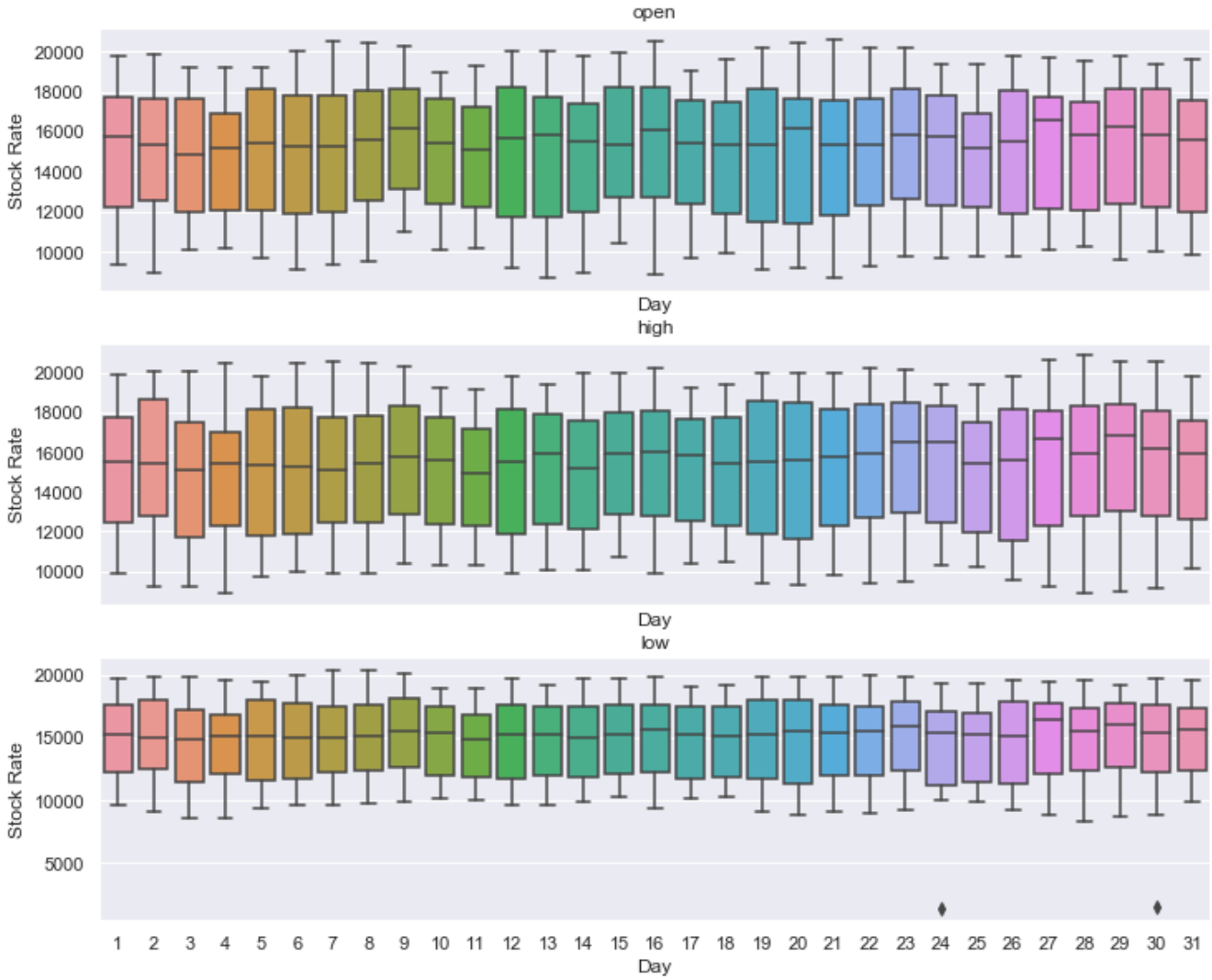


Fig.36 Box Plot (Stocks Time Series Distribution by Days)

Histogram and Curve Distribution

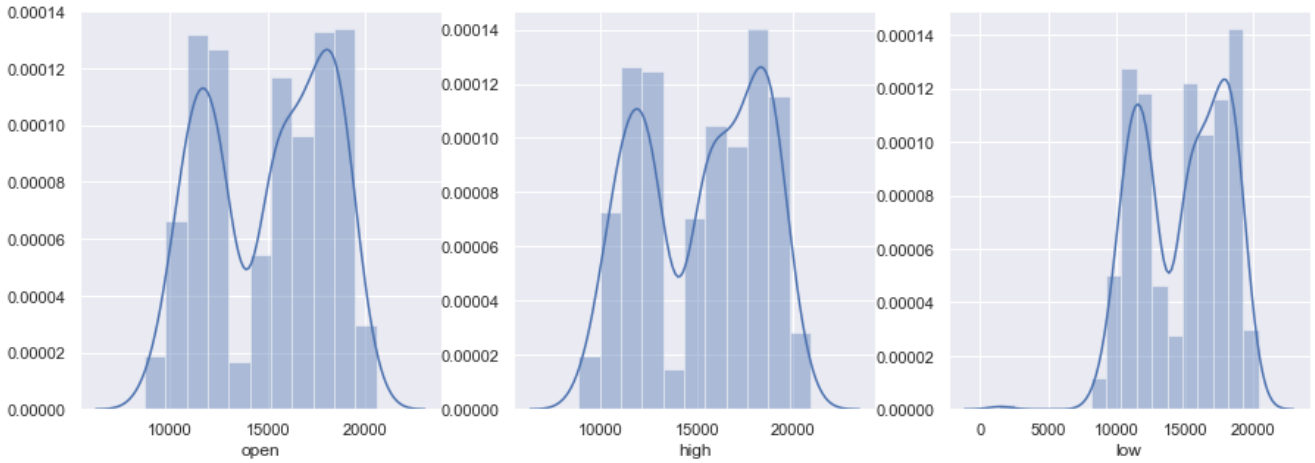


Fig.37 Histogram and Curve Distribution

Trend Line by Days

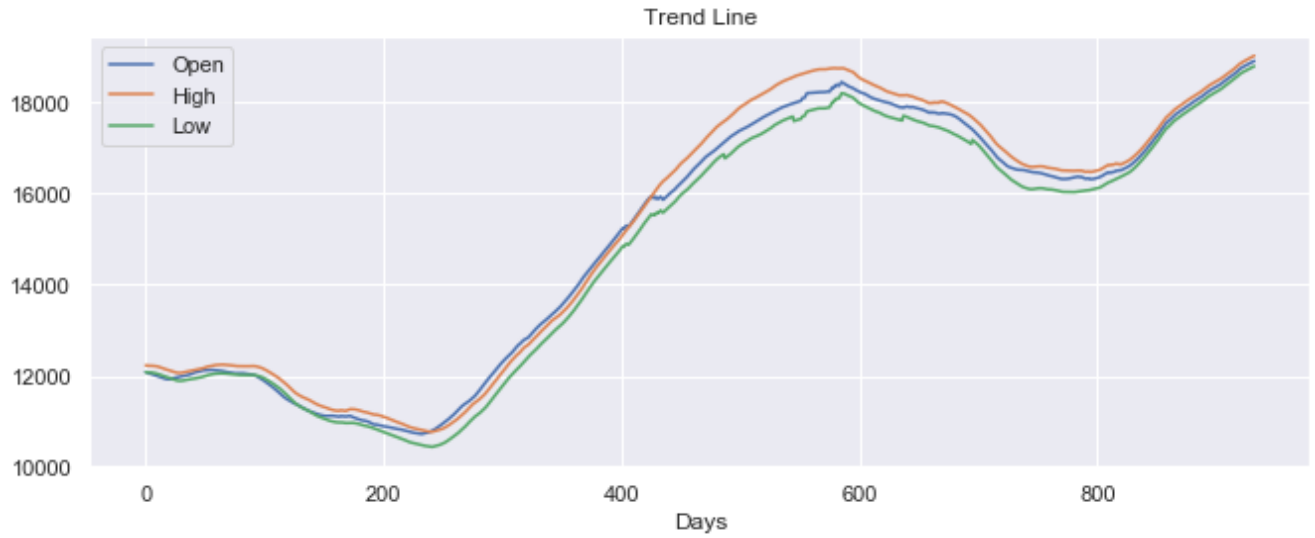


Fig.38 Stocks Trend Line by Days

Now EDA on stock data sets aggregated by months is performed.

Time series distribution by Months:

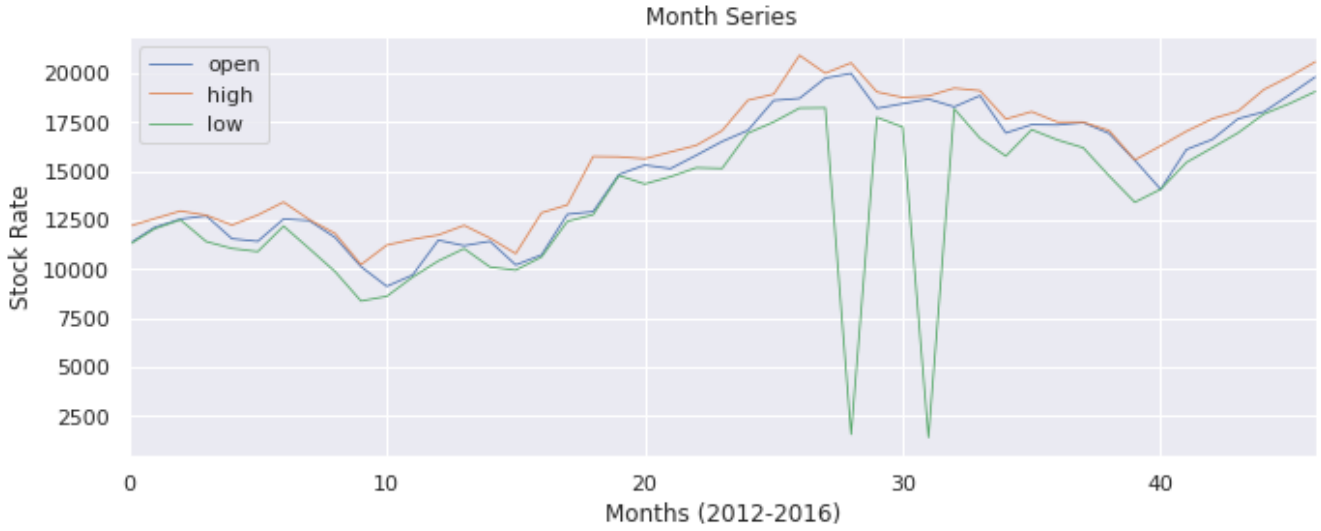


Fig.39 Stocks Time Series Distribution by Months

Box Plots

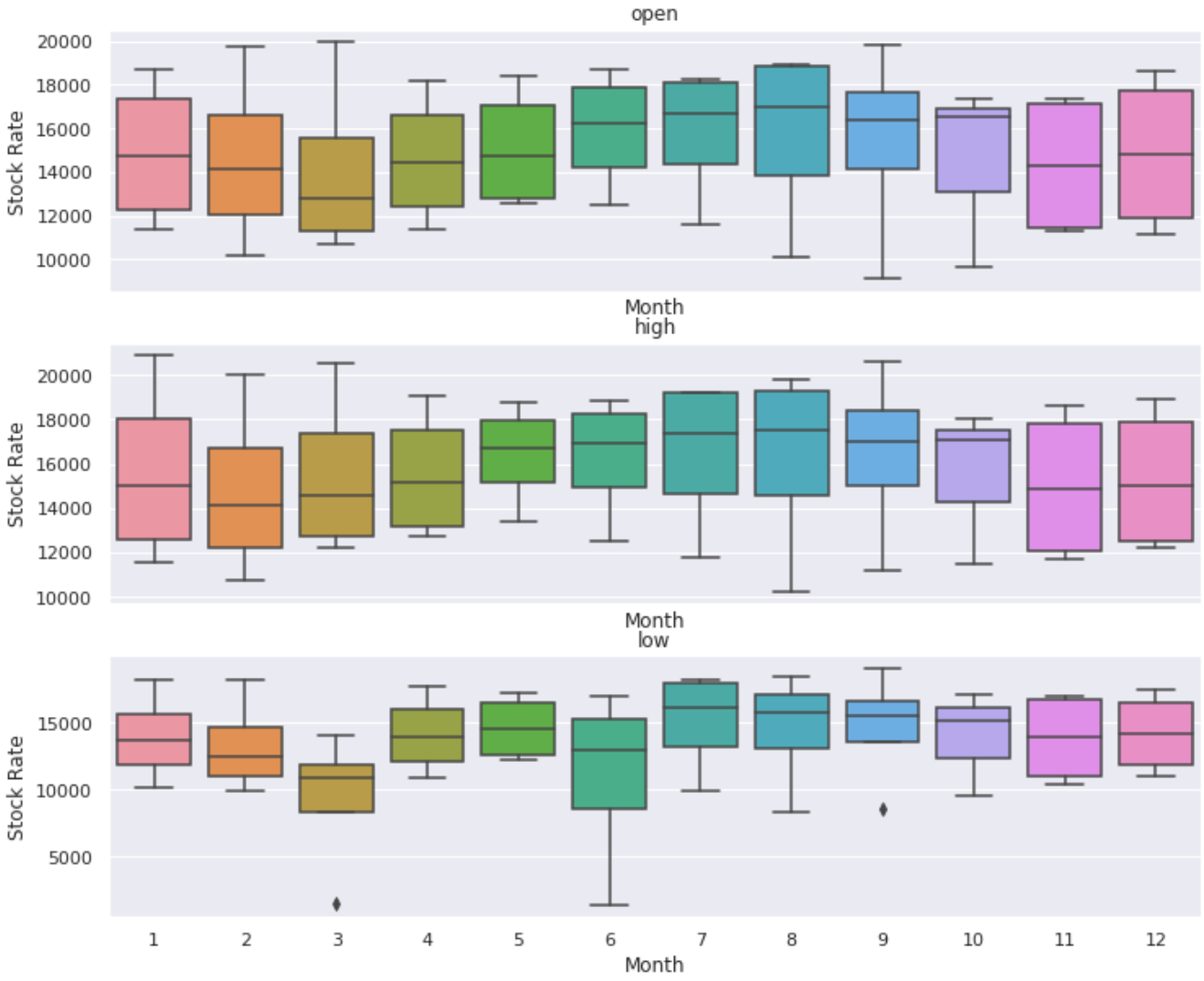


Fig.40 Box Plot (Stocks Time Series Distribution by Hours)

Histogram and Curve Distribution

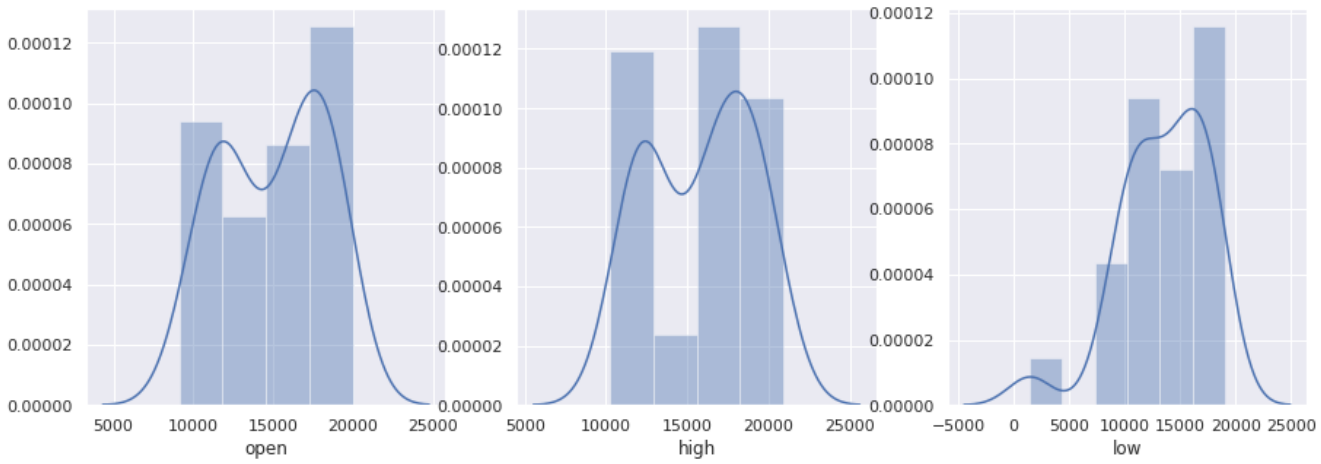


Fig.41 Histogram and Curve Distribution

Trend Line by Months

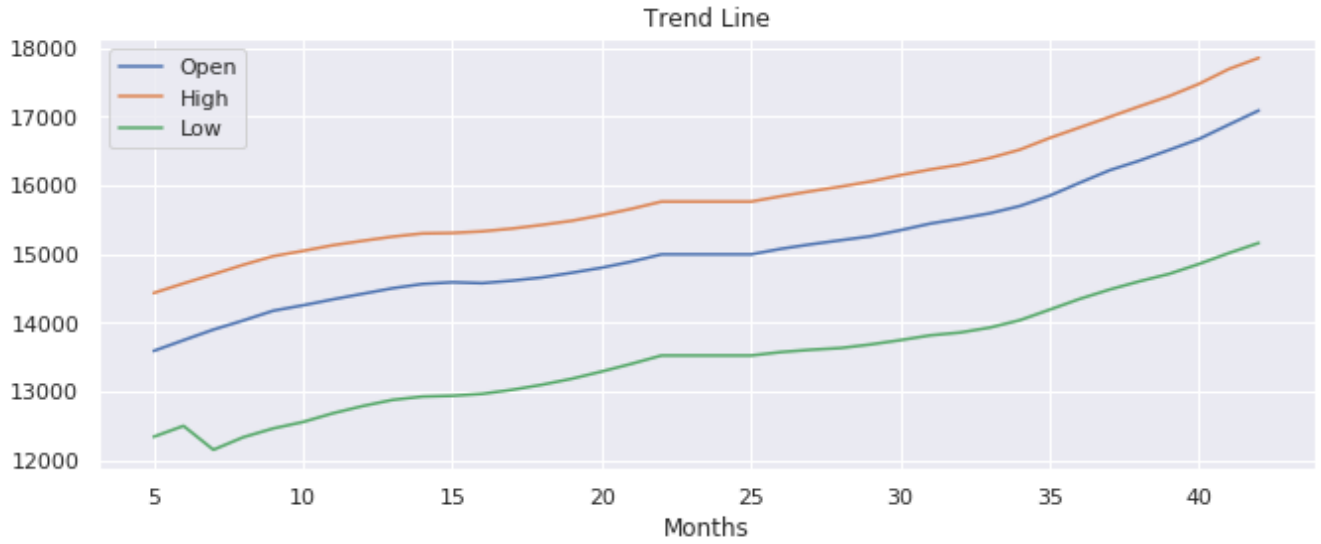
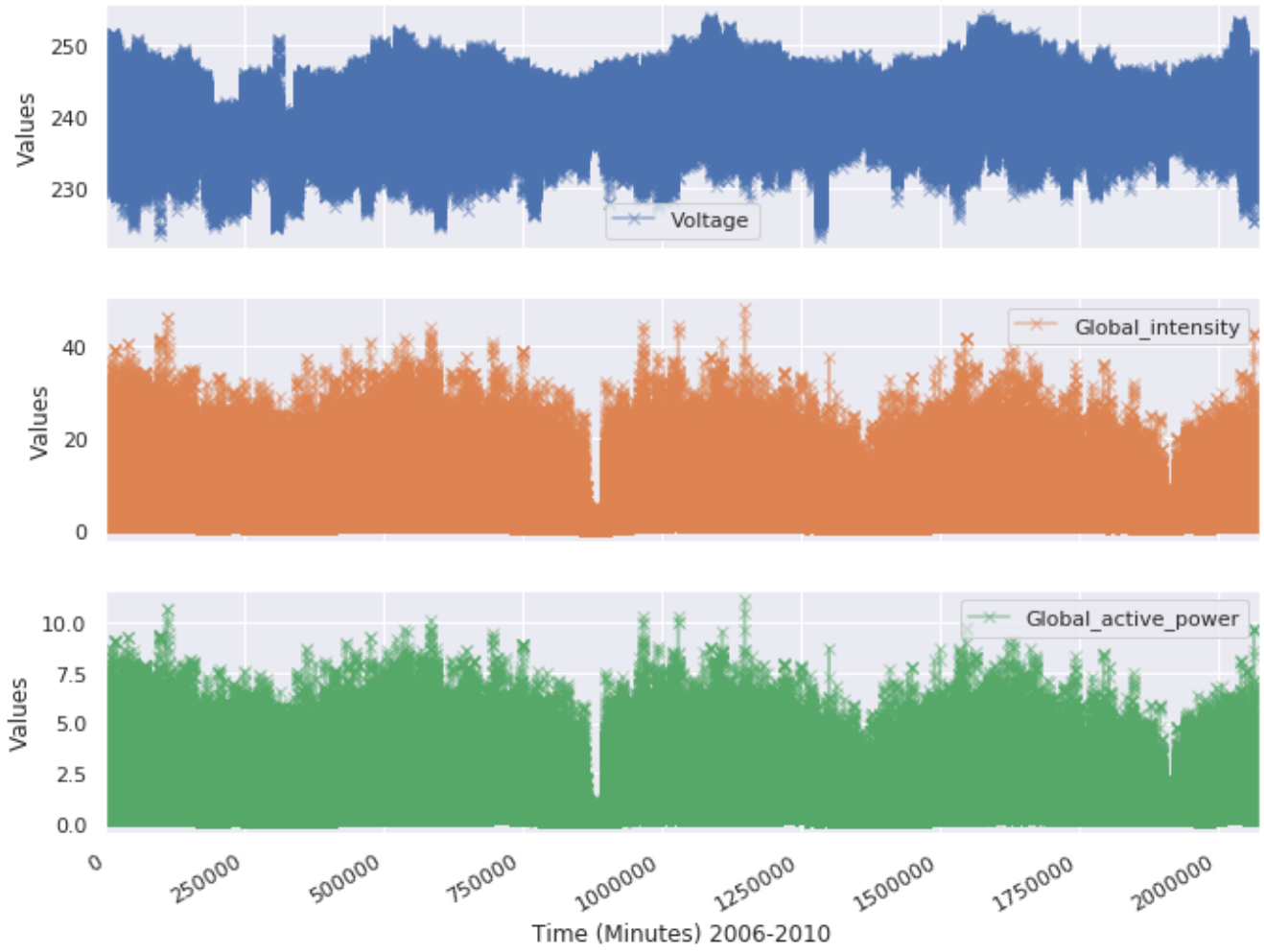
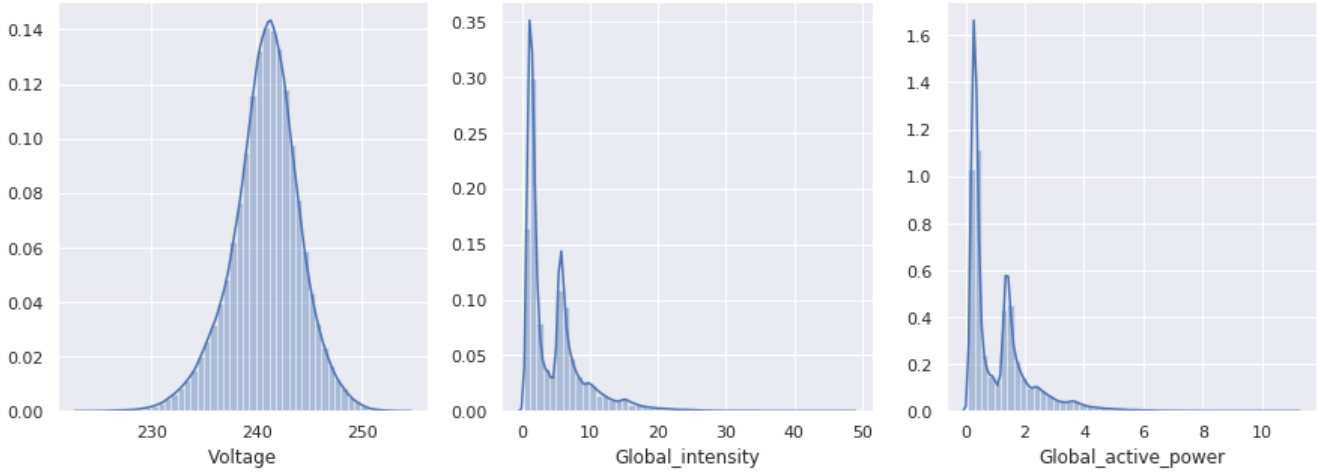


Fig.42 Stocks Trend Line by Months

EDA is performed on house hold electricity consumption data set.


Values
250
240
230
Voltage
Global_intensity
40
20
0
Global_active_power
10.0
7.5
5.0
2.5
0.0
0
250000
500000
750000
1000000
1250000
1500000
1750000
2000000
Time (Minutes) 2006-2010


0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
230
240
250
Voltage
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
10
20
30
40
50
Global_intensity
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
10
Global_active_power

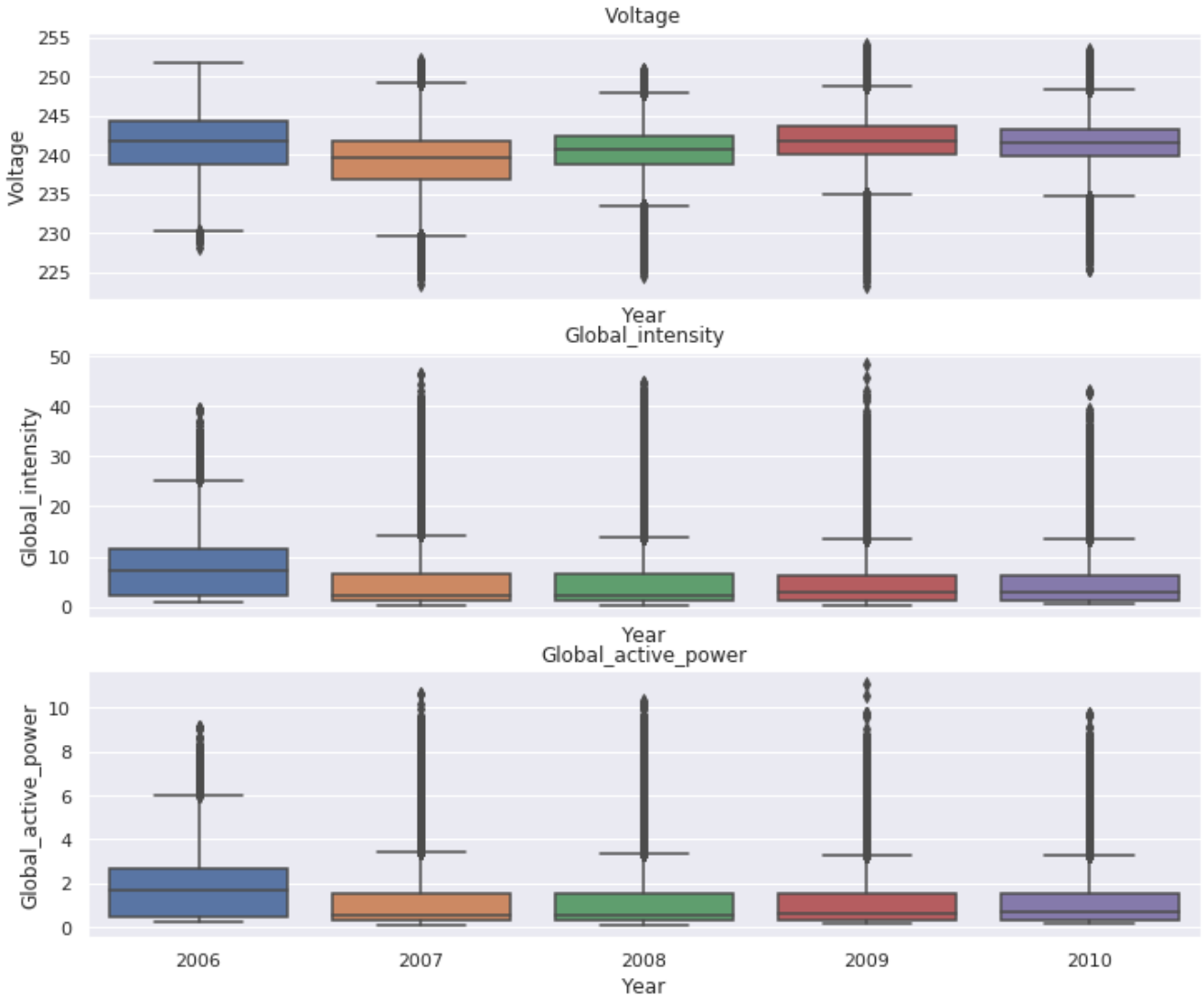


Fig.43 Household Electricity Power Consumption Ditsribution

5.2 Factors Affecting Performance of a Model

Many times model performance is dependent on the data. If a model is giving bad performance on a data set, if this data is tuned or number samples are increased, the same model may have improvements in results. In short words, if number of observations in a data set are varied or the values are scaled then the performance of a model will be affected. Therefore, it is very important while multiple models are compared and evaluated, all of them must have same data. By changing data slightly in a model the symmetry of data is changed and comparing them is illegitimate. Many times there are outliers in the observed data, and adding these values effects the efficiency of the model. The outlier creates uneven distribution of data points, therefore, these values are removed from the observed data set.

Apart from outliers in the dataset, many times the data observations are unevenly distributed. To tackle this situation, data is normalized. Usually nsormalization is scaling data between low (0) and high (1) values. The minimum value in a data set becomes 0 and maximum value becomes 1, and rest of the values are in between them. It is also a good technique when distribution of data is not known or the distribution of data is not gaussian (bell shaped). (Shcherbakov et al. 2013)

If the normalization is applied then it should be kept in mind that the model will always require normalized input values from same function which was used for training input features and output predicted is also normalized which needs to be inversely transform. In this research, MAE (mean absolute error), MSE (mean square error) and RMSE (root mean square error) are used. (Shcherbakov et al. 2013)

5.3 Evaluation Matrix

To summarize the capabilities and skills of the model used in predicting the outputs, a time series prediction performance measure is carried out. The problem is usually choosing how to interpret the results and the measure to use since there are many to choose from. The forecast error values are averaged, mean absolute error, and the forecast values are made positive. The squared forecast error values are also averaged to give the mean squared error.

The mean squared error score is increased by squaring the very large forecast errors which drags the mean of squared forecast errors. The forecast that have large wrong forecast therefore gets a bigger score.

It can be transformed back into the original units of the predictions by taking the square root of the mean squared error score. This is called the root mean square error, or RMSE.

If same pre-processing procedures such as data cleaning are performed and the time series forecasting performance is evaluated with the same scale then it would be advisable to use MAE (mean absolute error) and RSME (root mean squared error) however for the different scales values these measurements are not applicable.

The evaluation metric is used to evaluate the model and based on its result we need to decide to deploy it for real time forecasting or not. Additionally, multiple models are developed, and the purpose is to choose the model with optimal results. However, there are

certain criteria that must be fulfilled to evaluate these models. The evaluation metrics are of two types. If the output values are finite, then classification evaluation metrics are used such as confusion matrix, accuracy, precision, recall and F1 score. While for continuous values, regression evaluation metrics are employed. The regression evaluation metrics are mean absolute error, mean square error, root mean square error, mean absolute percentage error and r2 score.

Since the output of stocks and household power consumption are continuous values, therefore, regression metrics are employed in this research.

5.3.1 Mean Absolute Error

Mean absolute error (MAE) measures the average magnitude of the errors in a set of predictions, without considering their direction. It's the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight. (Shcherbakov et al. 2013)

$$MAE = \frac{1}{n}\Sigma \mid y_j - y_i \mid$$

where, y_j is true output value

y_i is predicted output value

5.3.2 Mean Square Error

Mean square error is the average of squared differences between prediction and actual observation.

$$MSE = \frac{1}{n}\Sigma (y_j - y_i)^2$$

By observing these values model performance is observed. The lower these values are the better predictions are made by the model. (Shcherbakov et al. 2013)

5.3.3 Root Mean Square Error

It measures how far the data points are from regression line. In simple words it means how spread data points are from line of best fit. It is the square root of mean square error.

Mathematically it is given by:

$$RMSE = \sqrt{(\frac{1}{n}\Sigma\ (y_j - y_i)^2)}$$

(Shcherbakov et al. 2013).

5.4 Results

5.5 Results by Month Grouping

The results obtained on test set from different trained models is shown in following table. By observing all values of evaluation metrics, best results model is selected. If we look at the results of LSTM and random forest trained models, their results are optimum for month grouping data set. The generalized models of LSTM and random forest are efficient for real time predictions.

The test set predicted results are evaluated by MAE and MSE metrics. The results for LSTM model are shown below:

5.5.1 Results of LSTM

The trained LSTM model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

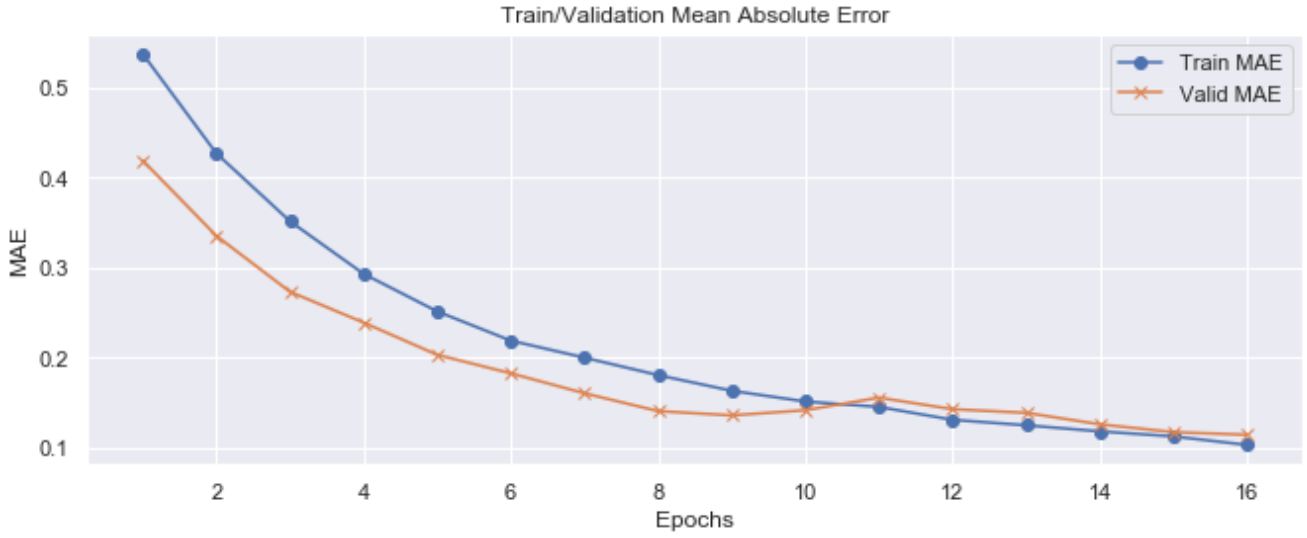


Fig.44 LSTM Mean Absolute Error

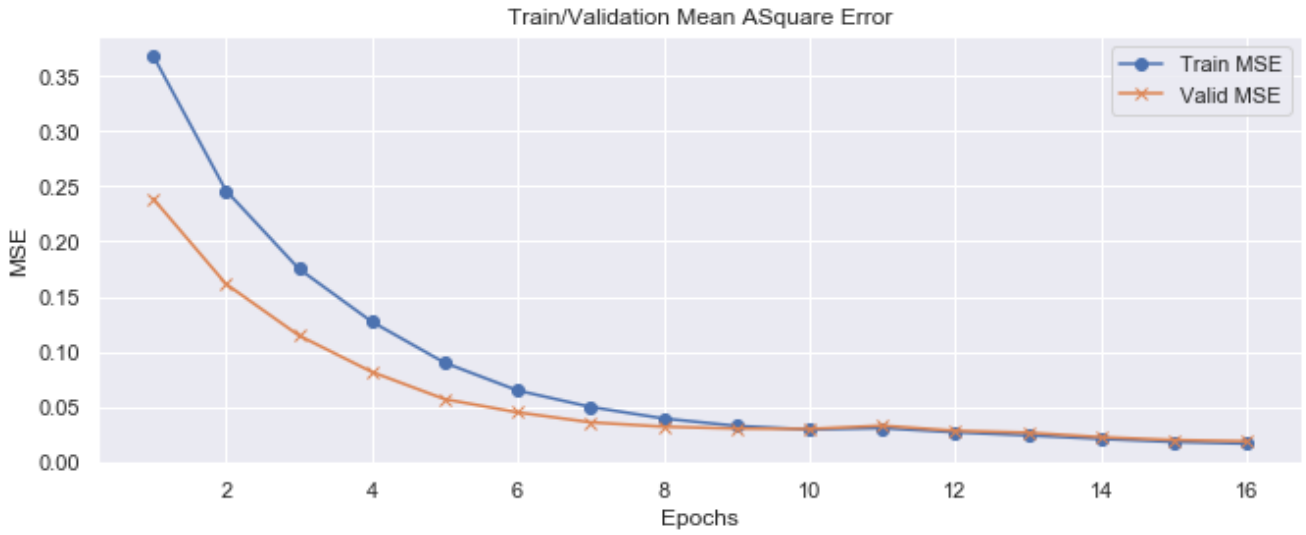


Fig.45 LSTM Mean Square Error

Below is the test set predicted and true values of stock with respect to time,

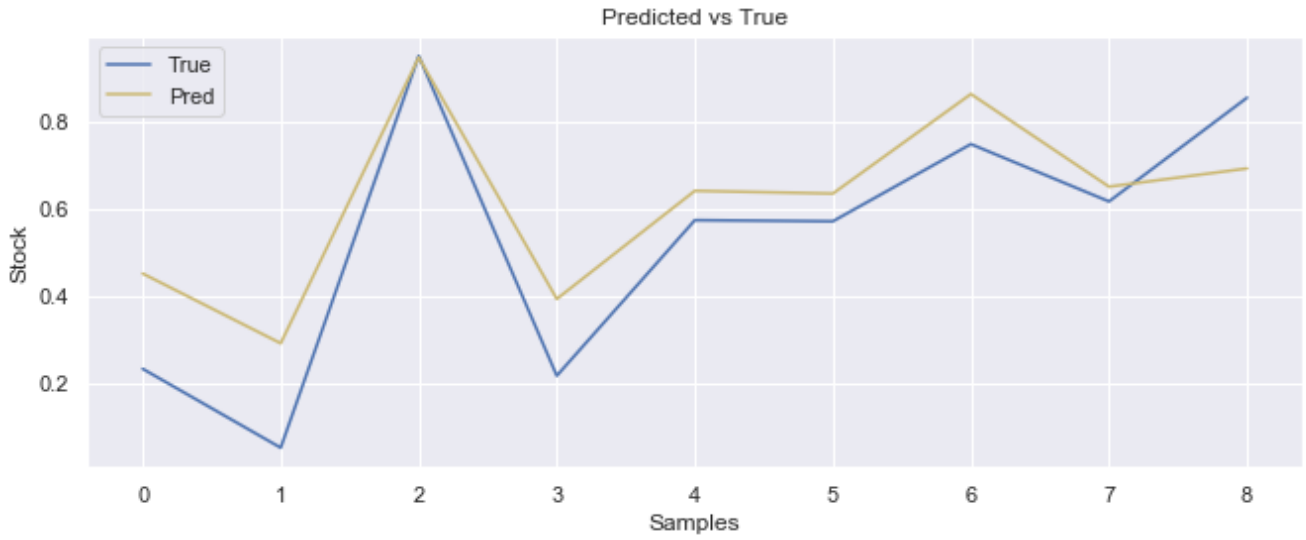


Fig.46 LSTM Predicted vs True Stock Values

5.5.2 Results of CNN

The trained CNN model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

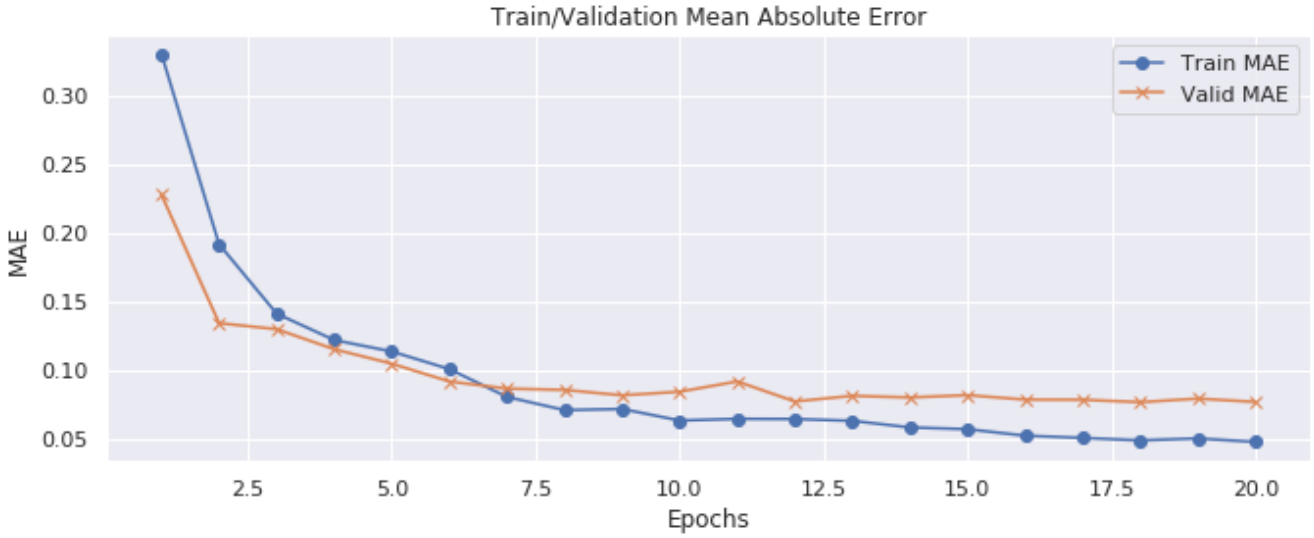


Fig.47 CNN Mean Absolute Error

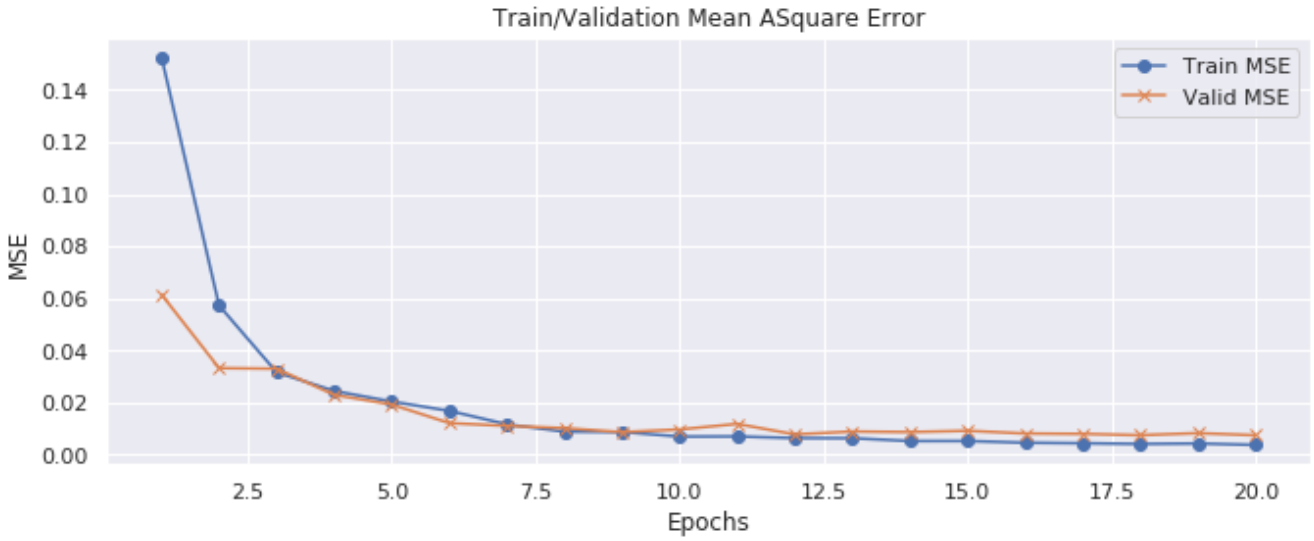


Fig.48 CNN Mean Square Error

Below is the test set predicted and true values of stock with respect to time,

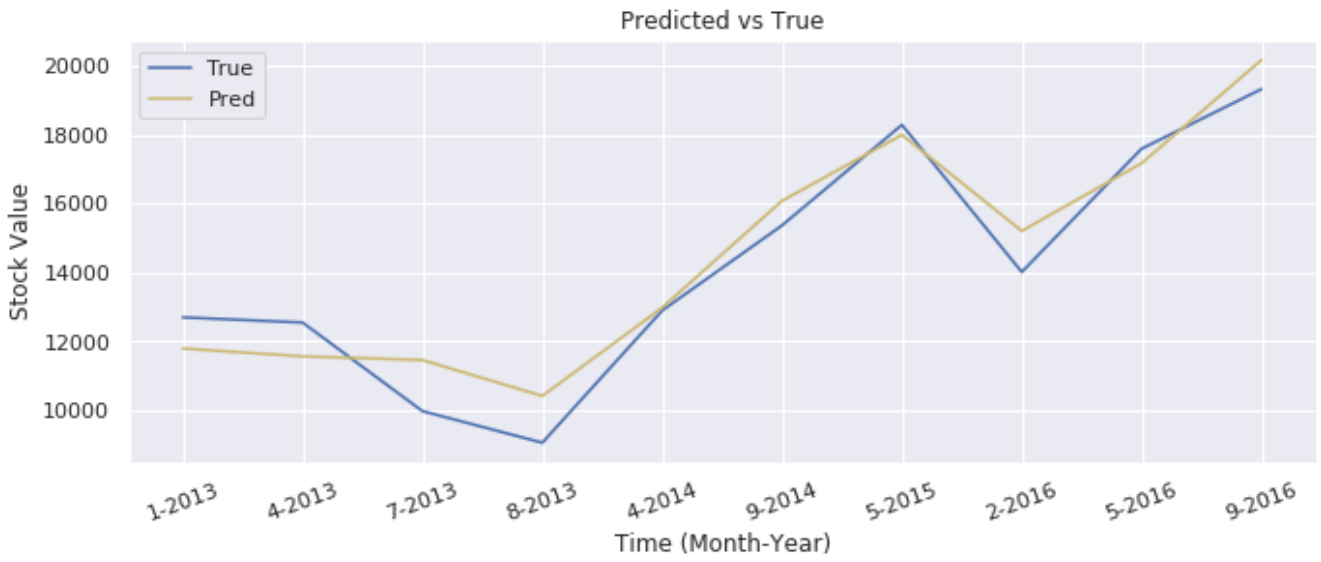


Fig.49 CNN Predicted vs True Stock Values

5.5.3 Results of GRU

The trained GRU model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

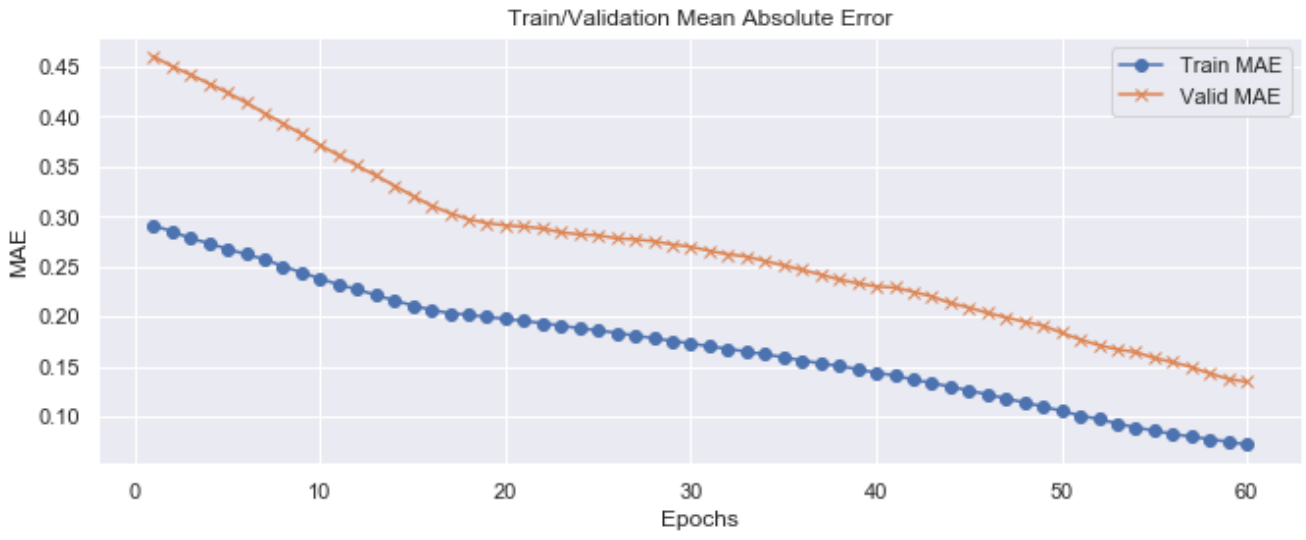


Fig.50 GRU Mean Absolute Error

Fig.51 GRU Mean Square Error

Below is the test set predicted and true values of stock with respect to time,

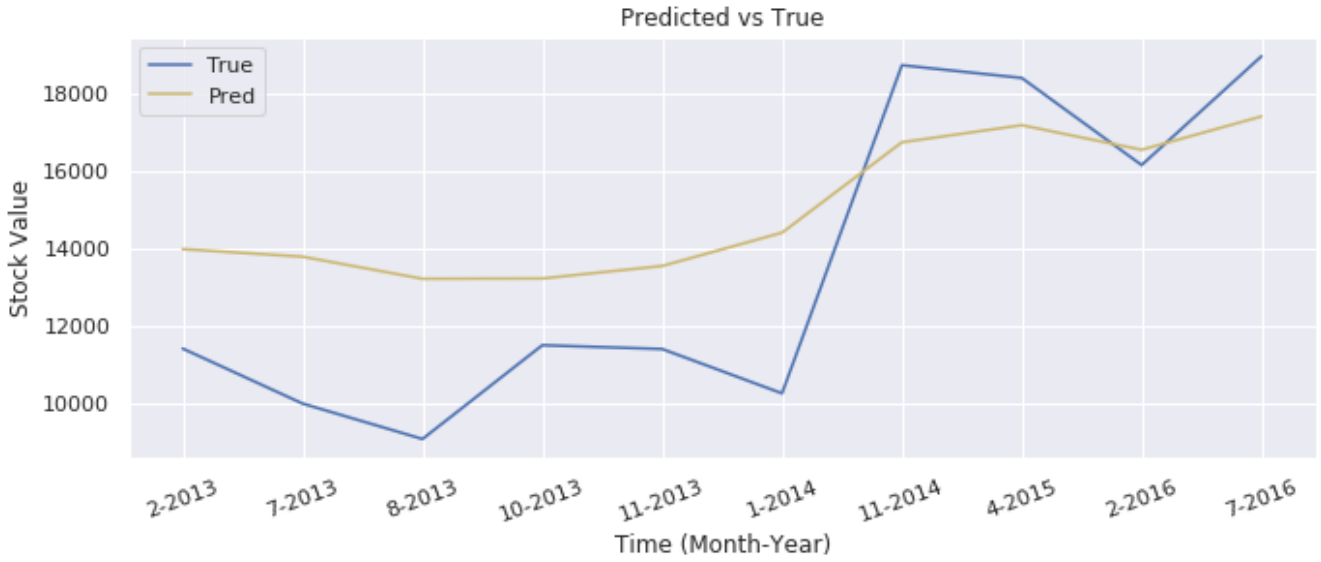


Fig.52 GRU Predicted vs True Stock Values

5.5.4 Results of MLP

The trained MLP model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

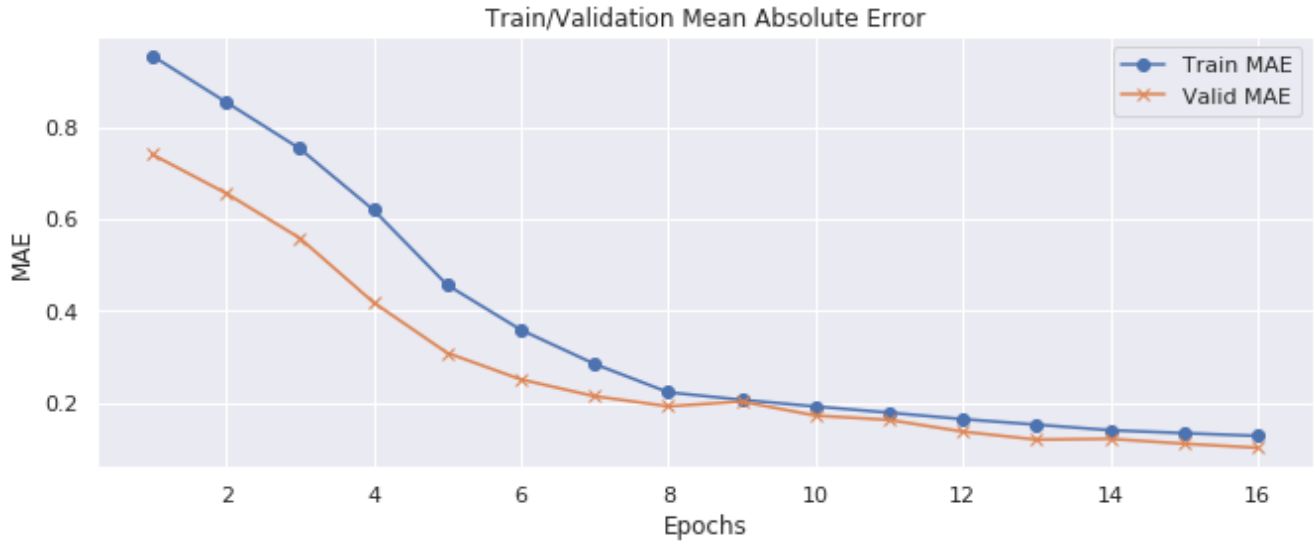


Fig.53 MLP Mean Absolute Error

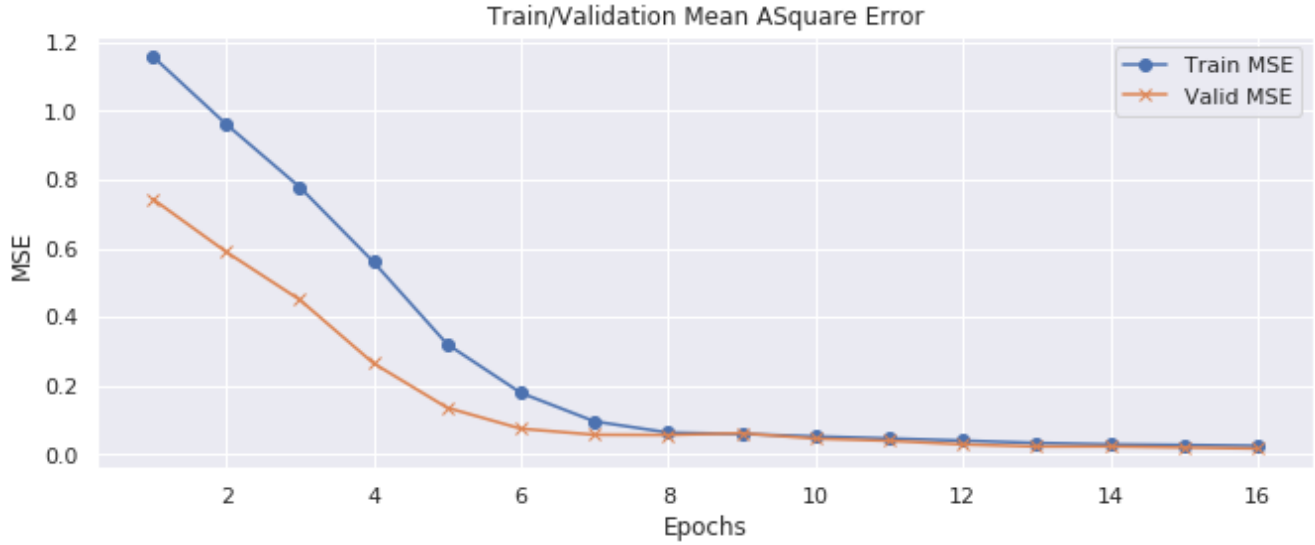


Fig.54 MLP Mean Square Error

Below is the test set predicted and true values of stock with respect to time,

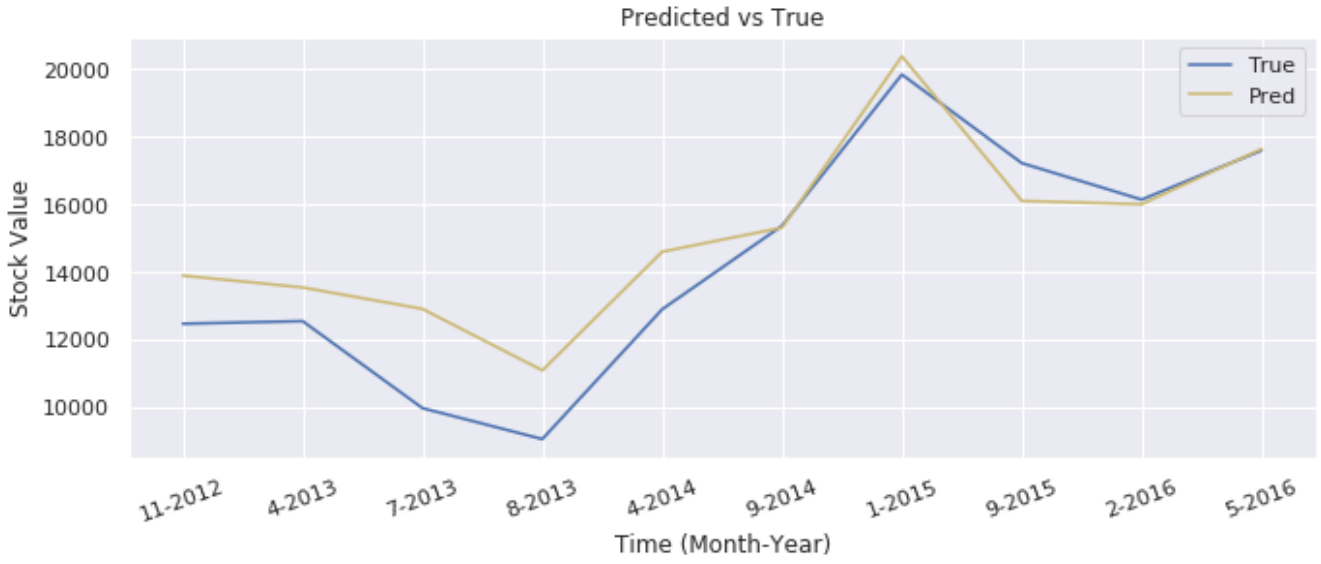


Fig.55 MLP Predicted vs True Stock Values

5.5.5 Results of Random Forest

Given below are the test set predicted and true values of stock with respect to time,

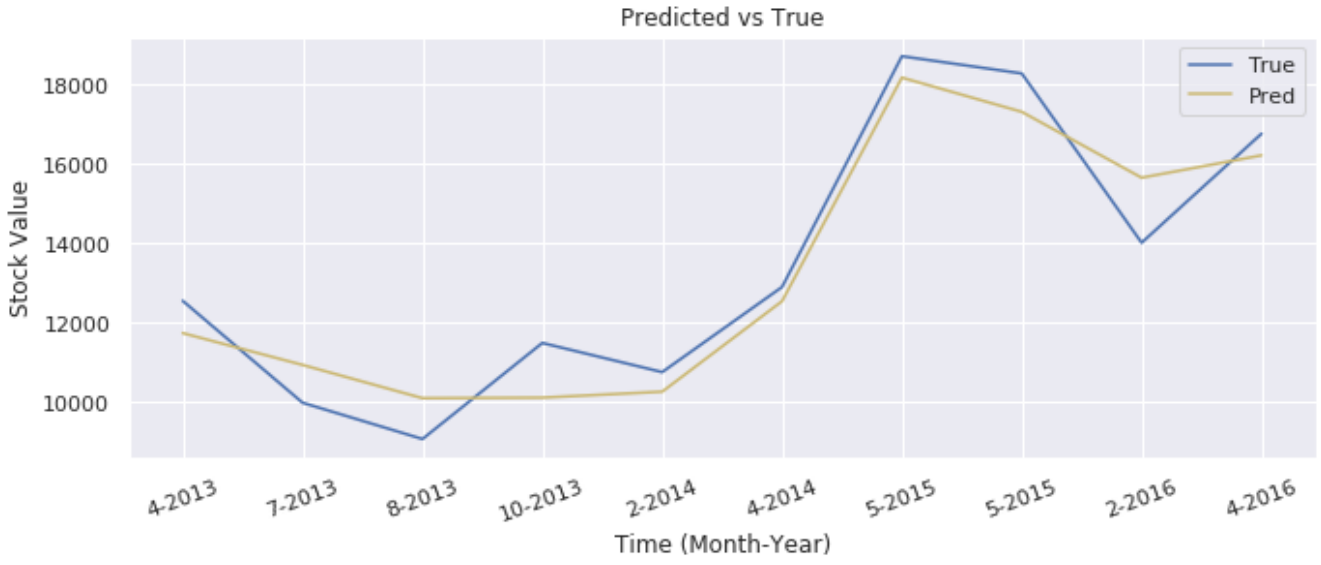


Fig.56 RF Predicted vs True Stock Values

5.5.6 Results of VAR

Given below are the test set predicted and true values of stock with respect to time,

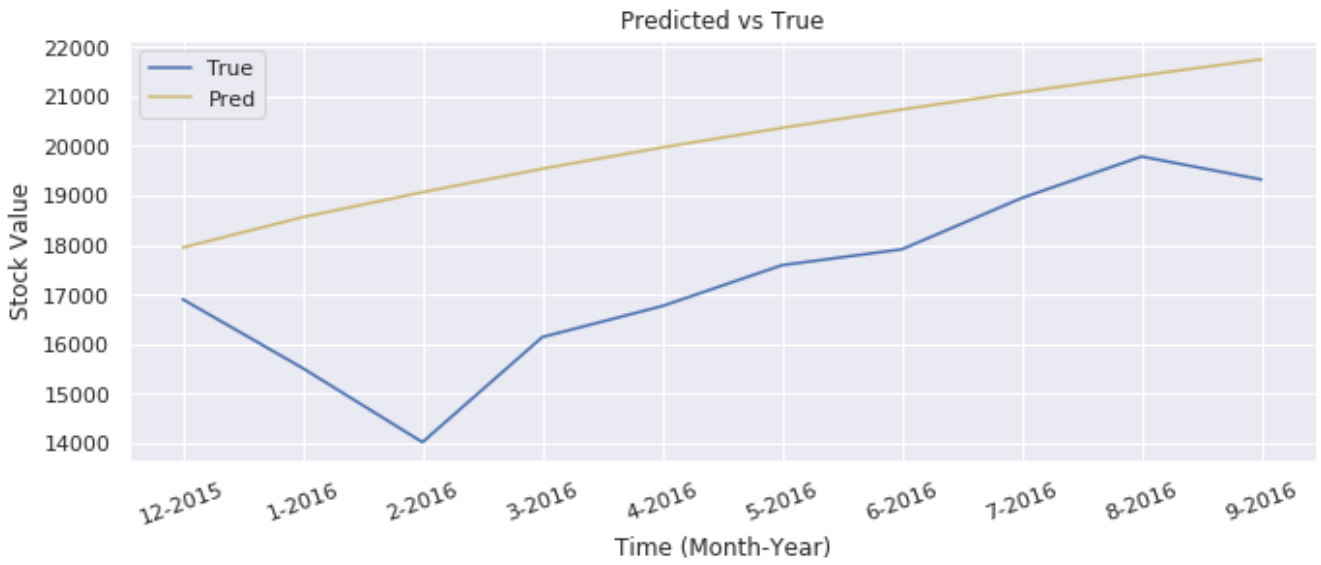


Fig.57 VAR Predicted vs True Stock Values

The result table is shown below:

Table:2. Month Grouping Results

Month Group	**Random Forest**	**MLP**	**CNN**	**LSTM**	**VAR**	GRU
R2-Score (%)	91.3 8	81.48	91.82	91.48	-1.92	50.52
MAE	0.081	0.101	0.076	0.069	2755	0.219
MSE	0.007	0.017	0.007	0.008	8655302	0.061
RMSE	0.088	0.131	0.086	0.093	2941	0.247

From table 1 it can be seen that the worst results we obtained are of vector auto regression. The reason is that the vector auto regression is used for multiple time series predictions, but here we had only single time series.. The results of gated recurrent unit are also not good. Usually gated recurrent unit performs better than LSTM, but in our case this objective is not achieved. One of the reasons of gated recurrent unit model poor performance is that month group dataset has few samples, and as gated recurrent unit is a deep learning model, therefore, it requires large number of samples for training set. With increase in number of samples of training set the model is optimally generalized. Multi layer perceptron model results are good, but if we compare it to other models result, then multi layer perceptron is not a wise option to be used for prediction. Random forest results are optimum which can be deployed to predict stock prices for any random case. The convolution neural network

capture the features of given data very efficiently because this model is basically developed for images, and the data of images is always complex and huge. The LSTM results are also very much close to convolution 1-D model and as this architecture is especially developed for time series modeling, its results are very optimum.

5.6 Results by Day Grouping

Results obtained from trained models over day basis stock data are shown in table 2. From the results it is clear that random forest, CNN and LSTM models are efficient. Results of random forest are a bit better than LSTM and CNN. These three models are suitable for making predictions on real time data. The results are shown below:

5.6.1 Results of LSTM

The trained LSTM model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

Fig.58 LSTM Mean Absolute Error

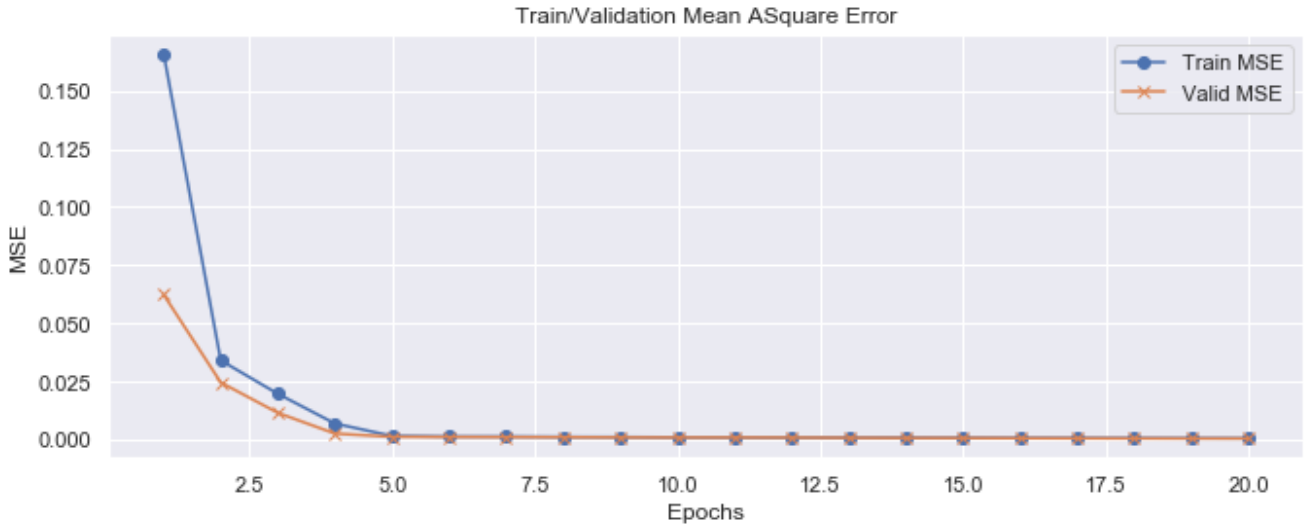


Fig.59 LSTM Mean Square Error

Below are the test set predicted and true values of stock with respect to time,

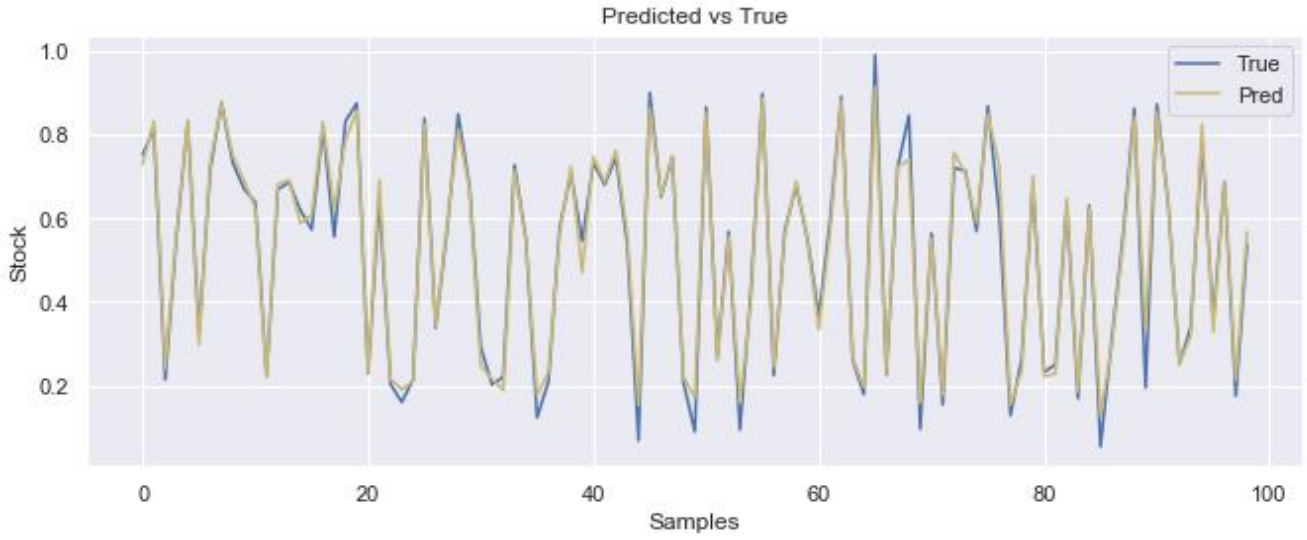


Fig.60 LSTM Predicted vs True Stock Values

5.6.2 Results of CNN

The trained CNN model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

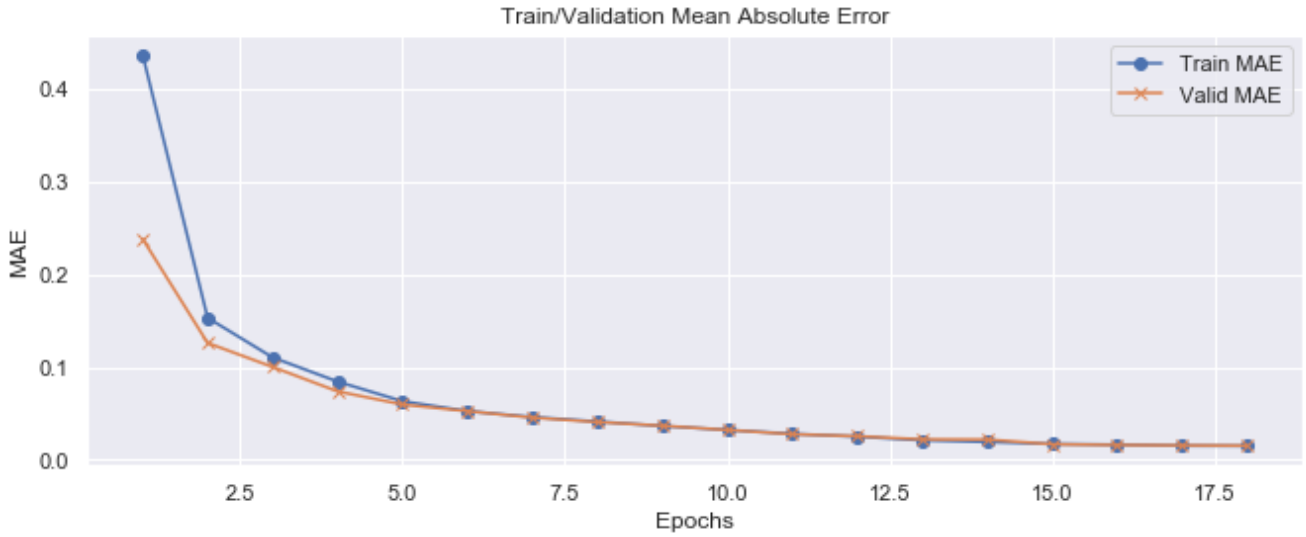


Fig.61 CNN Mean Absolute Error

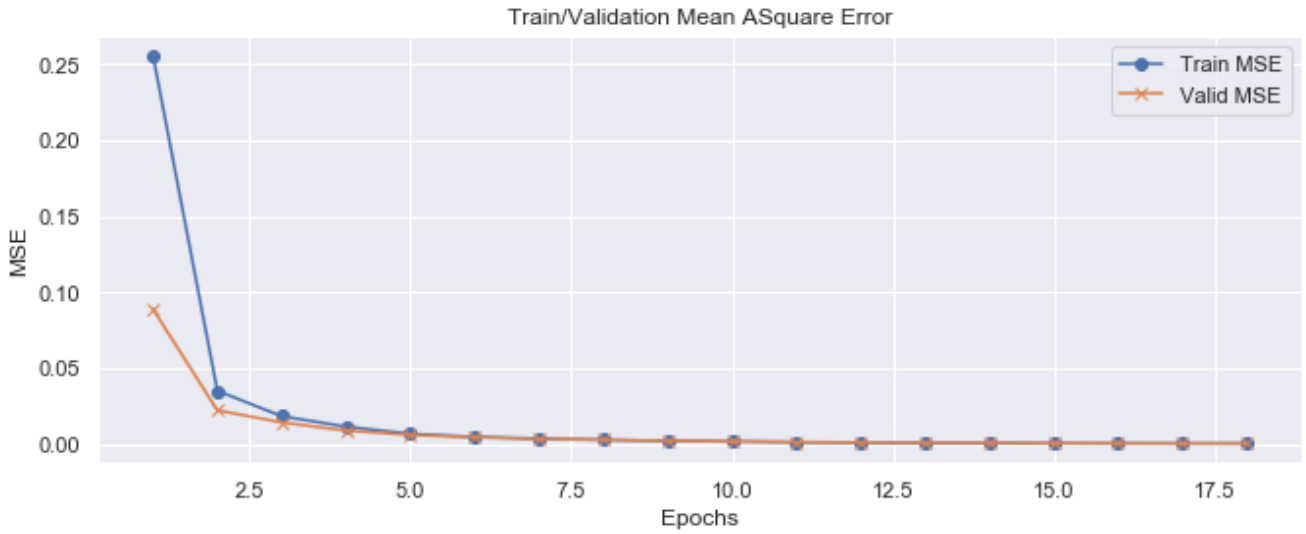


Fig.62 CNN Mean Square Error

Below is the test set predicted and true values of stock with respect to time,

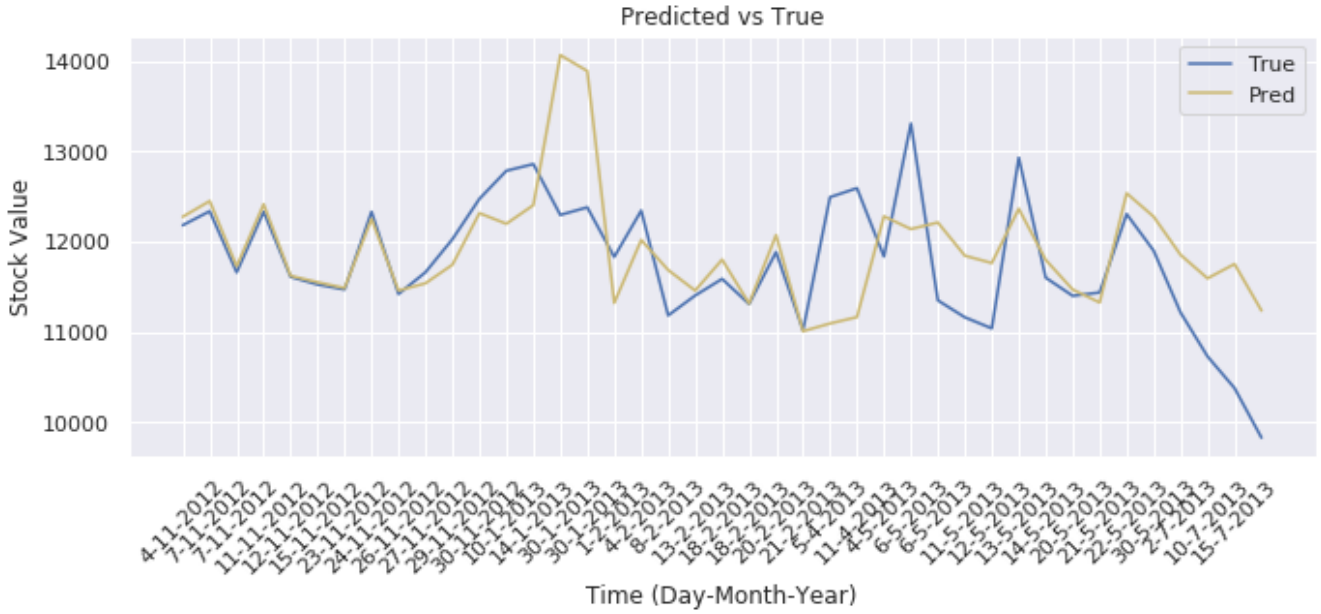


Fig.63 CNN Predicted vs True Stock Values

5.6.3 Results of GRU

The trained GRU model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

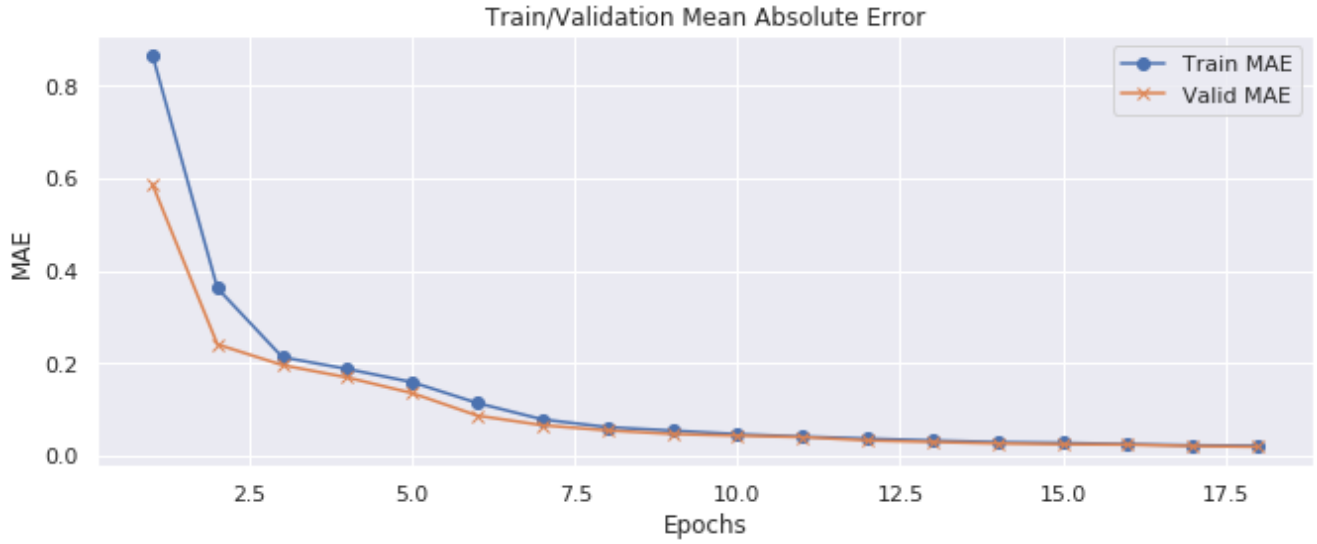


Fig.64 GRU Mean Absolute Error

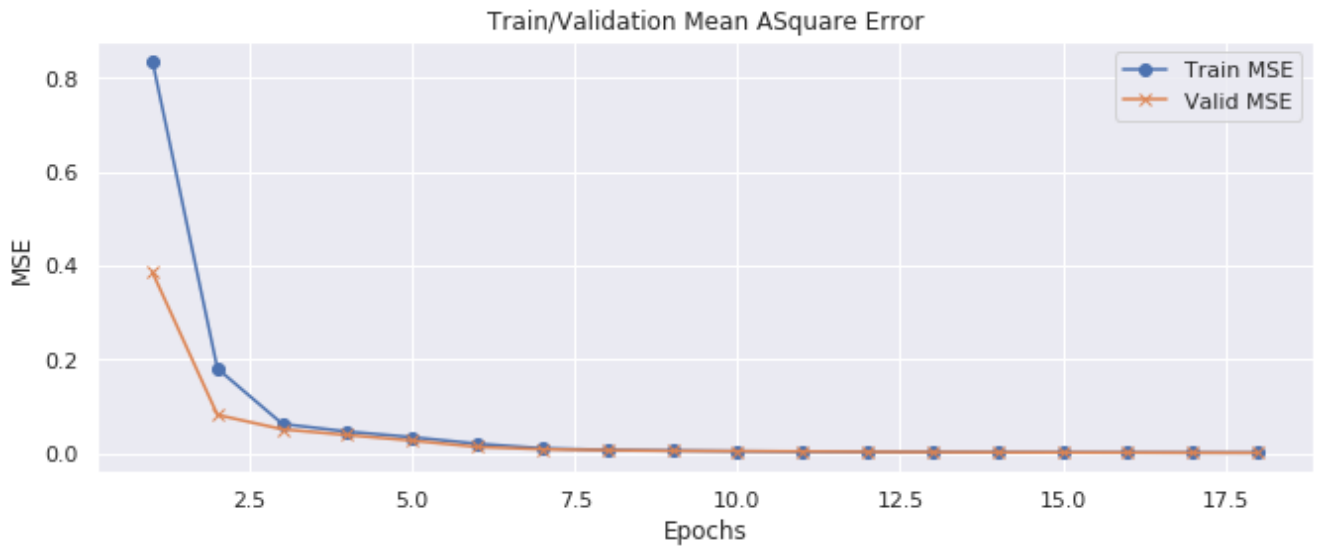


Fig.65 GRU Mean Square Error

Below are the test set predicted and true values of stock with respect to time,

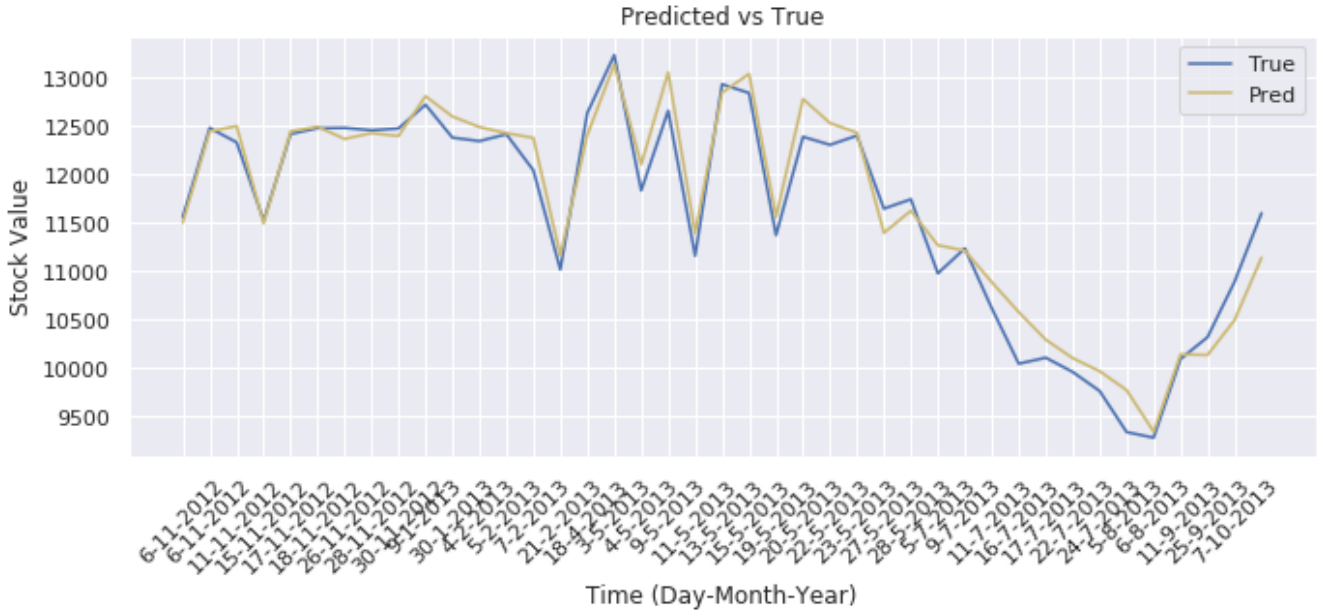


Fig.66 GRU Predicted vs True Stock Values

5.6.4 Results of MLP

The trained MLP model evaluation metric plots are shown in figure Errors in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

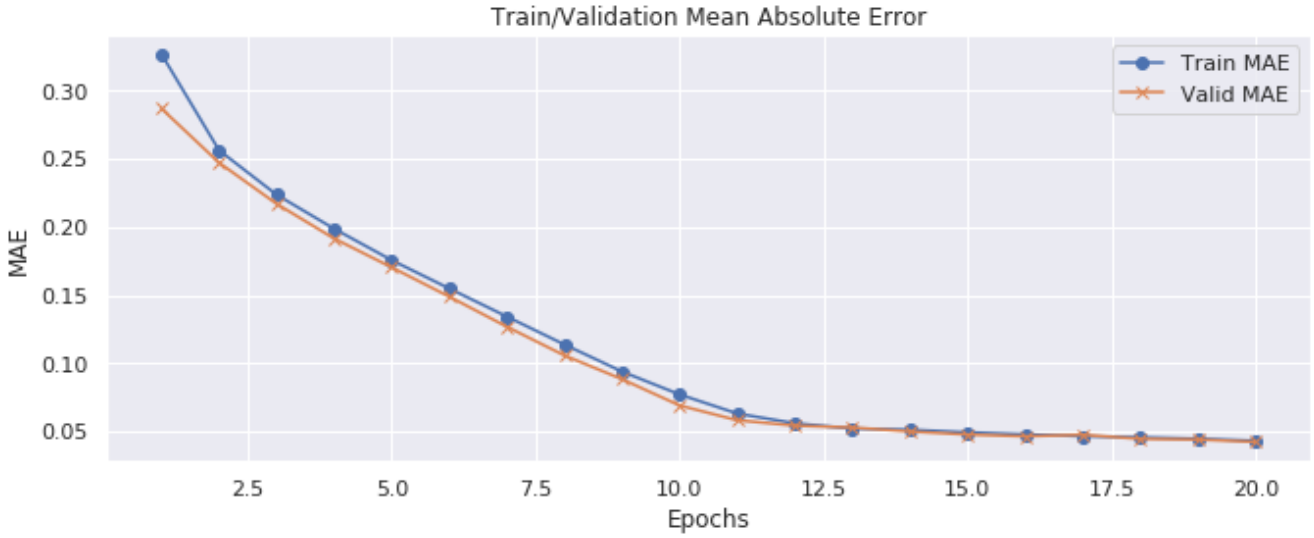


Fig.67 MLP Mean Absolute Error

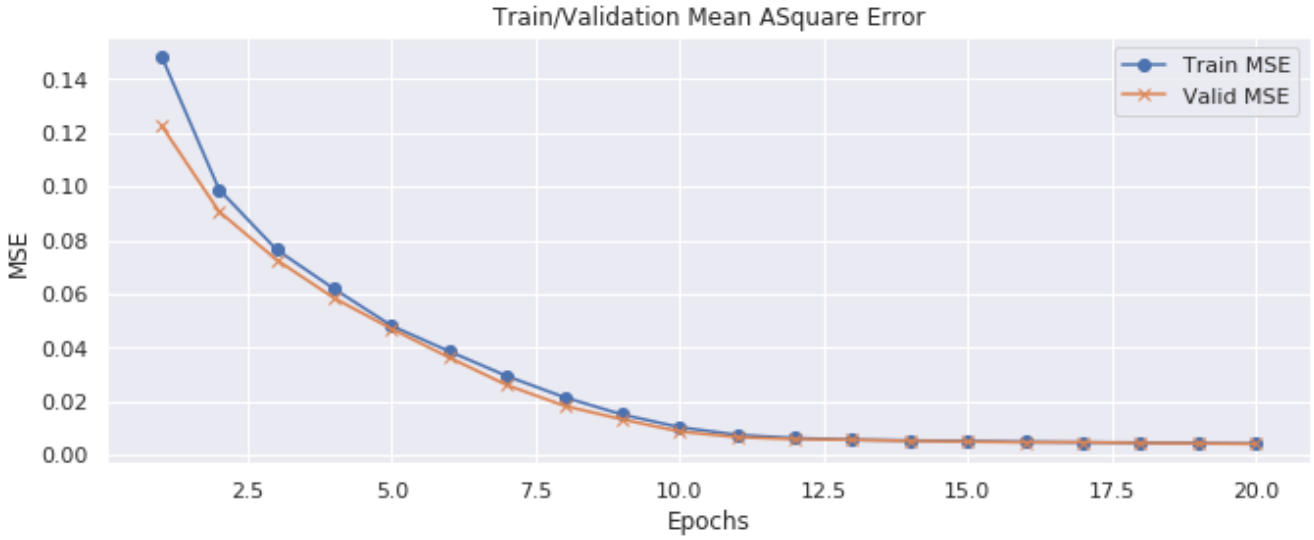


Fig.68 MLP Mean Square Error

Below are the test set predicted and true values of stock with respect to time,

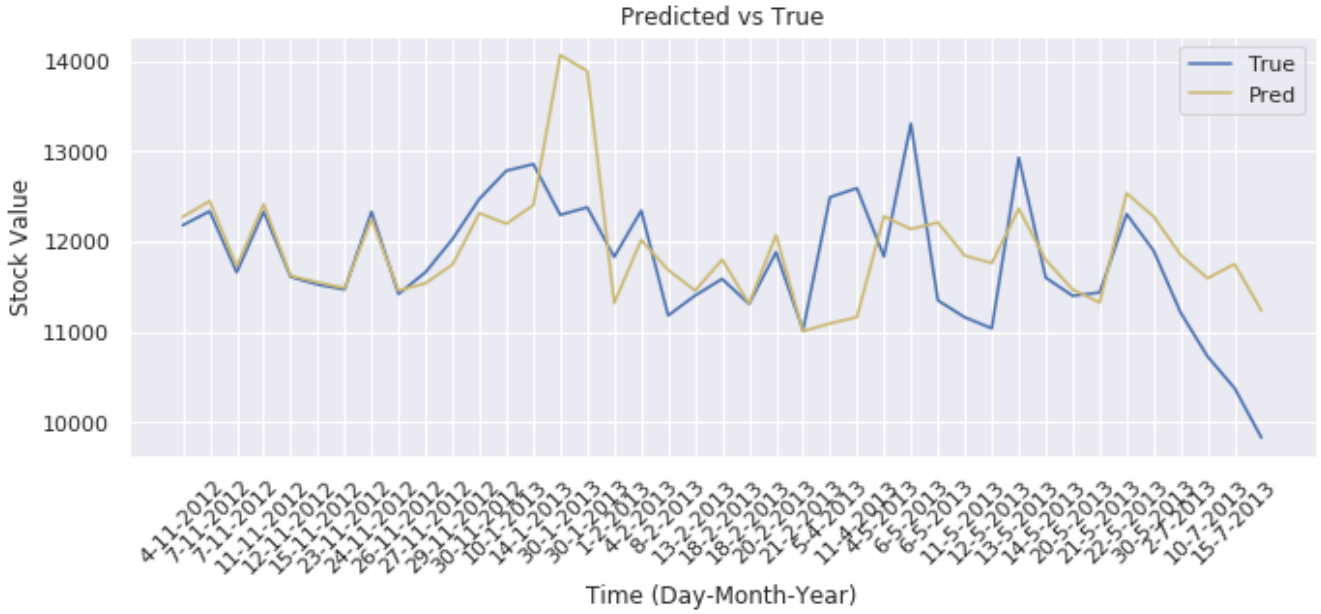


Fig.69 MLP Predicted vs True Stock Values

5.6.5 Results of Random Forest

Below are the test set predicted and true values of stock with respect to time,

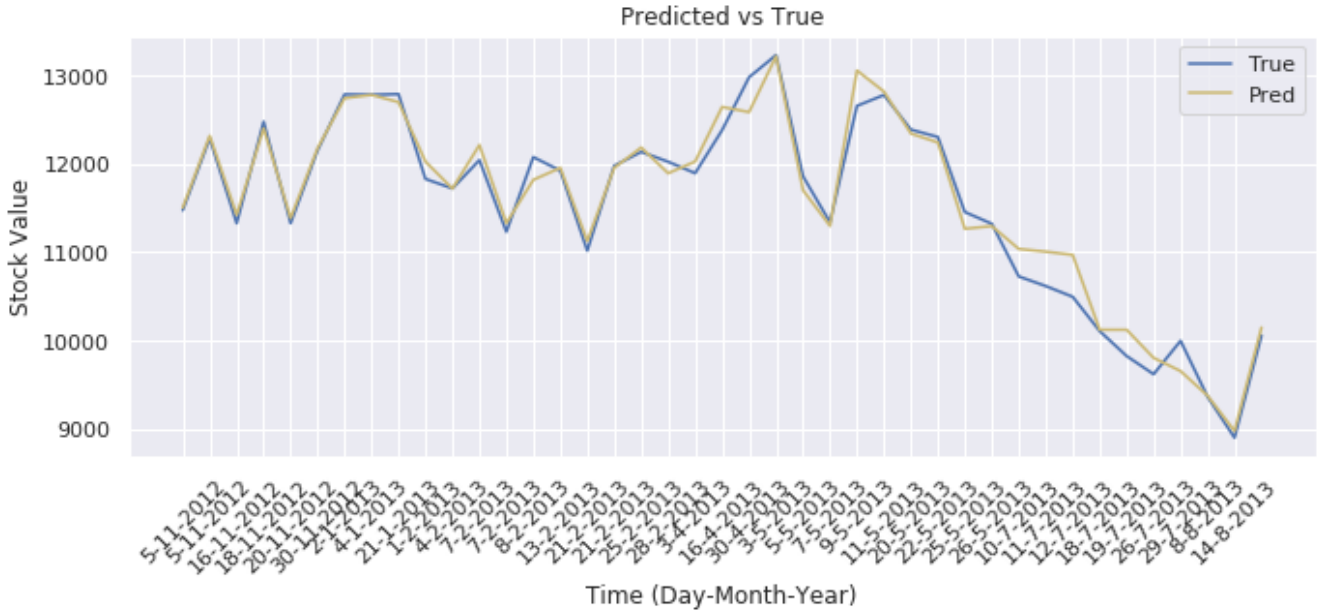


Fig.70 RF Predicted vs True Stock Values

5.6.6 Results of VAR

Below are the test set predicted and true values of stock with respect to time,

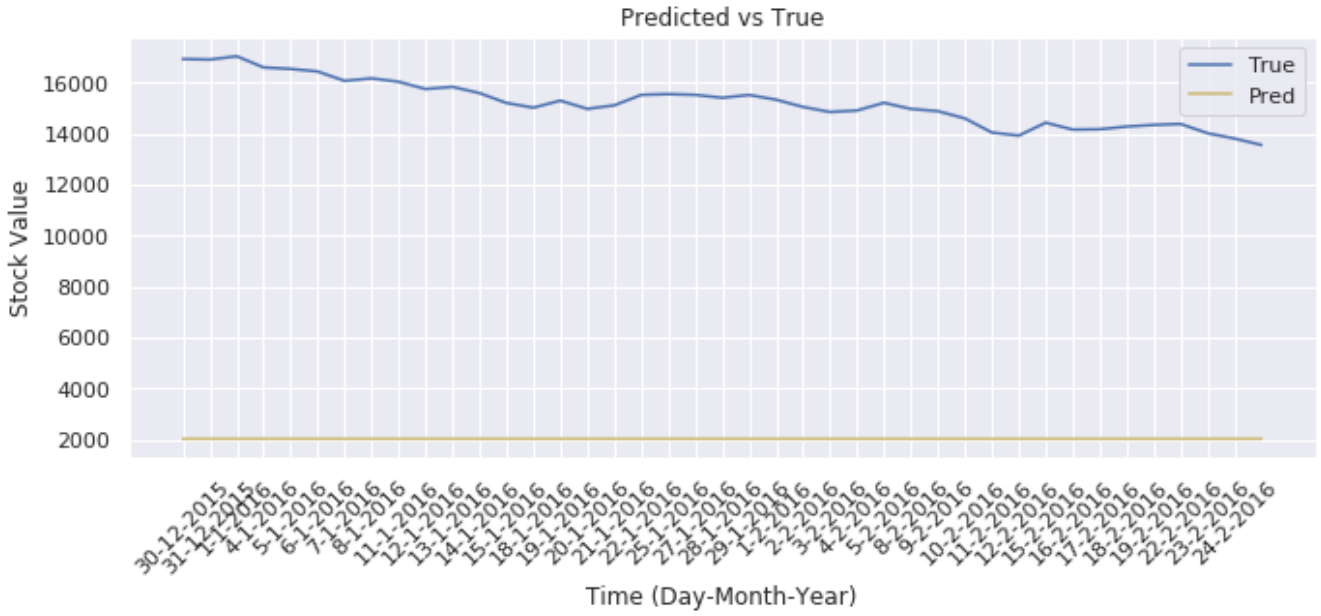


Fig.71 VAR Predicted vs True Stock Values

Table:3. Day Grouping Results

Day Group	**Random Forest**	**MLP**	**CNN**	**LSTM**	**VAR**	GRU
R2-Score (%)	99.42	93.91	98.91	99.21	-1.38	98.57
MAE	0.013	0.042	0.019	0.016	2461	0.02
MSE	0.001	0.004	0.001	0.001	7421587	0.001
RMSE	0.195	0.064	0.027	0.023	2724	0.029

By looking at table 2 it can be seen that the worst results we obtained for day grouping is also of vector auto regression. The reason is previously discussed that the vector auto regression is used for multiple time series predictions, but here we had only single time series. Due to this the vector auto regression model in underfitted. Multi-layer perceptron model results are good, but if we compare it to other models result, then multi-layer perceptron is not a good choice to be used for prediction. The results of gated recurrent unit are optimized. Previously the number of samples in train test were small due to which the

model did not performed up to the expectations. Now the train set size is increased and the generalization of the model is almost similar to LSTM model. Random forest results are highly optimum which can be deployed to predict stock prices for any random case. The convolution neural network capture the features of given data very efficiently because this model is basically developed for images, and the data of images is always complex and huge. The LSTM results are also very much close to convolution model and as this architecture is especially developed for time series modeling, its results are very optimum. The best model having maximum generalization is of random forest.

5.7 Results of Hour Grouping

Results of hour based stocks data is shown in table 3. MLP, LSTM and CNN models results are better. However, random forest model generalization is also effective and can be deployed for real time predictions of stocks data on hourly basis. The results of hour based stock models is shown next:

5.7.1 Results of LSTM

LSTM model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

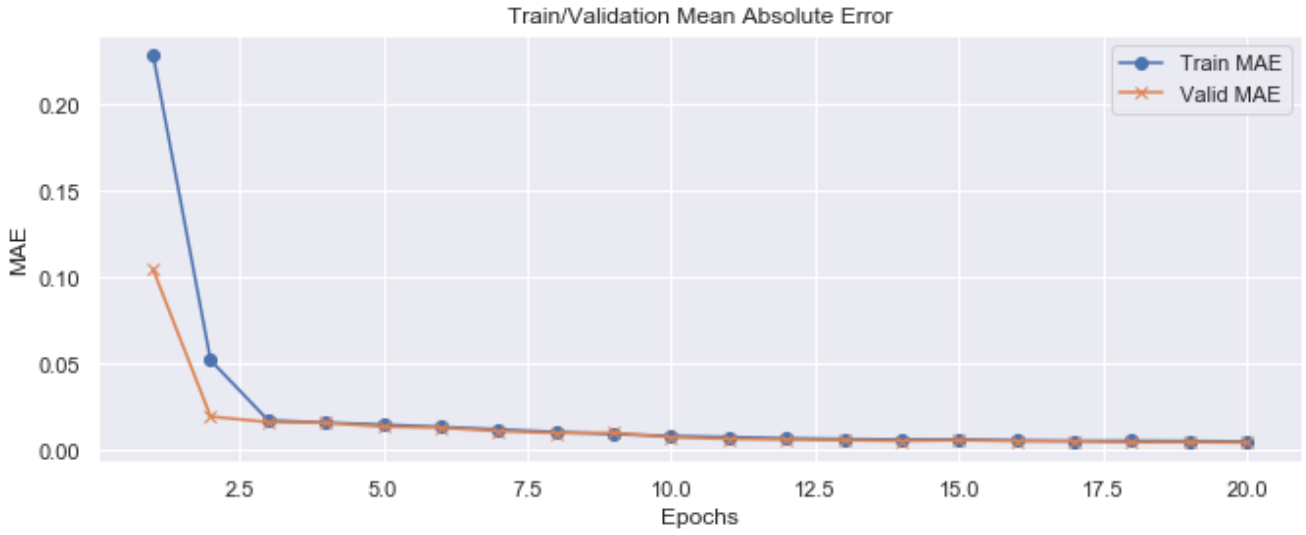


Fig.72 LSTM Mean Absolute Error

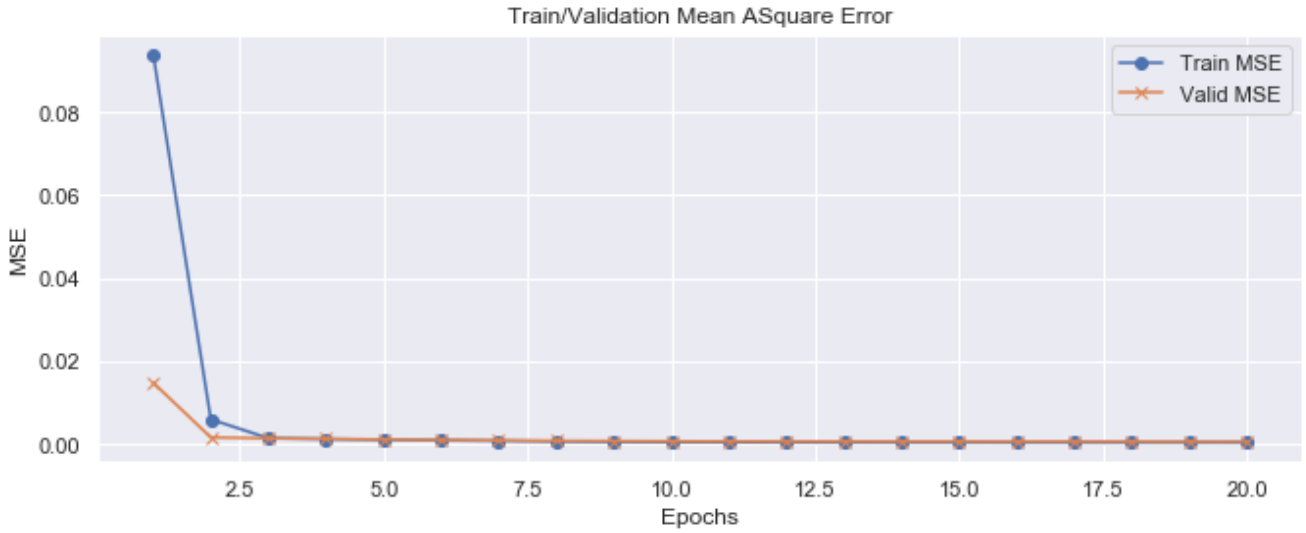


Fig.72 LSTM Mean Square Error

Below are the test set predicted and true values of stock with respect to time,

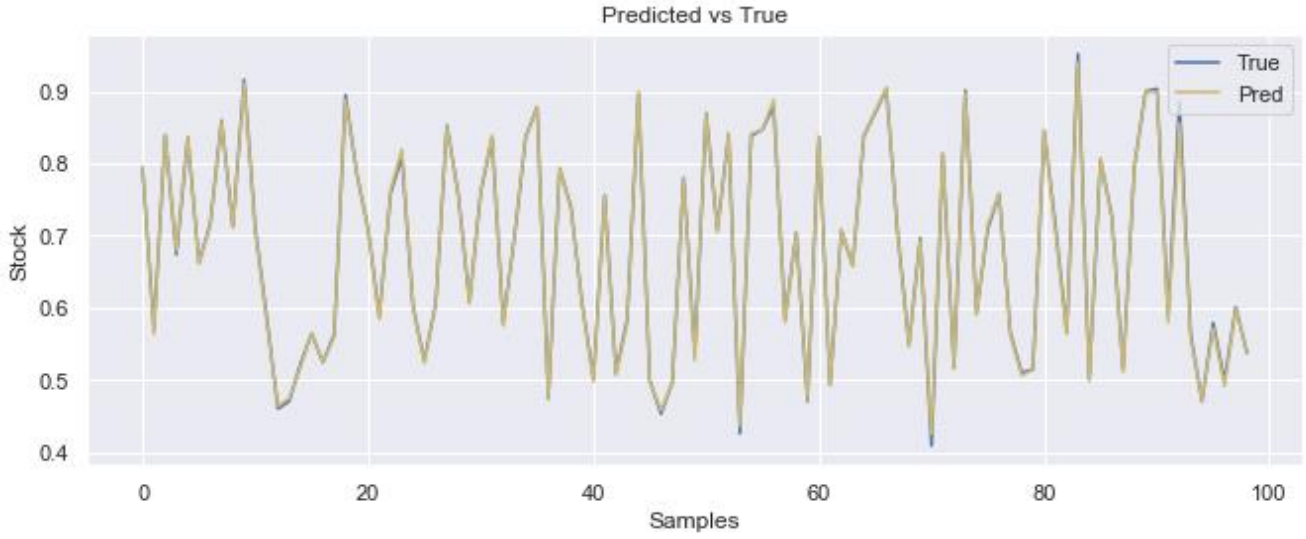


Fig.73 LSTM Predicted vs True Stock Values

5.7.2 Results of CNN

CNN model evaluation metric plots are shown in figure Error values in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

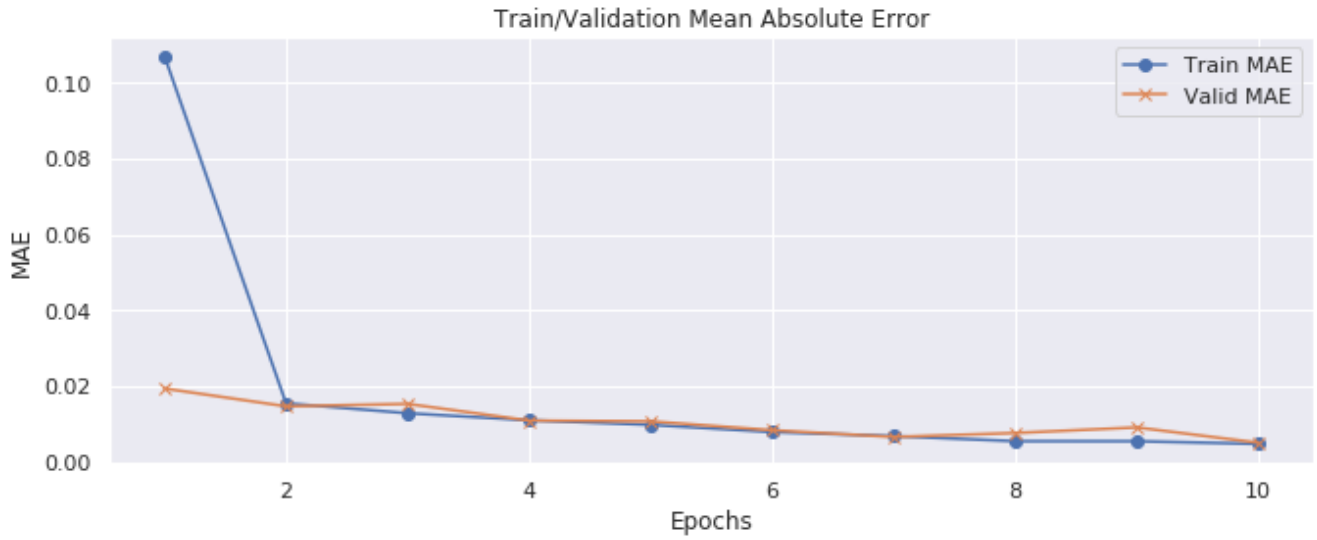


Fig.74 CNN Mean Absolute Error

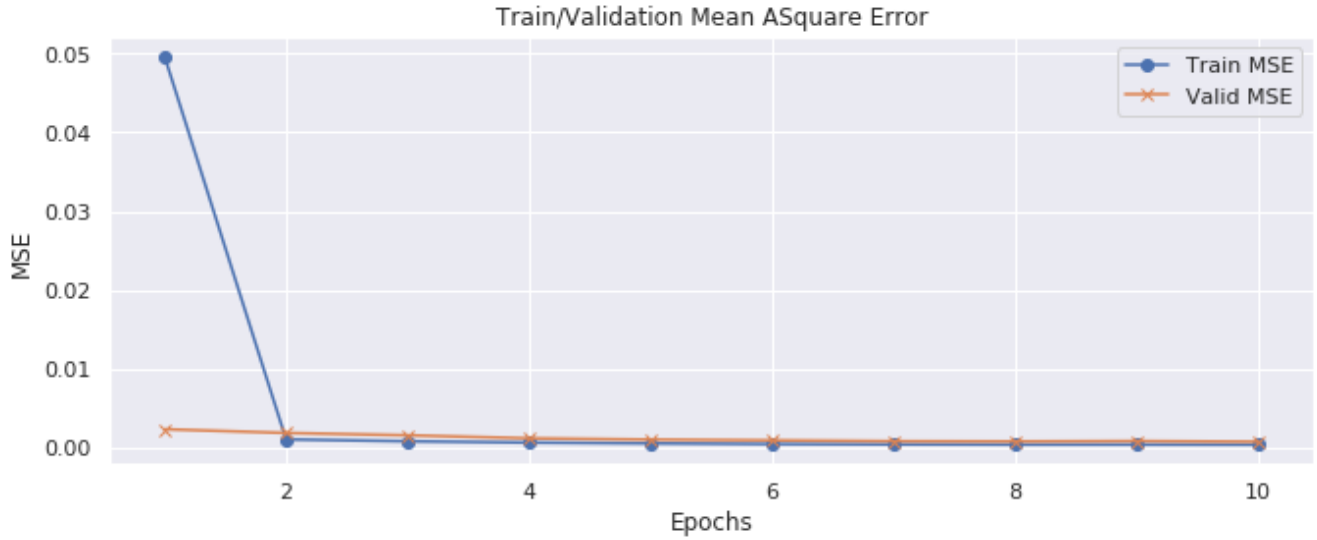


Fig.75 CNN Mean Square Error

Below the test set predicted and true values of stock with respect to time.

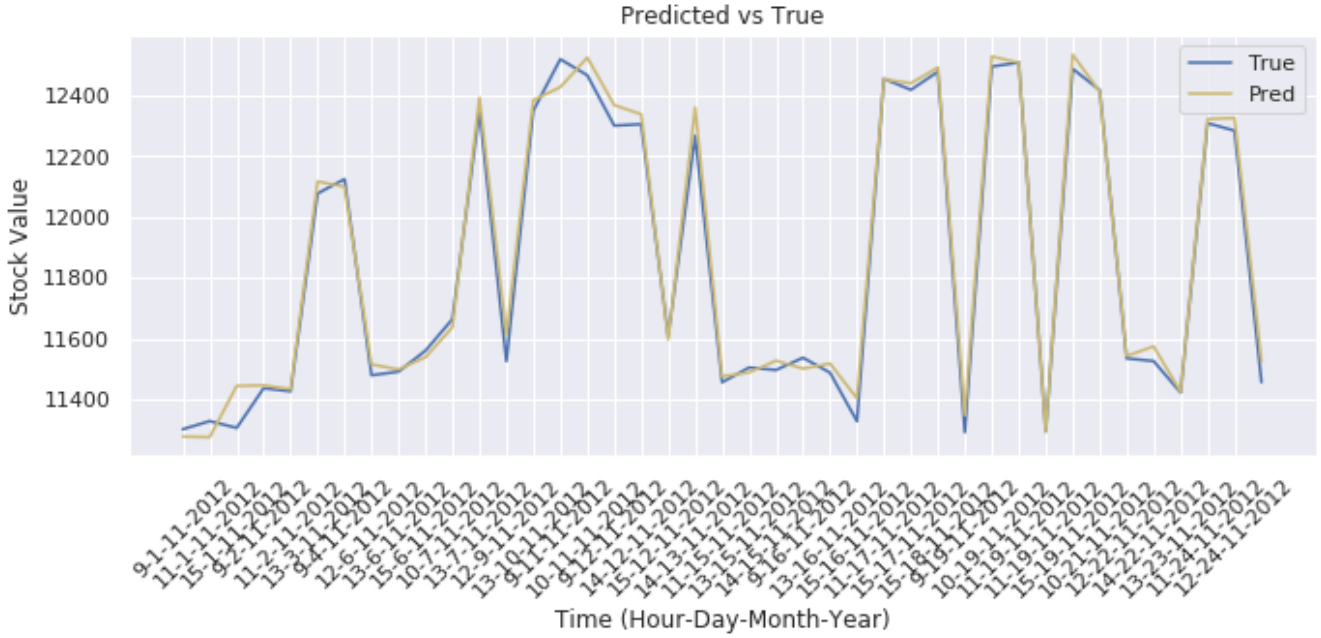


Fig.76 CNN Predicted vs True Stock Values

5.7.3 Results of GRU

The trained GRU model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

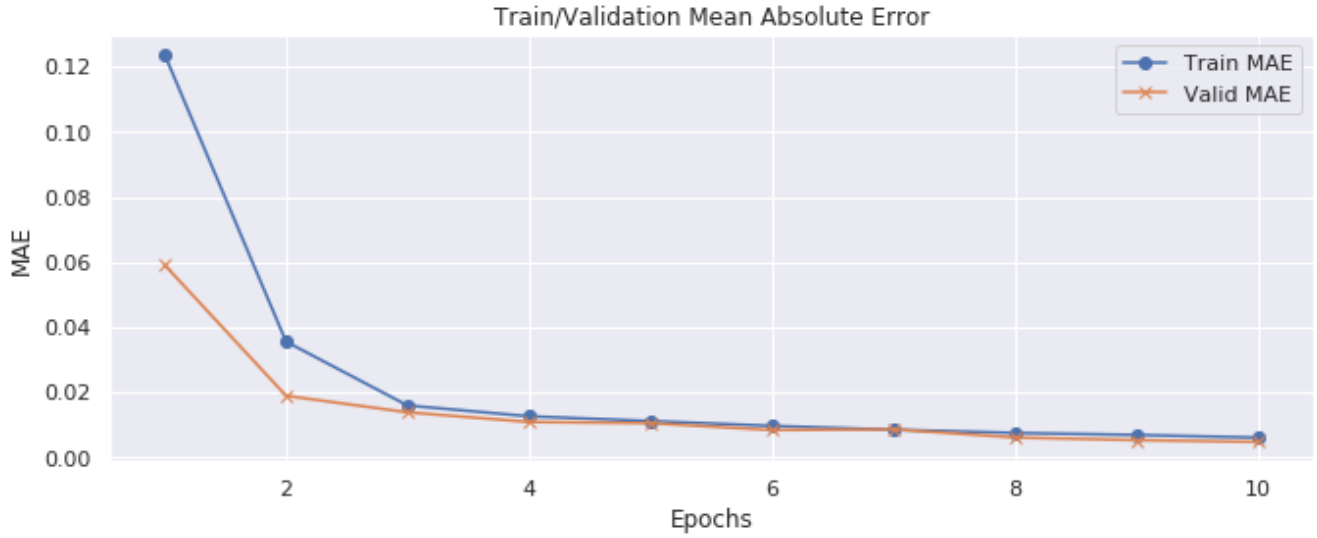


Fig.77 GRU Mean Absolute Error

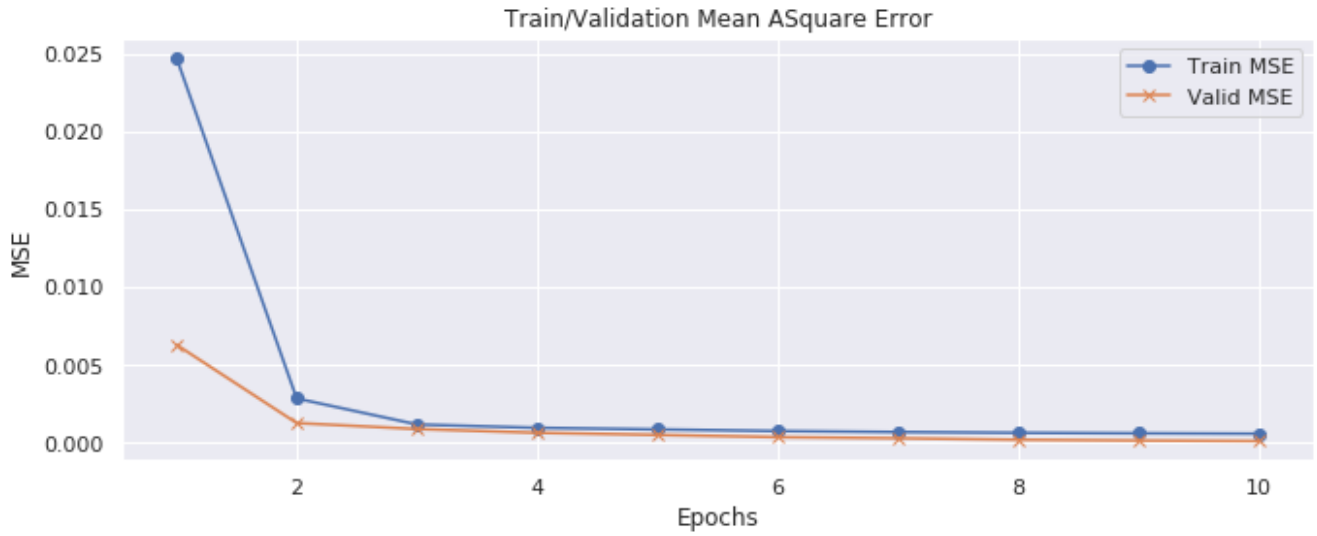


Fig.78 GRU Mean Square Error

Below are the test set predicted and true values of stock with respect to time.

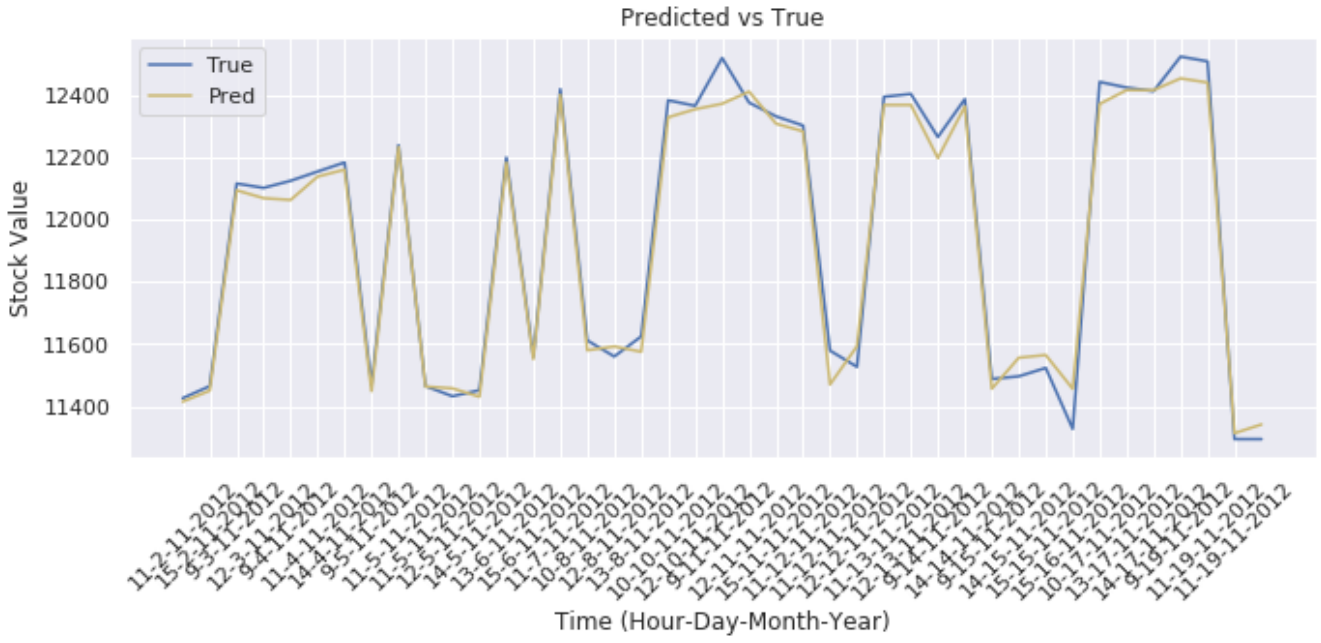


Fig.79 GRU Predicted vs True Stock Values

5.7.4 Results of MLP

The trained MLP model evaluation metric plots are shown in figure Error values in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

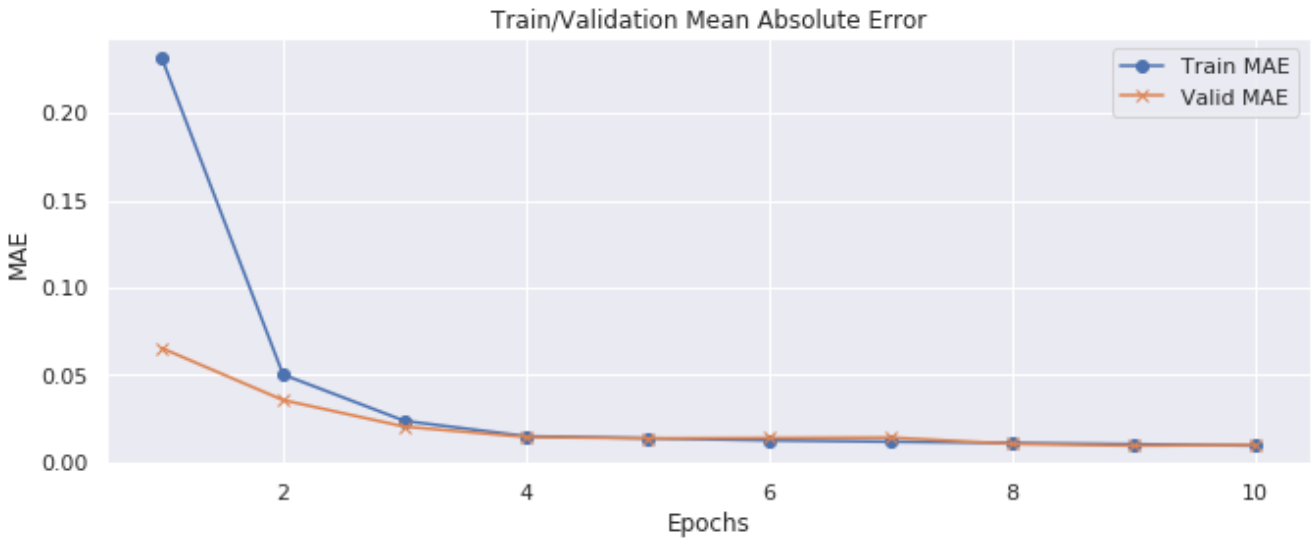


Fig.80 MLP Mean Absolute Error

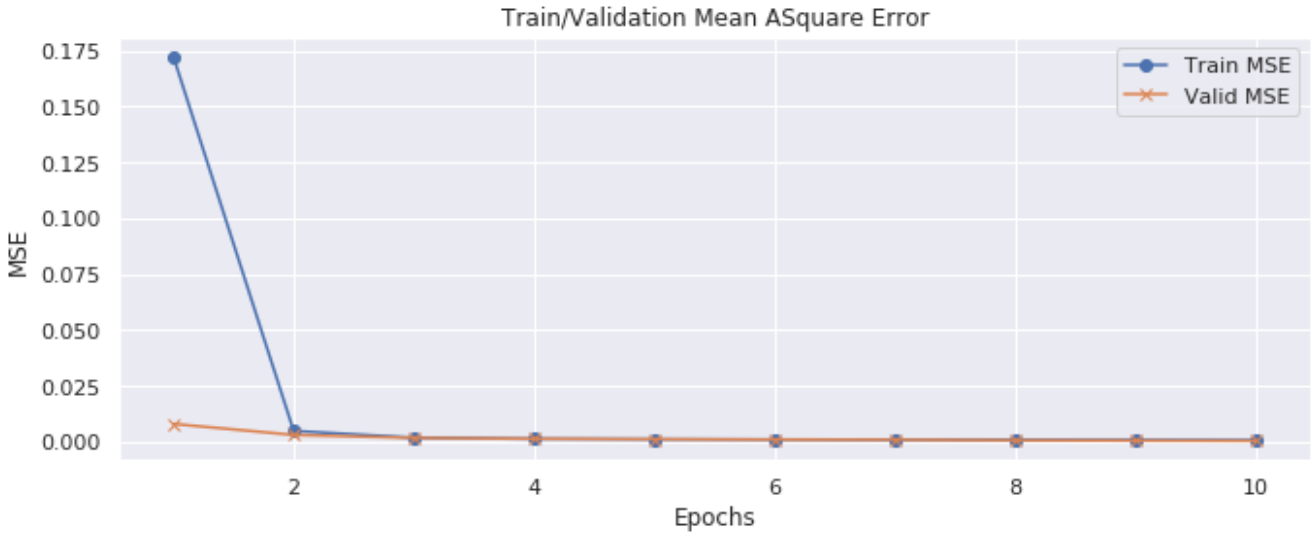


Fig.81 MLP Mean Square Error

Below are the test set predicted and true values of stock with respect to time.

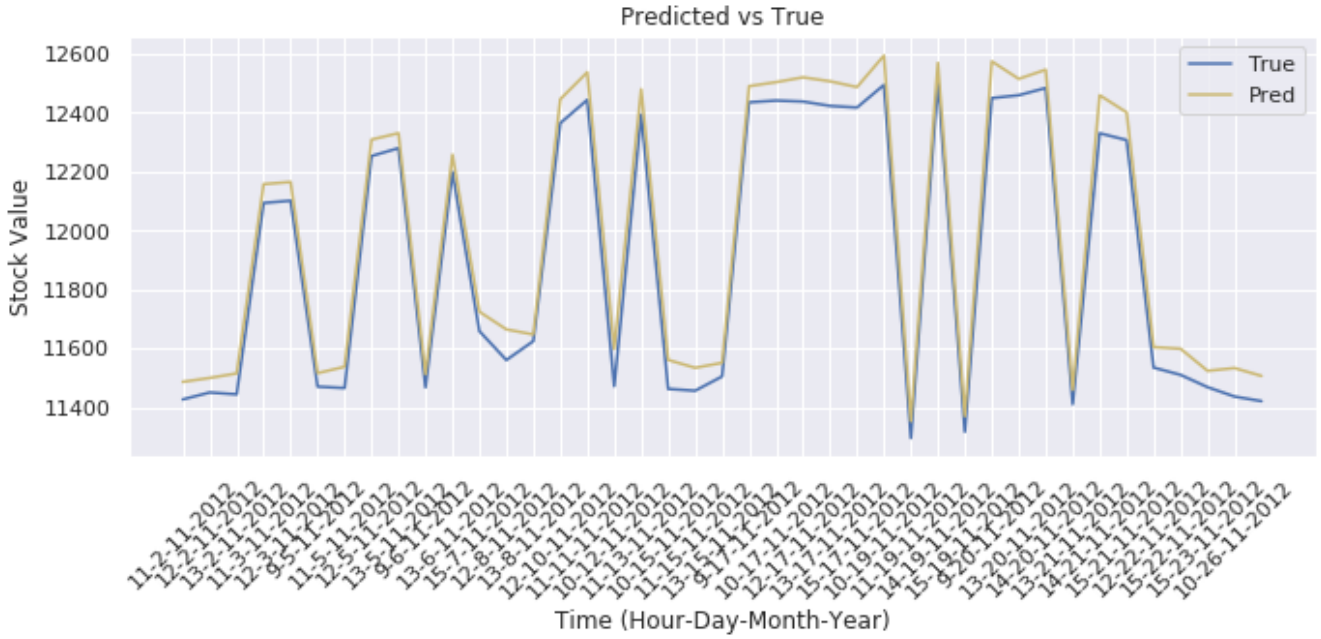


Fig.82 MLP Predicted vs True Stock Values

5.7.5 Results of Random Forest

Below are the test set predicted and true values of stock with respect to time.

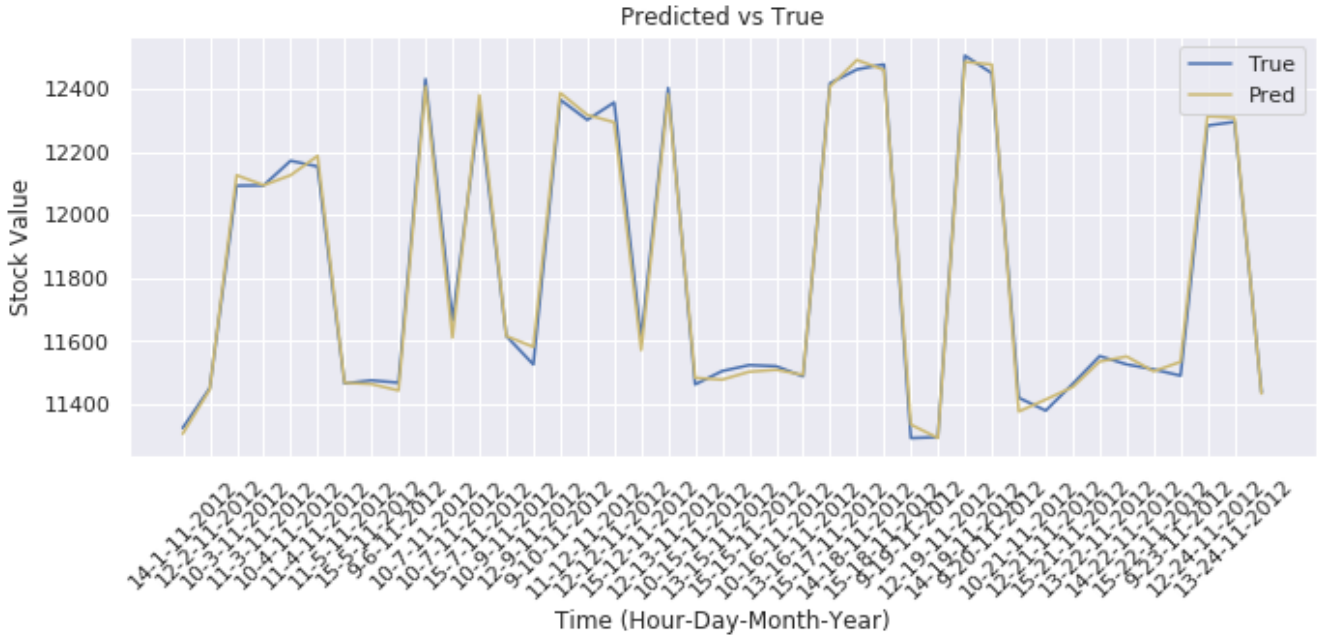


Fig.83 RF Predicted vs True Stock Values

5.7.6 Results of VAR

Below are the test set predicted and true values of stock with respect to time.

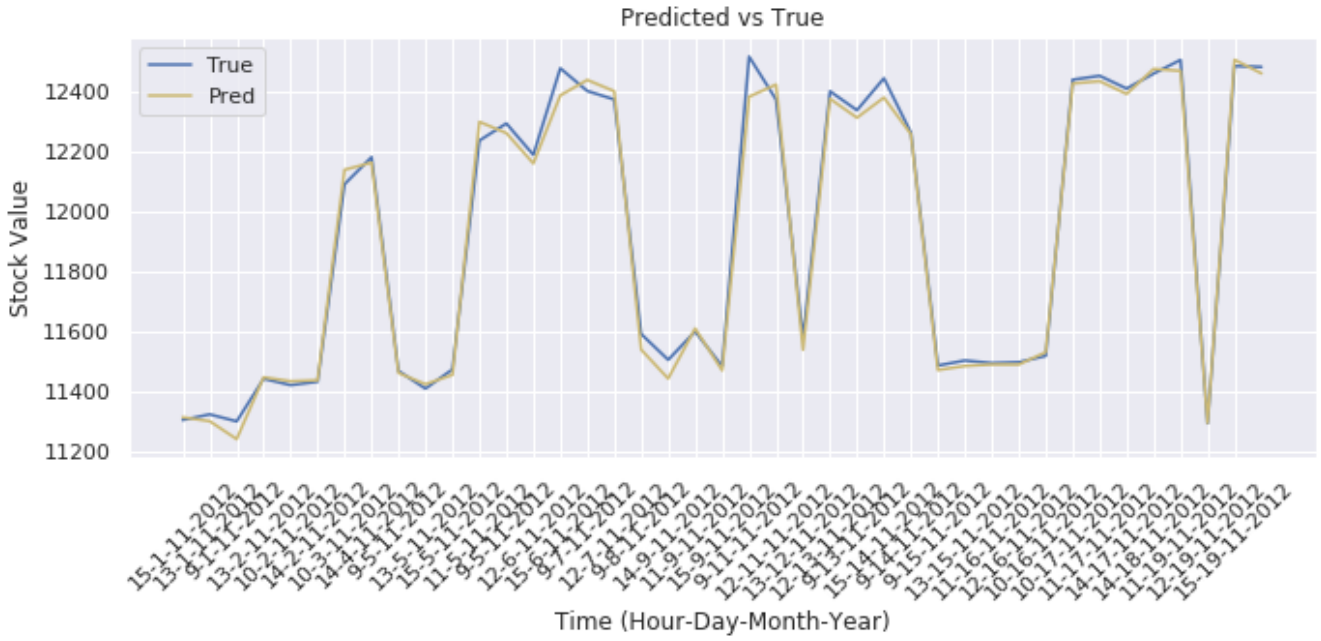


Fig.84 VAR Predicted vs True Stock Values

Table:4. Hour Grouping Results

Hour Group	Random Forest	MLP	CNN	LSTM	VAR	GRU
R2-Score (%)	95.94	97.85	97.13	97.72	-0.28	99.60
MAE	0.004	0.009	0.005	0.004	1686	0.004
MSE	0.001	0.001	0.001	0.001	3999797	0.001
RMSE	0.033	0.023	0.027	0.024	1999	0.011

By looking at table 3 it can be seen that the worst results for hour grouping is also of vector auto regression. Due to single time series in the given data set there is no chance of avoiding underfitting for the trained model. Due to this the vector auto regression model in underfitted. Random Forest model results are good, but if we compare it to other models

result, then random forest model is not a good choice to be used for prediction. The convolution neural network has better R2_score and errors are optimally reduced. The LSTM results are also very much close to convolution model and as this architecture is especially developed for time series modeling, its results are very optimum. The results of multi-layer perceptron are very much close to convolution and LSTM model. Gated recurrent unit model has outperformed other models. The number of samples in train set are more than month and day aggregation groupings, and it was previously discussed that gated recurrent unit was developed as an improvement to LSTM architecture, and here we can see that this objective is fulfilled. The best generalized model for real time hourly stock values prediction is of GRU.

5.8 Household Electricity Consumption Results

The household power consumption trained model results are shown in the table 4. The results of CNN are better than other models, but MLP, CNN, random forest and LSTM can also be deployed for predicting global active power in real time.

5.8.1 Results of LSTM

The trained LSTM model evaluation metric plots are shown in figure Error in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

Fig.85 LSTM Mean Absolute Error

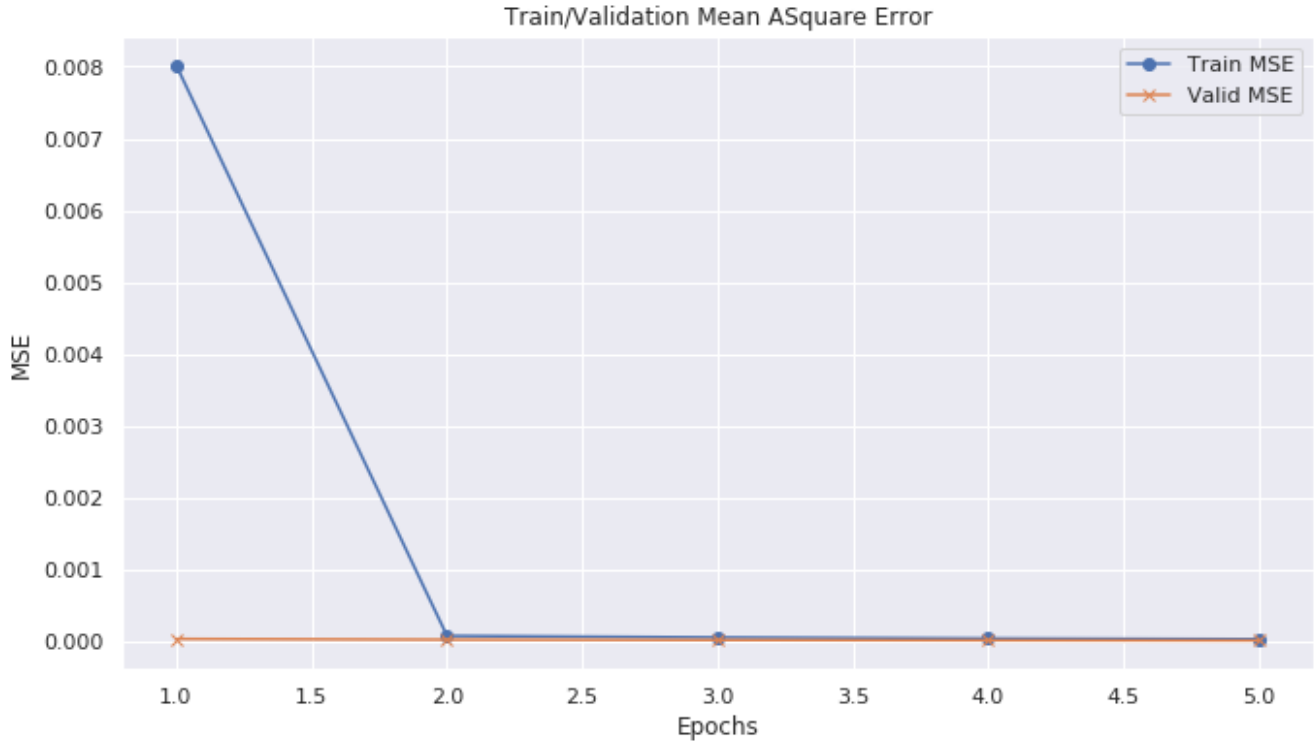


Fig.86 LSTM Mean Square Error

Below is the test set predicted and true values of stock with respect to time.

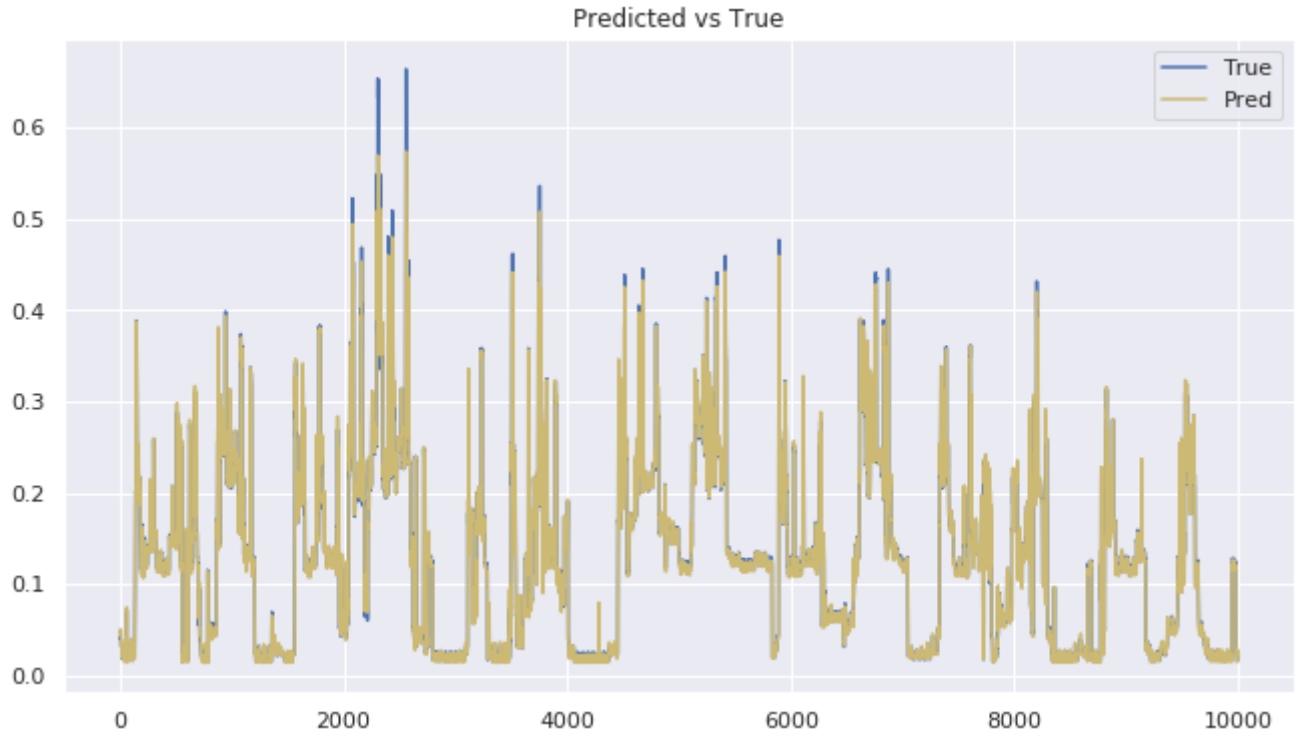


Fig.87 LSTM Predicted vs True Stock Values

5.8.2 Results of CNN

The trained CNN model evaluation metric plots are shown in figure Error values in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

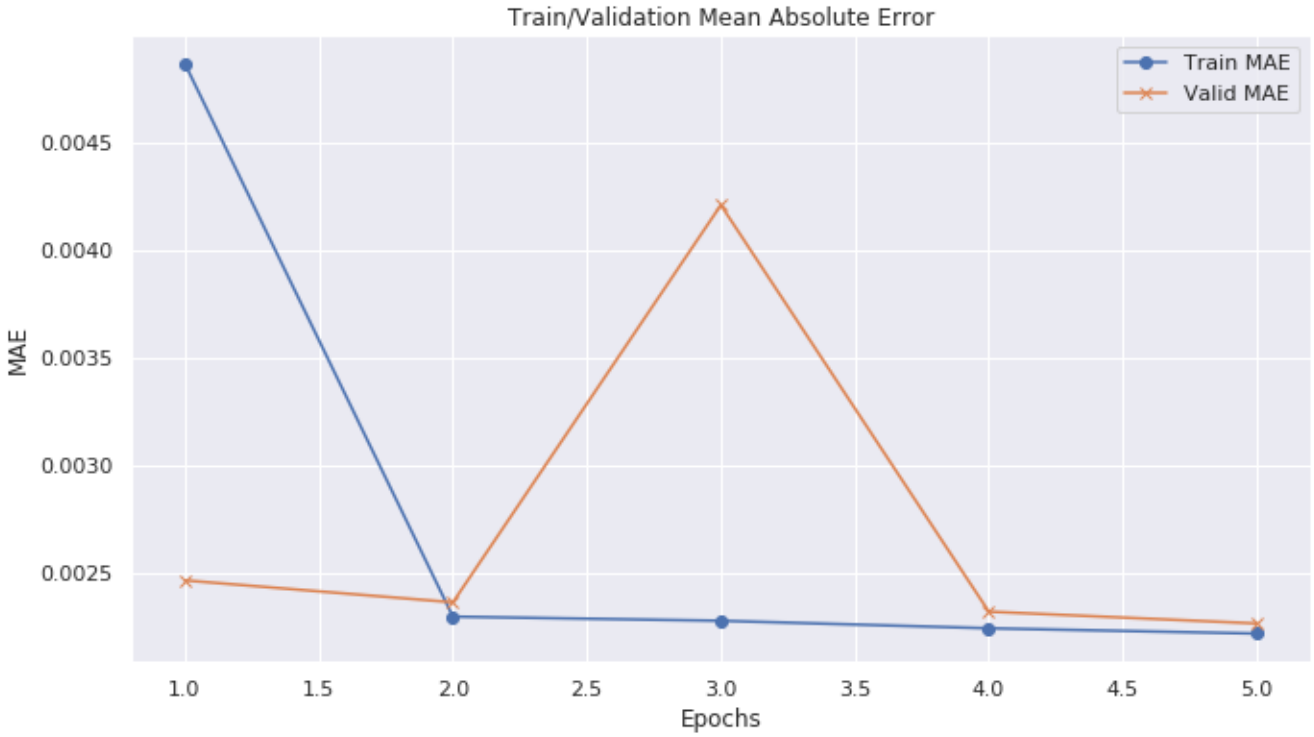


Fig.88 CNN Mean Absolute Error

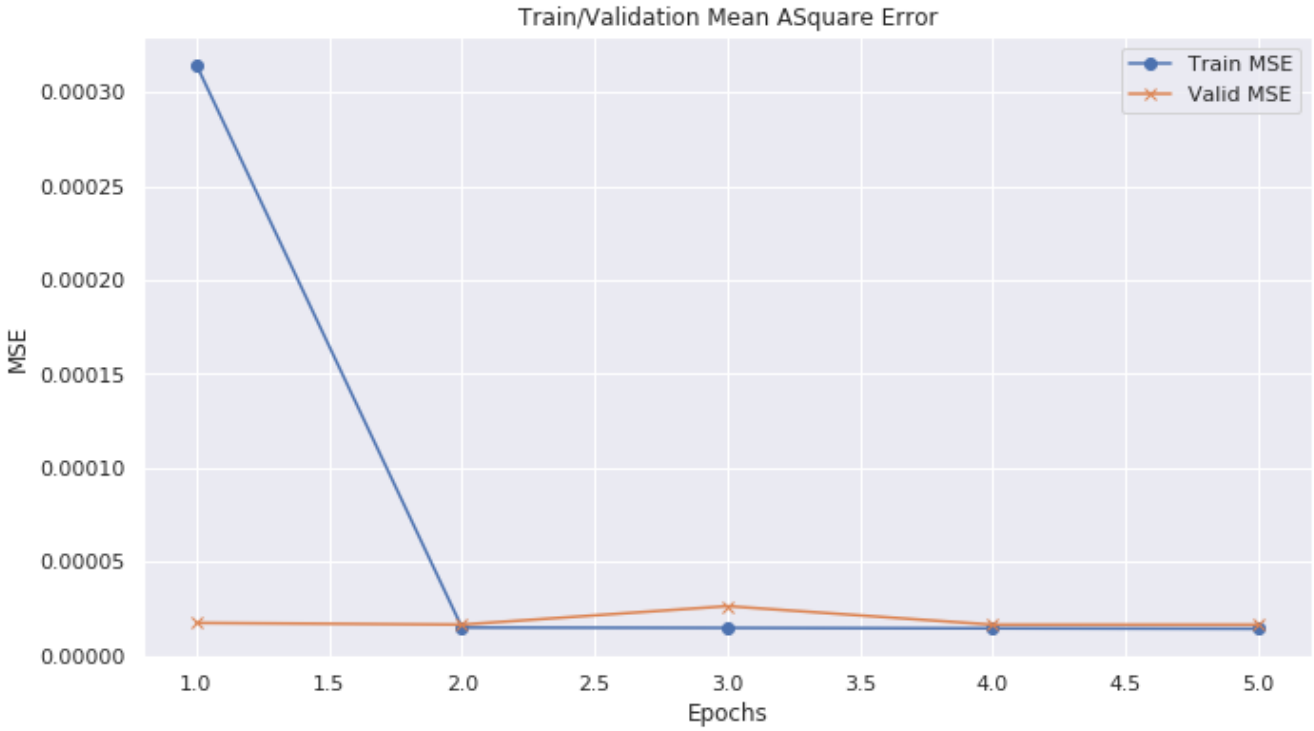


Fig.89 CNN Mean Square Error

Below are the test set predicted and true values of stock with respect to time.

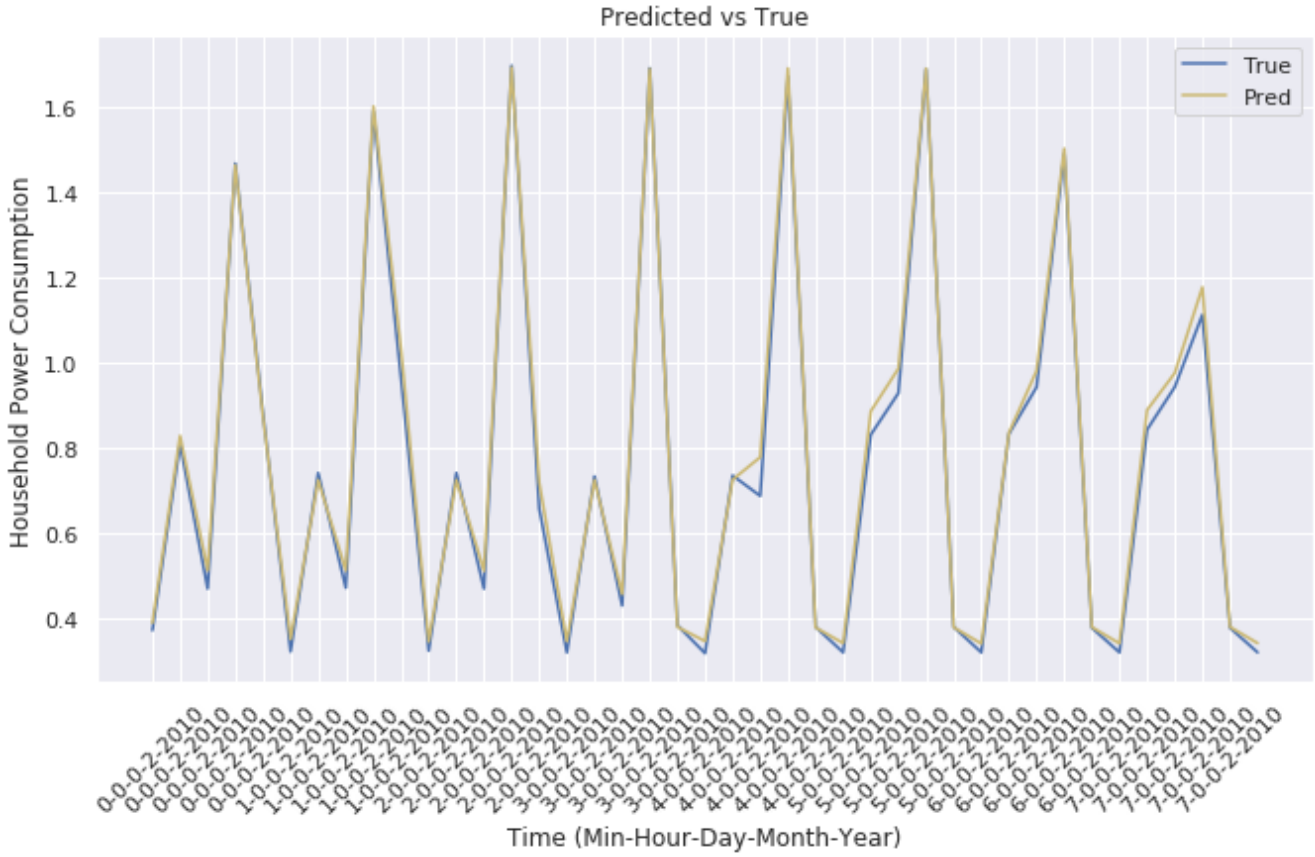


Fig.90 CNN Predicted vs True Stock Values

5.8.3 Results of GRU

The trained GRU model evaluation metric plots are shown in figure Error values in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

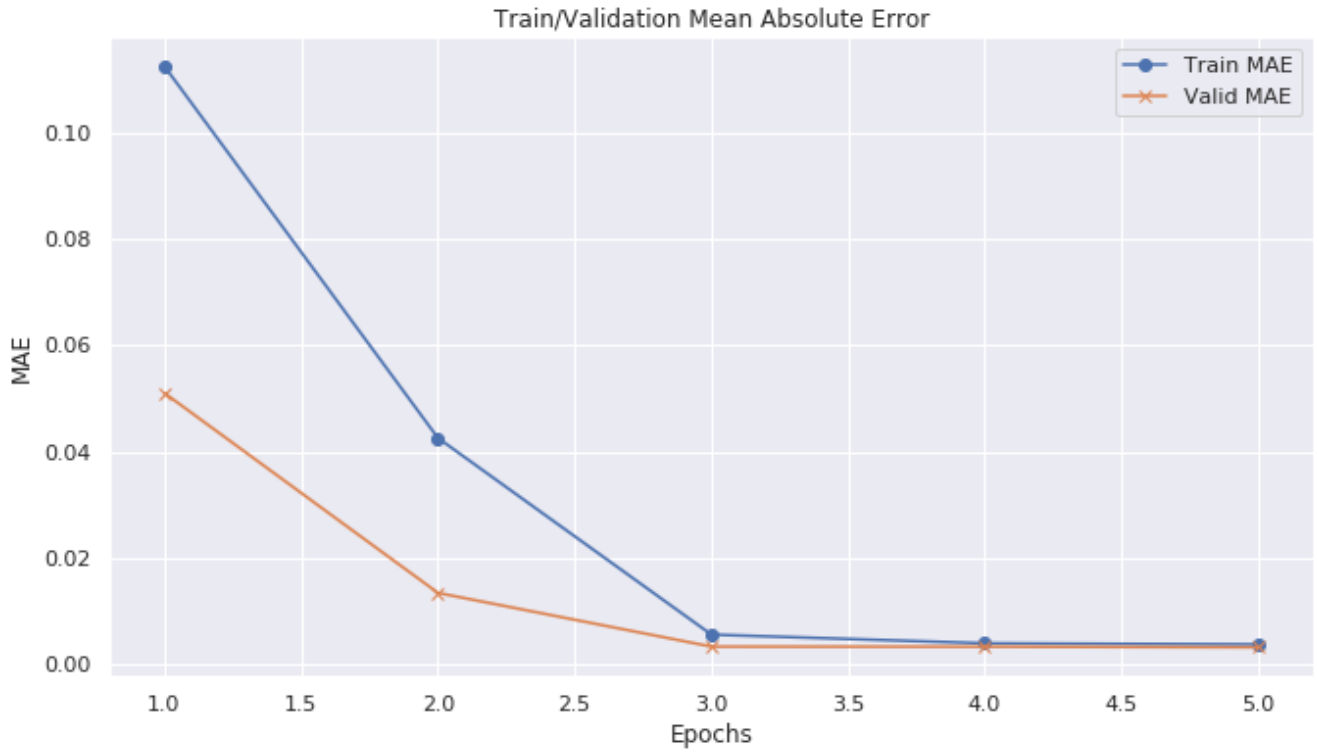


Fig.91 GRU Mean Absolute Error

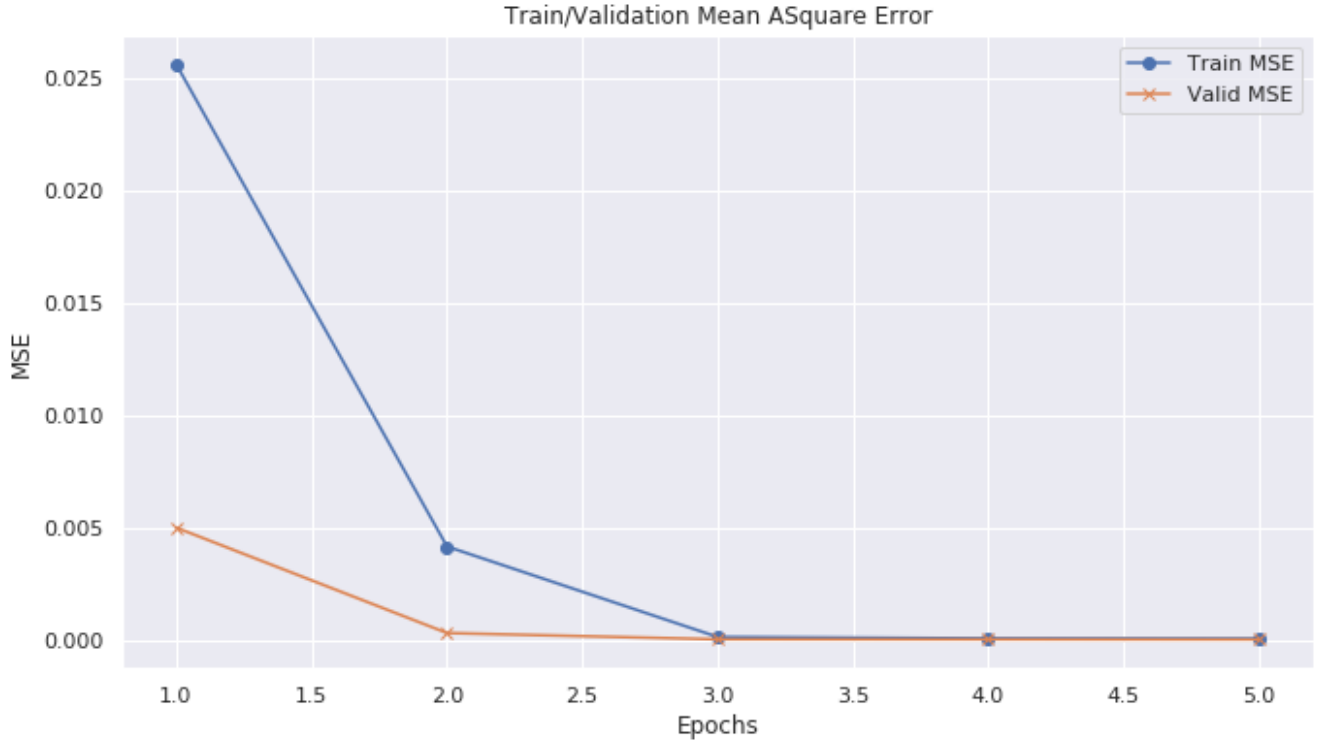


Fig.92 GRU Mean Square Error

Below are the test set predicted and true values of stock with respect to time.

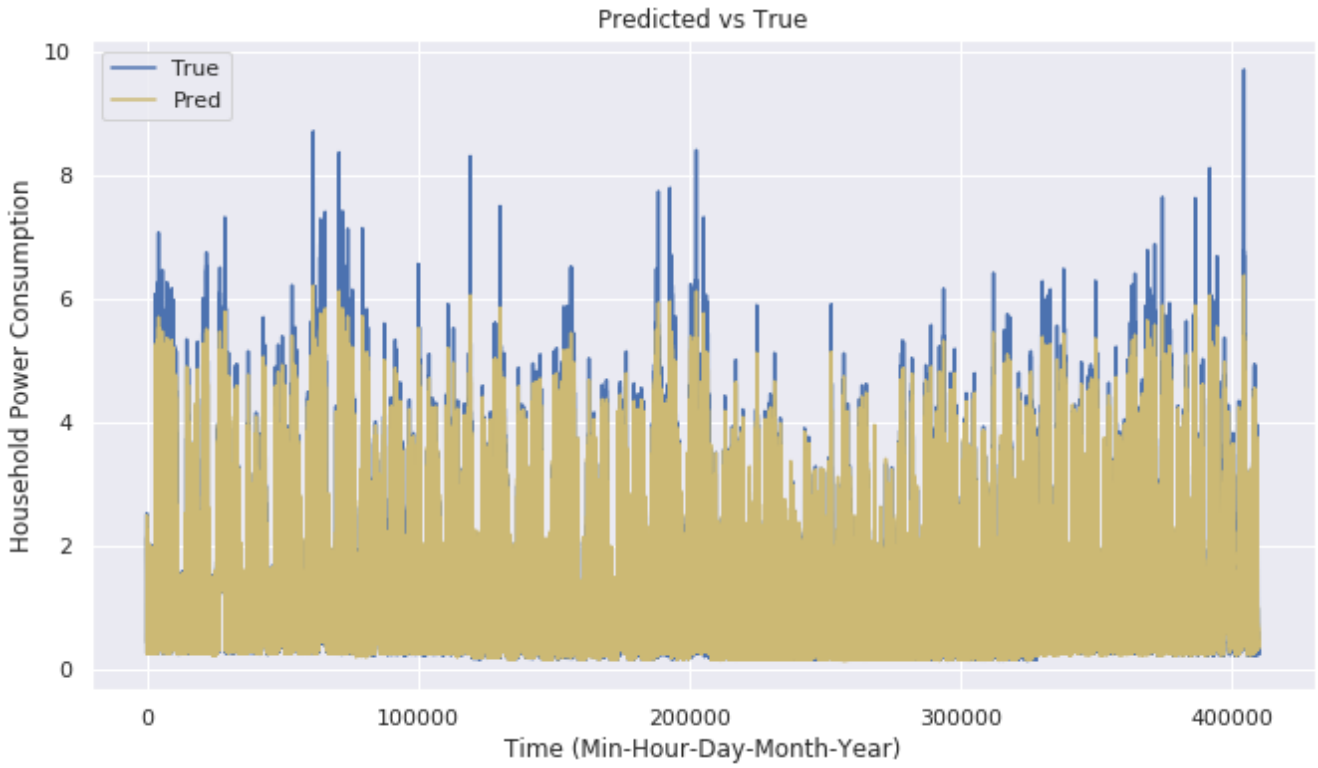


Fig.93 GRU Predicted vs True Stock Values

5.8.4 Results of MLP

The trained MLP model evaluation metric plots are shown in figure Error values in train and test predictions in each epoch are plotted. The plots for mean absolute error and mean square error are shown below:

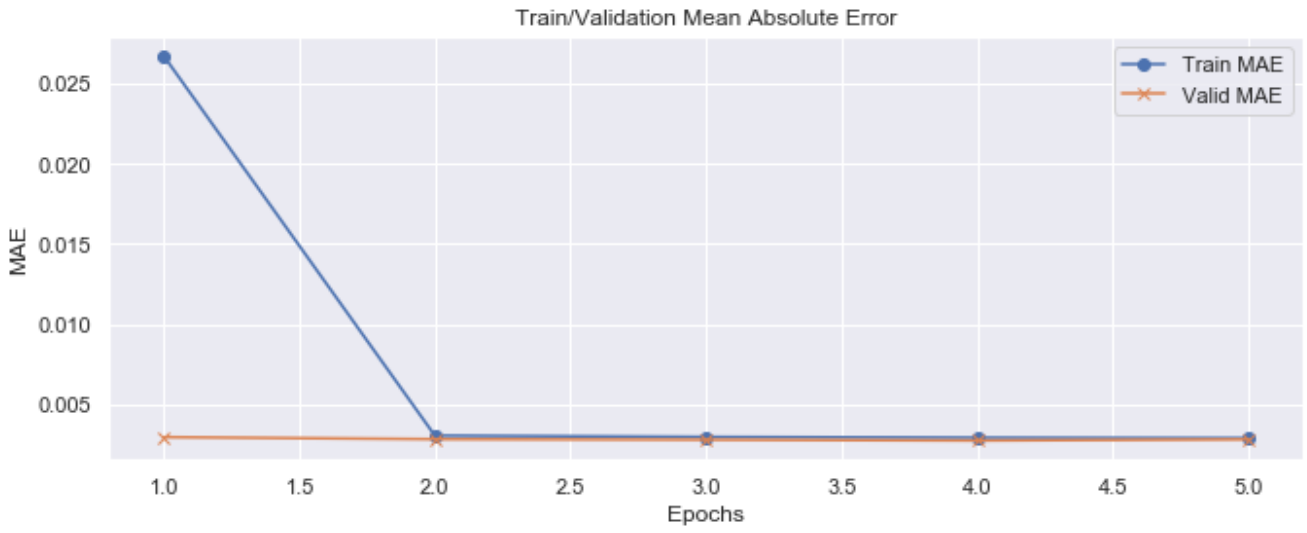


Fig.94 MLP Mean Absolute Error

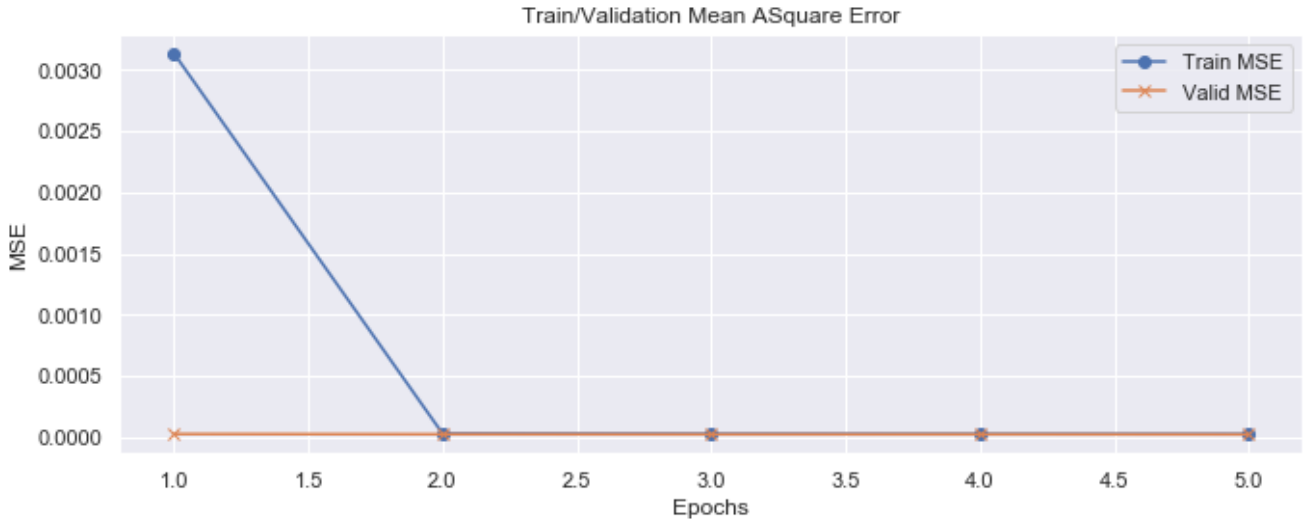


Fig.95 MLP Mean Square Error

Below are the test set predicted and true values of stock with respect to time.

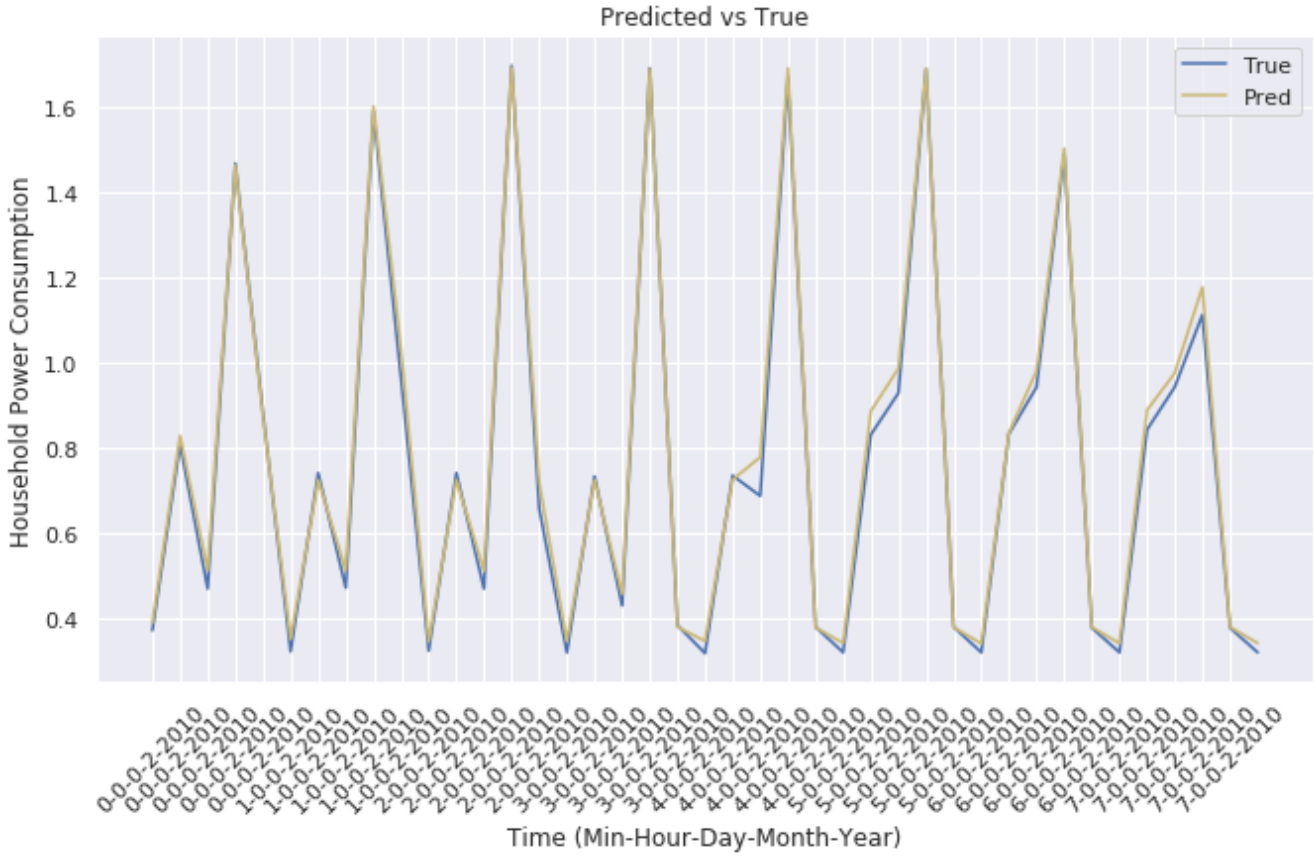


Fig.96 MLP Predicted vs True Stock Values

5.8.5 Results of Random Forest

Below are the test set predicted and true values of stock with respect to time.

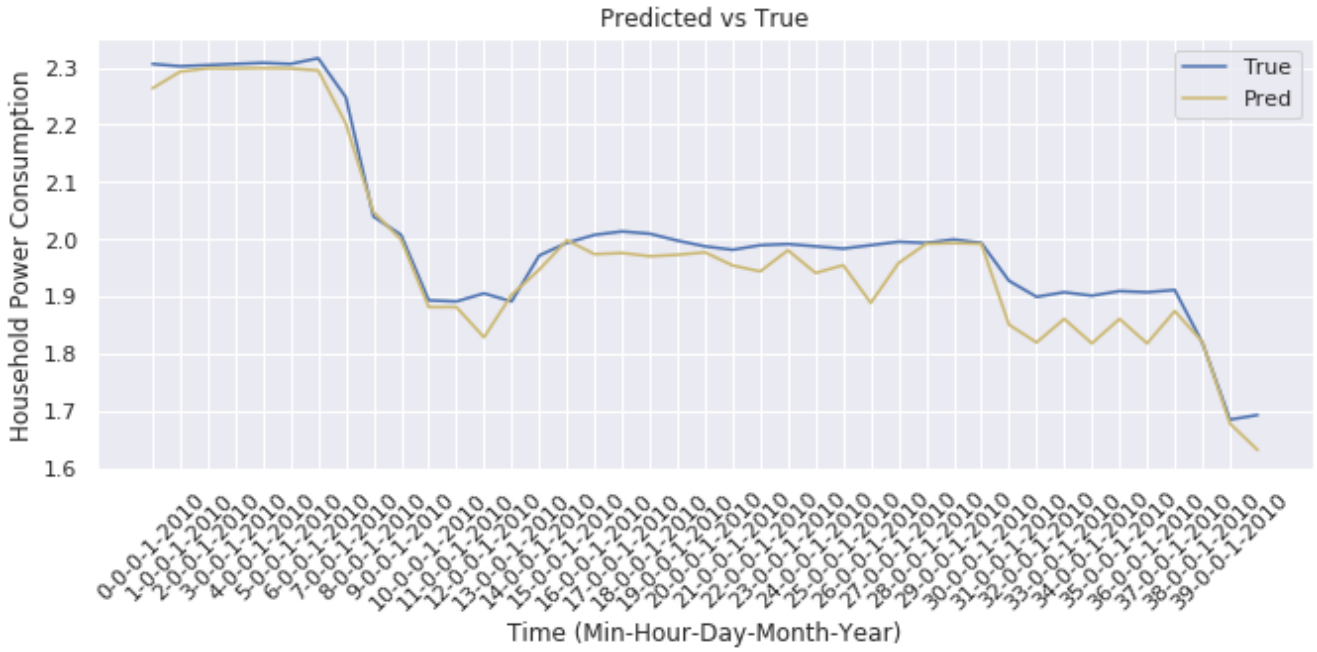


Fig.97 RF Predicted vs True Stock Values

5.8.6 Results of VAR

Below are the test set predicted and true values of stock with respect to time.

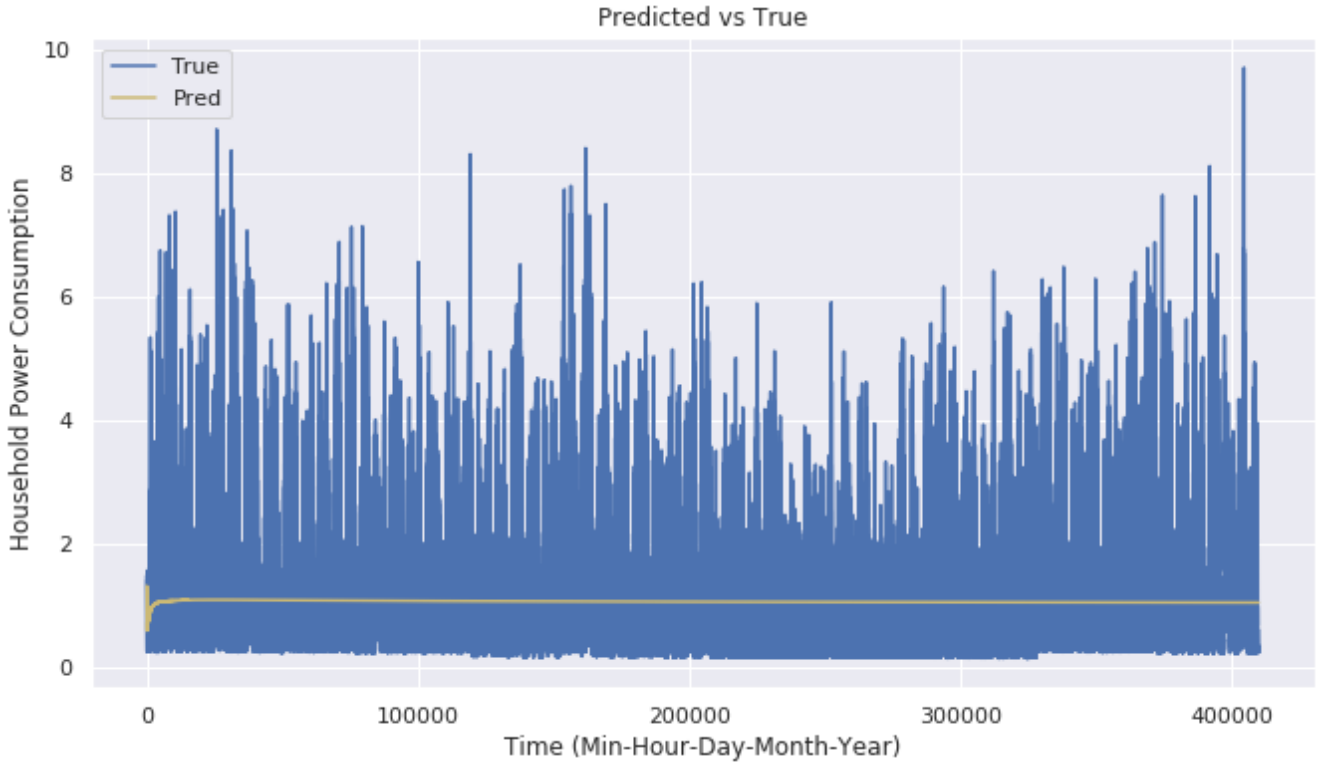


Fig.98 VAR Predicted vs True Stock Values

The evaluation metric values of all trained models on test set are shown in the table below:

Table:5. Household Power Consumption Results

	Random Forest	**MLP**	**CNN**	**LSTM**	**VAR**	**GRU**
R2-Score (%)	99.68	99.66	99.76	99.69	-0.001	99.60
MAE	0.002	0.002	0.002	0.002	0.715	0.004
MSE	0.001	0.001	0.001	0.001	0.803	0.001
RMSE	0.004	0.004	0.003	0.004	0.896	0.011

5.9 Outcomes of this Research

In this research multiple models were developed by extracting month, day and hour aggregation groupings from the acquired data set of stocks. The acquired data set has records of only five years, therefore, by extracting these aggregate groupings the number of observations were reduced significantly. Although, there is no proper rule for deep learning

training data set size, but a general thumb rule says ten thousand, which in our case was not possible. The maximum observations we obtained was for hour aggregate groupings (approximately eight thousand and six hundred). Additionally, the given data set has records of five years which limits the generalization of the developed models. Another limitation in the given data set was that there were a lot of missing records. In stocks data set the first ten months record of 2012 were missing. Similarly, in household data almost 25,000 recorded values were missing, which reduced the training set size. However, the developed models are still able to forecast the output efficiently.

6 Problems during Research

The first problem observed in both data sets was that they were not in ascending order according to time. To solve this issue both data sets were ordered in ascending according to time features (year, month, day, hour). The next problem faced was that there were approximately 25,000 missing values in household data. The missing values appear as null in the pandas's Dataframe. Machine learning models cannot be trained with missing values and will throw error. There were too many missing values and therefore interpolation or other statistical techniques to estimate those missing values was not possible. These missing values observation were removed from the data set.

The data point distribution of data sets especially stocks was uneven. To fix this problem data sets were normalized by scaling the observed between 0 and 1. The highest value in the data set is scaled to 1 and minimum value is scaled to 0. In this way both data set values are normalized.

The next problem was faced during training of LSTM, CNN and GRU models. Three dimensional input data is required by the input layer of these models, but the data we have was in two dimensional (number of observations are rows and number of input features are columns). To resolve the problem a third dimension was created with value equal to 1. After fixing this problem the training of these models was carried out smoothly.

Time series are related to the inspection of different quantities achieved through time interval. The presence of a time series is spotted in significant areas. The analysis of time series faces difficulties in understanding the features of the model in time series (Thompson & Wilson 2016).

The challenges which are faced in time series related to high dimension, curve value and non-stationary are:

1. **Re-define:** Non-stationary data is defined as problem-dependent, which helps in proposing new models. It further helps in forecasting and making financial data (Xu & Beck 2016).
2. **Propose new procedures:** This is done for reduction in dimensions of problem-dependent. This will help in constructing a new way to think about financial data of time series which dimensions are high (Yang et al. 2017). This approach is considered as problem dependent.

3. **Statistical model:** This relates to proposing new techniques to select models which are high dimensional and related to problems of regression in time series. It also includes the setting of non stationary. The new methods are used in the fields of statistical market and financial forecasting. The selection of the model must be appropriate for making an economic decision.
4. **Dependence structure:** Investigation is done for identifying new techniques and selecting models which are high dimensional. The financial return is one of the examples. In these changes in secondary structure is observed. The methods which are identified helps in analysing data of finance and make a business decision. (Zhao et al. 2016).
5. **Curves related to time series:** the methods which are proposed to analyse time series. It can be non-stationary in different ways.

7 Conclusions & Future Work

In this research time series machine learning and deep learning models are developed to forecast the stock values and household power consumption in real time based on time features. Multiple models are developed for this purposed including random forest, vector auto regression, multi-layer perceptron, long short term memory, convolution neural networks and gated recurrent unit. The reason behind developing multiple models is because these models are dependent on data. One model may work better than the other based on data distribution and randomness. Another important point is as the final selected model will be deployed for real time forecasting, therefore, it is required to develop optimum model from the recorded values data set especially for stocks where risk of wrong forecasting can cause huge economic losses for investors. Additionally, in case of household power consumption the peak power consumption are forecasted for hour, day, month and year.

Different time aggregations are developed from the acquired data set of stocks and by utilizing these aggregations, models are created. For stock month aggregate CNN was having best results. For stocks day aggregate data set random forest was having best results and for hour aggregate data set gated recurrent unit was having optimum results. For household the results of LSTM were best.

It is recommended for the future to work on hybrid model as in the future data will become more complex therefore combination of any two or three models such as Random Forest and Neural network can give significant result.

References

Aizenberg, I, Sheremetov, L, Villa-Vargas, L & Martinez-Muñoz, J 2016, 'Multilayer neural network with multi-valued neurons in time series forecasting of oil production', *Neurocomputing*, vol. 175, pp. 980-989.

Banerjee, D 2014, 'Forecasting of Indian stock market using time-series ARIMA model', in *2014 2nd International Conference on Business and Information Management (ICBIM)*, pp. 131-135.

Bell, J 2014, *Machine learning: hands-on for developers and technical professionals*, John Wiley & Sons.

Botchkarev, A 2018, 'Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology', *arXiv preprint arXiv:1809.03006*.

Box, G, Jenkins, G, Reinsel, G & Ljung, G 2015, *Time series analysis, control, and forecasting . Hoboken*, New Jersey: John Wiley & Sons.

Brechmann, EC & Czado, C 2015, 'COPAR—Multivariate time series modeling using the copula autoregressive model', *Applied Stochastic Models in Business and Industry*, vol. 31, no. 4, pp. 495-514.

Chatfield, C 2013, *The analysis of time series: theory and practice*, Springer.

Chen, J-F, Wang, W-M & Huang, C-M 1995, 'Analysis of an adaptive time-series autoregressive moving-average (ARMA) model for short-term load forecasting', *Electric Power Systems Research*, vol. 34, no. 3, pp. 187-196.

Chen, JL, Li, G, Wu, DC & Shen, S 2019, 'Forecasting seasonal tourism demand using a multiseries structural time series method', *Journal of Travel Research*, vol. 58, no. 1, pp. 92-103.

Corchado, J., Fyfe, C. and Lees, B., 1998, March. Unsupervised learning for financial forecasting. In *Proceedings of the IEEE/IAFE/INFORMS 1998 Conference on Computational Intelligence for Financial Engineering (CIFEr)(Cat. No. 98TH8367)* (pp. 259-263). IEEE.

Davis, RA, Drees, H, Segers, J & Warchoł, M 2018, 'Inference on the tail process with application to financial time series modeling', *Journal of econometrics*, vol. 205, no. 2, pp. 508-525.

Dias, JG, Vermunt, JK & Ramos, S 2015, 'Clustering financial time series: New insights from an extended hidden Markov model', *European Journal of Operational Research*, vol. 243, no. 3, pp. 852-864.

Dudek, G 2015, 'Short-term load forecasting using random forests', in *Intelligent Systems' 2014*, Springer.

Gong, X, Si, Y-W, Fong, S & Biuk-Aghai, RP 2016, 'Financial time series pattern matching with extended UCR Suite and Support Vector Machine', *Expert Systems with Applications*, vol. 55, pp. 284-296.

Guresen, E, Kayakutlu, G & Daim, TU 2011, 'Using artificial neural network models in stock market index prediction', *Expert Systems with Applications*, vol. 38, no. 8, pp. 10389-10397.

Hamzaçebi, C 2007, 'Forecasting of Turkey's net electricity energy consumption on sectoral bases', *Energy policy*, vol. 35, no. 3, pp. 2009-2016.

Harris, E, Abdul-Aziz, A & Avuglah, R 2012, 'Modeling annual Coffee production in Ghana using ARIMA time series Model', *International Journal of Business and Social Research*, vol. 2, no. 7, pp. 175-186.

Hontoria, L, Aguilera, J & Zufiria, P 2002, 'Generation of hourly irradiation synthetic series using the neural network multilayer perceptron', *Solar Energy*, vol. 72, no. 5, pp. 441-446.

Jain, G & Mallick, B 2017, 'A study of time series models ARIMA and ETS', *Available at SSRN 2898968*.

Jian, L, Zhao, Y, Zhu, Y-P, Zhang, M-B & Bertolatti, D 2012, 'An application of ARIMA model to predict submicron particle concentrations from meteorological factors at a busy roadside in Hangzhou, China', *Science of the Total Environment*, vol. 426, pp. 336-345.

Kalekar, PS 2004, 'Time series forecasting using holt-winters exponential smoothing', *Kanwal Rekhi School of Information Technology*, vol. 4329008, no. 13.

Kingma, DP & Ba, J 2014, 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980*.

Kong, W, Dong, ZY, Jia, Y, Hill, DJ, Xu, Y & Zhang, Y 2017, 'Short-term residential load forecasting based on LSTM recurrent neural network', *IEEE Transactions on Smart Grid*, vol. 10, no. 1, pp. 841-851.

Kotsiantis, S, Kanellopoulos, D & Pintelas, P 2006, 'Data preprocessing for supervised leaning', *International Journal of Computer Science*, vol. 1, no. 2, pp. 111-117.

Lai, G, Chang, W-C, Yang, Y & Liu, H 2018, 'Modeling long-and short-term temporal patterns with deep neural networks', in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 95-104.

Laptev, N, Yosinski, J, Li, LE & Smyl, S 2017, 'Time-series extreme event forecasting with neural networks at uber', in *International Conference on Machine Learning*, vol. 34, pp. 1-5.

Lin, T, Guo, T & Aberer, K 2017, 'Hybrid neural networks for learning the trend in time series', in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pp. 2273-2279.

Lora, AT, Santos, JMR, Riquelme, JC, Expósito, AG & Ramos, JLM 2003, 'Time-series prediction: Application to the short-term electric energy demand', in *Conference on Technology Transfer*, pp. 577-586.

Lv, Y, Duan, Y, Kang, W, Li, Z & Wang, F-Y 2014, 'Traffic flow prediction with big data: a deep learning approach', *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865-873.

Markham, IS, Mathieu, RG & Wray, BA 2000, 'Kanban setting through artificial intelligence: A comparative study of artificial neural networks and decision trees', *Integrated Manufacturing Systems*, vol. 11, no. 4, pp. 239-246.

Menezes, JM & Barreto, GA 2006, 'On recurrent neural networks for auto-similar traffic prediction: A performance evaluation', in *2006 International Telecommunications Symposium*, pp. 534-539.

Montori, F, Liao, K, Jayaraman, PP, Bononi, L, Sellis, T & Georgakopoulos, D 2018, 'Classification and Annotation of Open Internet of Things Datastreams', in *International Conference on Web Information Systems Engineering*, pp. 209-224.

Nwankpa, C, Ijomah, W, Gachagan, A & Marshall, S 2018, 'Activation functions: Comparison of trends in practice and research for deep learning', *arXiv preprint arXiv:1811.03378*.

Ozturk, S & Ozturk, F 2018, 'Forecasting energy consumption of Turkey by Arima model', *Journal of Asian Scientific Research*, vol. 8, no. 2, p. 52.

Patterson, J & Gibson, A 2017, *Deep learning: A practitioner's approach*, " O'Reilly Media, Inc.".

Qiu, X, Ren, Y, Suganthan, PN & Amaratunga, GA 2017, 'Empirical mode decomposition based ensemble deep learning for load demand time series forecasting', *Applied Soft Computing*, vol. 54, pp. 246-255.

Rawat, W & Wang, Z 2017, 'Deep convolutional neural networks for image classification: A comprehensive review', *Neural computation*, vol. 29, no. 9, pp. 2352-2449.

Thomakos, DD & Nikolopoulos, K 2015, 'Forecasting multivariate time series with the theta method', *Journal of Forecasting*, vol. 34, no. 3, pp. 220-229.

Thompson, JR & Wilson, JR 2016, 'Multifractal detrended fluctuation analysis: Practical applications to financial time series', *Mathematics and Computers in Simulation*, vol. 126, pp. 63-88.

Tsantekidis, A, Passalis, N, Tefas, A, Kanniainen, J, Gabbouj, M & Iosifidis, A 2017, 'Forecasting stock prices from the limit order book using convolutional neural networks', in *2017 IEEE 19th Conference on Business Informatics (CBI)*, vol. 1, pp. 7-12.

Wang, Y, Liao, W & Chang, Y 2018, 'Gated recurrent unit network-based short-term photovoltaic forecasting', *Energies*, vol. 11, no. 8, p. 2163.

Wenbin Wu & Mugen Peng, 2017. A Data Mining Approach Combining K -Means Clustering With Bagging Neural Network for Short-Term Wind Power Forecasting. IEEE Internet of Things Journal, 4(4), pp.979–986.

Wu, W & Peng, M 2017, 'A Data Mining Approach Combining $ K $-Means Clustering With Bagging Neural Network for Short-Term Wind Power Forecasting', *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 979-986.

Xie, L, Xiao, J, Hu, Y, Zhao, H & Xiao, Y 2017, 'China's energy consumption forecasting by GMDH based auto-regressive model', *Journal of Systems Science and Complexity*, vol. 30, no. 6, pp. 1332-1349.

Xu, D & Beck, C 2016, 'Transition from lognormal to χ2-superstatistics for financial time series', *Physica A: Statistical Mechanics and its Applications*, vol. 453, pp. 173-183.

Yang, P, Shang, P & Lin, A 2017, 'Financial time series analysis based on effective phase transfer entropy', *Physica A: Statistical Mechanics and its Applications*, vol. 468, pp. 398-408.

Zaremba, W, Sutskever, I & Vinyals, O 2014, 'Recurrent neural network regularization', *arXiv preprint arXiv:1409.2329*.

Zhang, GP 2003, 'Time series forecasting using a hybrid ARIMA and neural network model', *Neurocomputing*, vol. 50, pp. 159-175.

Zhang, H, Wang, X, Cao, J, Tang, M & Guo, Y 2018, 'A multivariate short-term traffic flow forecasting method based on wavelet analysis and seasonal time series', *Applied Intelligence*, vol. 48, no. 10, pp. 3827-3838.

Zhang, J.S. and Xiao, X.C., 2000. Predicting chaotic time series using recurrent neural network. *Chinese Physics Letter*, *17*(2), p.88.

Zhang, Y, Zhong, M, Geng, N & Jiang, Y 2017, 'Forecasting electric vehicles sales with univariate and multivariate time series models: The case of China', *PloS one*, vol. 12, no. 5, p. e0176729.

Zhao, ZY, Xie, M & West, M 2016, 'Dynamic dependence networks: Financial time series forecasting and portfolio decisions', *Applied Stochastic Models in Business and Industry*, vol. 32, no. 3, pp. 311-332.

Zivot, E & Wang, J 2007, *Modeling financial time series with S-Plus®*, vol. 191, Springer Science & Business Media.