Analyzing performance of various machine learning algorithms on Walmart Stock data.

Aakarsh Nadella[1], Prem Chand Avanigadda[1], Vinay Kumar Nalla[2], Veera Venkata Sai Naveen Bodangi[1]

[1]Department of Computer and Information Sciences, School of Science, IUPUI.

[2]Department of Mechanical Engineering, School of engineering and Technology, IUPUI.

Abstract- Building an Intelligent system using Predictive Analytics on Equities, we can provide suggestions to the investor whether it is wise to invest or not. Applying the concepts of machine learning algorithms such as Neural Networks, ARIMA (Auto-regressive Integrated Moving Average) and Support Vector Machines (SVM) on Stock data we can derive hidden patterns by building a predictive model. Upon building the models, the performance of the model is evaluated, and the efficiency of the algorithms applied on this dataset are compared.

Keywords- Intelligent systems, machine learning, Neural Networks, ARIMA, SVM, predictive model, Stock data.

1. Introduction

Building an Intelligence system for forecasting stock prices is one of the most challenging problem of modern AI. As in the real world, the stock market is dynamic and complex to understand. The prices of the stocks fluctuate randomly from time to time. At times it is difficult for a naive investor to understand this high degree of uncertainty in price movements. So, to provide a system which collaborates the financial data along with technical knowledge, ensures an easier way for the end user to give suggestions whether to invest in a stock or not. This forecasting of stock data can be done using predictive analytics which comes under the broad category of Data Analytics. The methodology in which a dataset is analyzed to find the hidden patterns and behavior, thereby providing insights which are useful is called Data mining. Various machine learning algorithms are used to facilitate this process.

Supervised Learning Algorithms and unsupervised learning algorithms are 2 branches of the machine learning algorithms [1]. In supervised learning algorithms, the dataset generally is a labelled dataset which is trained to build a predictive model.

Unsupervised learning are self-organizing algorithms which deal with unlabeled data. Algorithms are designed to infer the hidden patterns and underlying relationship between the data. They are left to their own direction of making clusters and associations.

2. Problem Statement

The main aim of the project is to understand and implement some of machine learning algorithms based on the dataset chosen. As the dataset is a time-series dataset, we used algorithms such as Recurrent Neural Networks, ARIMA (Auto Regressive Integrated Moving Average) and Support Vector Machines. Upon building the predictive models for each algorithm, we shall forecast the values and compare the algorithm's performance on this dataset.

3. Dataset

The dataset chosen for this project is Walmart Stock Price dataset. This is a time-series dataset containing thirteen attributes, which are as shown in Fig 1. This dataset contains stock prices and their trading numbers starting from August 1972 to present.

- Date
- Open
- High
- Low
- Close
- Split
- Dividend
- Volume
- Adj_Open
- Adj_High
- Adj_Low
- Adj_Close
- Adj_Volume

Fig 1: Dataset Attributes

4. Methodology

The whole process, leading to goal of prediction and performance analysis is broken down into five different phases to make the problem have a simpler view. The first phase involves pre-processing of data. In the next phase we applied feature selection to find the best set of features for prediction, which is then followed by splitting of data into test and train datasets. Once the dataset is divided to test and train, we built a predictive model based on the machine learning algorithm used and predicted the values. In the last phase, based on the predicted values and actual values, performance of algorithms is analyzed based on the Root Mean Square Error. To perform all these tasks, we used R programming as a platform, as it gives better results and performs well with statistical data.

4.1 Data Pre-Processing

At times, the data may be noisy, such as outliers. Now to convert this raw data into structured data of readable format, we need to scale the data. This phase improves the quality of data, ensuring better results in the prediction process. We used '*Min-Max Normalization*' for data pre-processing. The formula for Min-Max normalization is as shown in Equation 1, where original value is represented by X normalized value is X'. In this normalization, the data is scaled over a range of value from 0 to 1.

$$X' = \frac{X - \min(X)}{\max(X) - \min(X)} \quad (1)$$

4.2 Feature Selection

Feature selection is finding right set of attributes for forecasting dependent variable. By doing so, the features that are irrelevant and redundant, which have almost no or adverse effect on the dependent variable which must be predicted are removed. The reason why we need to do feature selection is to improve efficiency of the predictive model by minimizing dimensionality and chance of model from becoming overfitted [2]. Also, the time taken for the model to be built can be reduced. There are various ways in which feature extraction can be done. In this work, we have used recursive feature elimination using random forest algorithm.

Recursive Feature Elimination is a methodology which responds back with best subset of attributes. Initially the process starts with all the attributes as input. Then the classifier is build and based on importance each attribute is assigned a rank. The attribute with least rank is left behind. Now the new set of attributes, leaving behind the attribute in the first iteration is sent as input to next iteration. The whole process is repeated until there is only one attribute left to be send to the input. Fig 2 shows the implementation of recursive feature elimination. Where using 'rfeControl', an object is created which specifies that classifier must be trained using random forest algorithm. Later this object is passed to rfe function to perform feature selection.

```
> control <- rfeControl(functions=rfFuncs, method="cv")
> results <- rfe(walmart[,-5], walmart[,5], rfeControl=control)
```

Fig 2: Recursive Feature Elimination

Finally, based on ranking, the optimal set of features are returned as output as shown in Fig 3, which are used to build predictive models.

```
The top 5 variables (out of 12):
   High, Low, Open, Adj_Volume, Adj_Close
```

Fig 3: Results of Feature Selection

4.3 Splitting of Data

Before building the predictive model, we need to split the data to training data and testing data. The training data is used to build the predictive model whereas the testing data is later used to compare with the predicted values and find the error rate. In our work, we have divided the data into training and testing in the ratio 80:20. The reason for splitting the data in this ratio is to train the model efficiently so that it gives high accuracy. Splitting of data is done randomly using sample function as shown in Fig 4.

```
> indices <- sample(1:nrow(walmart),size = 0.2 * nrow(walmart))
> traindata <- walmart[-indices,]
> testdata <- walmart[indices,]
```

Fig 4: Splitting data to Train and Test

4.4 Recurrent Neural Network

Neural network is a kind of machine learning algorithm, in which the model learns from the trained data on its own. The structure of artificial neural network is similar to neural network of a human brain. This network consists of nodes (like neurons in human brain) in multiple layers which are arranged in a hierarchical manner. Usually there is a single input, and output layer, but there can be more than 1 hidden layers in between. The information is passed from one node to another node, over the layers using weights associated with each one. Activation function is applied on all weights and the output is calculated. The different neural networks that are widely used are: Feed Forward Neural Networks, Back Propagation Neural Networks and Recurrent Neural Networks.

When the data is passed over different layers in only one direction, the concept is referred to as Feed Forward Neural Networks. This network is generally used for Pattern Recognition, Speech Recognition etc. There might be some error associated each node of a layer. When this node sends its weight along with the error to other layers, the error at the final layer may increase. So, to reduce this, the error is sent back to the same layer or the previous layer, thereby improving efficiency. This type of neural network is Back Propagation Neural Network.

Recurrent Neural Network follows back propagation algorithm. As the dataset is time-series, we need a mechanism to store previous values, so for facilitating this purpose we used Elman Recurrent Neural Network whose architecture is as shown in Fig 5. In this network, all the outputs are propagated back to same layer and other previous layers. In addition to input, hidden and output layers, there is new layer called context layer which is used to store temporary values.

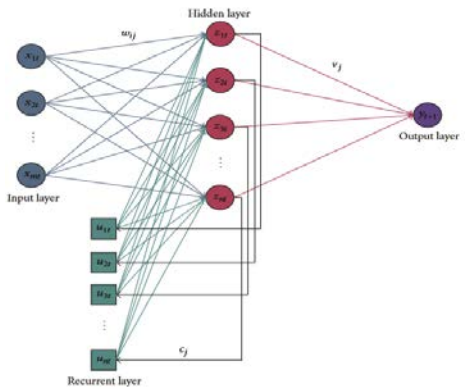

Fig 5: Elman Recurrent Neural Network [3]

In case of predicting a group of values, once the first value is predicted based on the predictive model built, the first value

that is predicted is stored in a temporary memory location in the context layer [3]. For the second value to be forecasted, both the predictive model along with first value is used as reference. Now the second value goes to the memory location. This process is iterated until all the values are predicted. As the memory location stores a value for short period of time and as this process lasts for long time, this recurrent neural network is referred to as Long Short-Term Memory Neural Network. The recurrent neural network model is built as shown in Fig 6 using '*elman*' function where dependent and independent variables are passed as argument. Size represents hidden layers and maxit is total iterations to be performed.

```
fit1 <- elman(traindata[,1:3], traindata[,4],
size = 50, maxit = 1000, learnFuncParams = c(0.01))
```

Fig 6: Building Recurrent Neural Network Model

4.5 ARIMA

Time series data is best analyzed by autoregressive and moving average models. Auto regression means that time series univariate values are regressed, which helps to predict the further value in the time series. This autoregressive model is made optimized by itself, finding autocorrelation between predicted value and previous series. This correlation coefficient is added to the series to make it efficient.

$$correlation(y_1, y_{1-k}) \qquad (2)$$

where *k* is called lag between two-time periods, correlation coefficient is added to regression model.

Moving average is used to make the linear regression model consistent. Using moving average, we calculate the difference between means of time series before and after adding of a predicted value. We add this difference to regression model to make it perfect fit.

ARIMA stands for autoregressive integrated moving average, which deals with time series data by combination of three methods auto regression, differencing and moving average. Advantage of ARIMA is it can work with non-stationary data. Stationary data means which is consistent throughout the period, where non-stationary data is seasonal i.e. data changes consistently through various time periods. ARIMA (*p, d, q*) [4] can be represented as

$$y_t = \varphi y_{t-1} + \varphi y_{t-2} + \cdots + \varphi_P y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} \qquad (6)$$

Where Y_t is time series values at different "*t*" times. φ and θ are constants, *P* and *q* are auto regressive and lagged error terms respectively. Firstly, we perform Augmented Dickey-Fuller unit root test to know the stationarity of data. as shown in figure 7.

```
> print(adf.test(stock))

        Augmented Dickey-Fuller Test

data:  stock
Dickey-Fuller = -4.7728, Lag order = 22, p-value = 0.01
alternative hypothesis: stationary
```

Fig 7: ADF test

If *p*-value is more than 0.05, then data has a unit root mean, i.e. data is non-stationary. We got *p* value as 0.01, so there is no seasonality in data indicates data is stationary. So, let's take differencing variable *d=0*, i.e. ARIMA (*p, 0, q*). If the data is non-stationary, we apply first order and second order differencing techniques to data to change data to stationary. If *d=1* it will find the difference between two consecutive values in each time series called as first order differencing. If *d=2*, it will find second order differencing.

Secondly, we convert the data to logarithmic returns to perform autocorrelation function(ACF) [4] and partial autocorrelation function(PACF) on data to get *q* and *p* values respectively. In ACF it finds the correlation between data points separated by the distance of lag. In PACF it removes the effect of smaller lags on ACF.

When we perform ACF on the logarithmic returns of data, we plotted the results to get a graph as shown in figure 8.

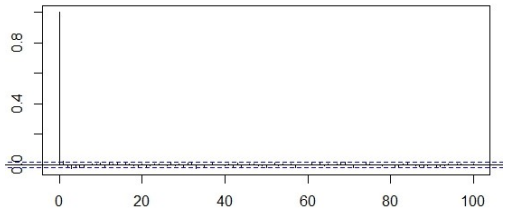

Fig 8: ACF test plot

We observed two spikes in the graph, which describes moving average at two lags with order q=2, i.e. ARIMA *(p, 0, 2).* In figure 9, we plotted the results of PACF.

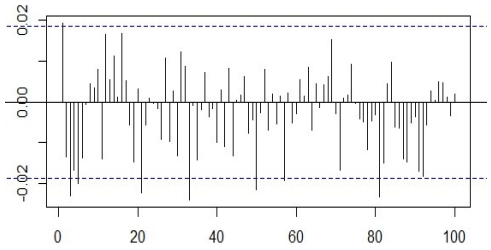

Fig 9: PACF test plot

We observed 4 spikes in the graph showing 4 lags in the data. so, value of *p* is 4 i.e. ARIMA *(4, 0, 2).* We build the model by ARIMA (4, 0, 2) with training data as show in figure 10.

```
fit = arima(stock_train, order = c(4, 0, 2),include.mean=FALSE)
arima.forecast = forecast(fit, h = 1,level=99)
```

Fig 10: Arima model

We get the predicted value through passing '*fit*' into forecast function with 99 percent confidence. Where '*h*' gives number of predictions. This predicted value is added to train set and again ARIMA model is refitted to predict the next time series value. Plot of the actual results and forecasted results, is illustrated in figure 11.

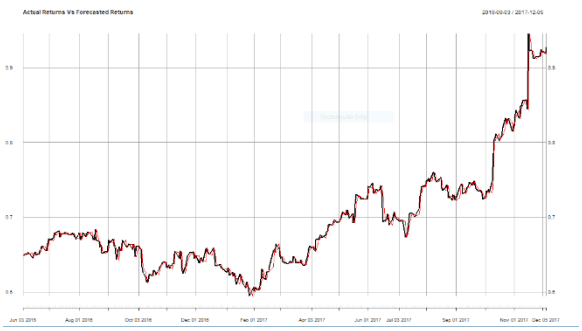

Fig 11: Forecast series plot

The black line is the actual series whereas red line is the forecasted series, which are overlapped. So, we can assume our model predicting with some actual values.

4.6 Support Vector Machines

Support vector machine is a supervised learning algorithm which creates a hyperplane between two different clauses of data points. Marginal data points near hyperplane are called support vectors. There are two methods in SVM namely classification and regression. While in regression method we construct a hyperplane such that it is close to maximum number of data points. SVM is widely used for its ability to deal with more number of dimensions by Kernel functions [5], which are used to convert the non-linear data with high dimensions into linearly separable data.

Firstly, we build the SVM model for train data. From summarizing this SVM model we get kernel information. with this

information we rebuild the SVM model. As shown in figure 12.

```
> svm_model <- svm(traindata$Close ~., data=traindata)
> model <- svm(traindata$Close~.,data=traindata,kernel='radial',gamma=0.3334)
> preds <- predict(model,testdata[,c(1,2,3)])
```

Fig 12: SVM model

From this SVM model we predict the outputs for test data. These predicted values are collected for RMSE test.

5. Performance Analysis

Performance of the models that are built are calculated using RMSE values. RMSE stands for Root Mean Square Error. It measures the error between predicted values and actual values. The difference between actual and predicted values are called residuals [6]. Applying square root to the average of squaring the residuals give root mean square error.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(Actual-predicted)^2}{n}} \qquad (7)$$

6. Conclusion

We calculated RMSE values for the predicted values of Recurrent Neural Networks, ARIMA, Support Vector Machines and actual values. Respective error rates are as shown in table 1.

	RNN	ARIMA	SVM
RMSE	0.0179	0.00885	1.6306

Table 1: Comparison of RMSE values

Among all models ARIMA shows least error rate, so ARIMA outperformed the RNN and SVM and became best model for Walmart stock data. As the dataset is stationary and has values from the last thirty years, ARIMA performed well.

References

[1] N. Mishra and D. Silakari, "Predictive Analytics: A Survey, Trends, Applications, Oppurtunities & Challenges", *International Journal of Computer Science and Information Technologies*, vol. 33, 2012.

[2] L. Ladha and T. Deepa, "FEATURE SELECTION METHODS AND ALGORITHMS", *International Journal on Computer Science and Engineering (IJCSE)*, vol. 3, no. 5, 2011.

[3] J. Wang, J. Wang, W. Fang and H. Niu, "Financial Time Series Prediction Using Elman Recurrent Random Neural Networks", *Computational Intelligence and Neuroscience*, vol. 2016, pp. 1-14, 2016.

[4] R. programming, "Forecasting Stock Returns using ARIMA model", *R-bloggers*, 2017. [Online]. Available: https://www.r-bloggers.com/forecasting-stock-returns-using-arima-model/. [Accessed: 11- Dec- 2017].

[5] "Support vector machines: The linearly separable case", *Nlp.stanford.edu*, 2017. [Online]. Available: https://nlp.stanford.edu/IR-book/html/htmledition/support-vector-machines-the-linearly-separable-case-1.html. [Accessed: 11- Dec- 2017].

[6] "RMS Error", *Statweb.stanford.edu*, 2017. [Online]. Available: http://statweb.stanford.edu/~susan/courses/s60/split/node60.html. [Accessed: 11- Dec- 2017].