International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163
Volume 7, Issue 03 (March 2020) www.ijirae.com
Special Issue – 6th International National Level Conference- "MEEMIC-2020"

STOCK VALUE PREDICTION USING MACHINE LEARNING

Prof.B.Deepa.[1], Anusiya.T[2], Dhinesh kumar.S[3] Nivetha.S[4]
Assistant Professor[1], Sengunthar College of Engineering, Tiruchengode.
UG Scholar [2][3][4]
Department of Information Technology,
Paavai Engineering College, Namakkal, Tamil Nadu, India
[1].anusiyait502@gmail.com,[2].dineshhits101@gmal.com,[3].nivethachens@gmail.com

Manuscript History
Number: **IJIRAE/RS/Vol.07/Issue03/Special Issue/35.MRAESCE10080**
Received: 15, February 2020
Final Correction: 27, February 2020
Final Accepted: 10, March 2020
Published: **14, March 2020**
Editor: Dr.A.Arul Lawrence selvakumar, Chief Editor, IJIRAE, AM Publications, India

Abstract: In the last few years, machine learning has become a very popular tool for analysing financial text data, with many promising results in stock price from financial news, and a past values. In this work, we explore the Regression techniques, KNN techniques and one of the deep learning methods like Recurrent Neural Networks. From all these three techniques, we have a conclusion of RNN techniques helps more to analysing the stock price easily and overcome with issues associated with the accuracy of the overall values given. This paper also presents a Web API creation that build using Django Framework. It automatically updates the current value of the stock prices. The successful prediction of Stock will be a big legacy for the stock investing institutions.

Index Terms: Stock Prediction; Recurrent Neural Network; K Nearest Neighbour; LSTM; Data pre-processing;

INTRODUCTION

A stock market prediction is an attempt to forecast the future value of an individual stock, a particular sector or the market, or the market as a whole. These forecasts generally use fundamental analysis of a company or economy, or technical analysis of charts, or a combination of the two. The prediction is expected to be robust, accurate and efficient. The system must work according to the real-life scenarios and should be well. Predictive method like RNN, regression and KNN techniques is used. The data's are collected from the yahoo finance website. RNN techniques perform actions multiple times with the data's; hence it increases the predictive accuracy and reduces the over fitting of the dataset. There are various methods implementing the prediction system like statistical analysis, Fundamental Analysis, Domain Knowledge of Stock values, Technical Analysis, Machine Learning, Stock Investing Institutions and dataset structuring. Machine learning involves with Artificial Intelligence which empowers the system to learn and improve the past experience without being programmed time and again. There is a need to accurately predict the stock market which can be used in the real-life scenario. The datasets of the stock market include prediction model include details like Open values, Close values, Adjacent Close values etc., From these values, best-one will be chosen to predict the price of the stock. It reduces the overfit problem of the dataset.

DATA MINING

Stock values are collected from the site called yahoo finance. Yahoo! Finance is a media property that is part of Yahoo!'s network. It provides financial news, data and commentary including stock quotes, press releases, financial reports ,and original content. Doesn't need to change a code to collect the present data. Data's are automatically uploaded in our system.

FEAUTURE ENGINEERING: Feature engineering is the process of using domain knowledge to extract features from raw data via data mining techniques. These features can be used to improve the performance of machine learning algorithms. Feature engineering can be considered as applied machine learning itself. Features are important to predictive models and influence results. A feature could be strongly relevant (i.e., the feature has information that doesn't exist in any other feature), relevant, weakly relevant (some information that other features include) or irrelevant. Even if some features are irrelevant, having too many is better than missing those that are important. Feature selection can be used to prevent over fitting.

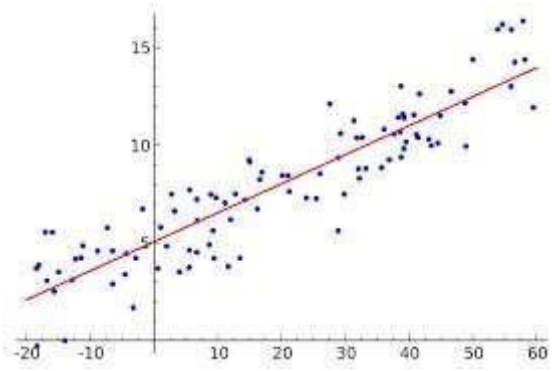
DATA PREPROCESSING:

Data pre-processing is a data mining technique which is used to transform the raw data in a useful and efficient format. There are so many techniques in data pre-processing. From that, Data transformation is preferred in this model. This step is taken in order to transform the data in appropriate forms suitable for mining process. This involves in Normalization. It is done in order to scale the data values in a specified range (-1.0 to 1.0 or 0.0 to 1.0). And then it performs Discretisation and Attribute Selection. In Discretisation, replace the raw values of numeric attribute by interval levels or conceptual levels. In Attribute Selection, new attributes are constructed from the given set of attributes to help the mining process.

MODEL USED:

LINEAR REGRESSION:

In statistics, **linear regression** is a linear approach to modelling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables).The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called **multiple linear regression**. This term is distinct from multivariate linear regression, where multiple correlated dependent variables are predicted, rather than a single scalar variable. It is a technique used to model the relationships between observed variables. The idea behind simple linear regression is to "fit" the observations of two variables into a linear relationship between them. Graphically, the task is to draw the line that is "best-fitting" or "closest" to the points (*xi*, *yi*), where *xi* and *yi* are observations of the two variables which are expected to depend linearly on each other.



The best-fitting linear relationship between the variables xx and yy.

Regression is a common process used in many applications of statistics in the real world. There are two main types of applications:

Predictions:

After a series of observations of variables, regression analysis gives a statistical mode for the relationship between the variables. This model can be used to generate predictions: given two variables x*x* and *y*, the model can predict values of y*y* given future observations of *x*. This idea is used to predict variables in countless situations, e.g. the outcome of political elections, the behaviour of the stock market, or the performance of a professional athlete.

Correlation: The model given by a regression analysis will often fit some kinds of data better than others. This can be used to analyse correlations between variables and to refine a statistical model to incorporate further inputs: if the model describes certain subsets of the data points very well, but is a poor predictor for other data points, it can be instructive to examine the differences between the different types of data points for a possible explanation. This type of application is common in scientific tests, e.g. of the effects of a proposed drug on the patients in a controlled study.

INTRODUCTION

Given a data set of n statistical units, a linear regression model assumes that the relationship between the dependent variable y and the p-vector of regressors **x** is linear. This relationship is modelled through a disturbance term or error variable ε — an unobserved random variable that adds "noise" to the linear relationship between the dependent variable and regressors. Thus, the model takes the form

$$\{yi, xi, \dots . . xip\} \lim_{i=1}^{n}$$

$$yi = \beta o + \beta 1 xi1 + \cdots + \beta p Xip + \epsilon i$$

where ^{T} denotes the transpose, so that $\mathbf{x}_i^T\boldsymbol{\beta}$ is the inner product between vectors $\mathbf{x}_i$ and $\boldsymbol{\beta}$. Often these n equations are stacked together and written in matrix notation as

$y = X_\beta + \varepsilon_i$ Where,

$$y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \begin{pmatrix} X_0^T \\ \vdots \\ X_n^T \end{pmatrix},$$

$$\beta_n = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix},$$

$$\varepsilon_n = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix},$$

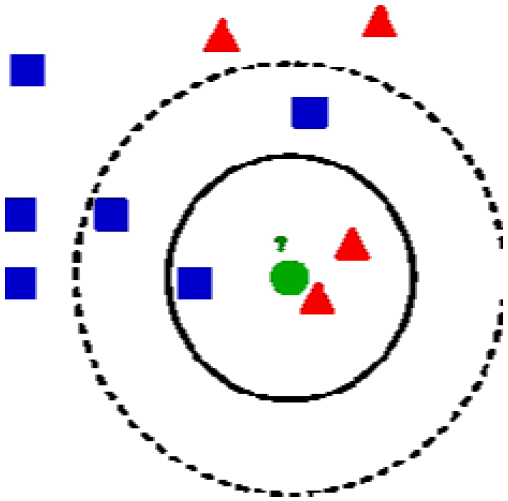
Some remarks on notation and terminology:

• y is a vector of observed values $(yi(i = 1, \dots , n))$ of the variable called the regression ,endogenous variable, response variable, measured variable, criterion variable, or dependent variable. This variable is also sometimes known as the predicted variable, but this should not be confused with predicted values, which are denoted y^. The decision as to which variable in a data set is modelled as the dependent variable and which are modelled as the independent variables may be based on ap resumption that the value of one of the variables is caused by, or directly influenced by the other variables.

K NEAREST NEIGHBOUR'S:

In pattern recognition, the k-nearest neighbours algorithm (k-NN) is a non parametric method used for classification and regression. In both cases, the input consist of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression: In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbour.
In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbours.

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until function evaluation. Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbours, so that the nearer neighbours contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbour a weight of 1/d, where d is the distance to the neighbour. The neighbours are taken from a set of objects for which the class (for k-NN classification) or the object property value (for k-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. A peculiarity of the k-NN algorithm is that it is sensitive to the local structure of the data.



Example of k-NN classification. The test sample (green dot) should be classified either to blue squares or to red triangles. If k = 3 (solid line circle) it is assigned to the red triangles because there are 2 triangles and only 1 square inside the inner circle. If k = 5 (dashed line circle) it is assigned to the blue squares.

PROPERTIES: *k*-NN is a special case of a variable-bandwidth, kernelensity "balloon" estimator with a uniform kernel. The naive version of the algorithm is easy to implement by computing the distances from the test example to all stored examples, but it is computationally intensive for large training sets. Using an approximate nearest neighbour search algorithm makes *k*-NN computationally tractable even for large data sets. Many nearest neighbour search algorithms have been proposed over the years; these generally seek to reduce the number of distance evaluations actually performed. *k*-NN has some strong consistency results. As the amount of data approaches infinity, the two class *k*-NN algorithm is guaranteed to yield an error rate no worse than twice the Bayes error rate(the minimum achievable error rate given the distribution of the data). Various improvements to the *k*-NN speed are possible by using proximity graphs.

For multi-class *k*-NN classification, Cover and Hart (1967) prove an upper bound error rate of

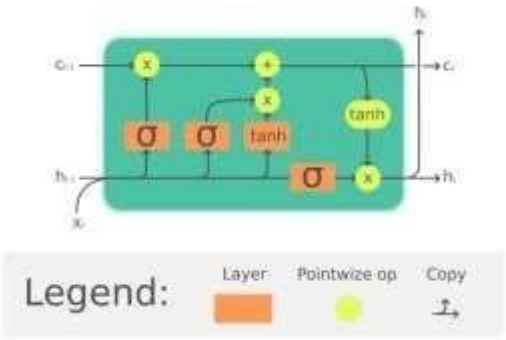
$$R^* \leq R_{KNN} \leq R^* \left(2 - \frac{M R^*}{R_{KNN}} M - 1\right)$$

Where is the Bayes error rate (which is the minimal error rate possible), is the *K*NN error rate, and *M* is the number of classes in the problem. For M = 2 and as the Bayesian error rate R^* approaches zero, this limit reduces to "not more than twice the Bayesian error rate.

LSTM (Long Short Term Memory Algorithm)

Long short-term memory (LSTM) is an artificial recurrent neural network(RNN) architecture used in the field of deep learning. Unlike standard feed forward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as un segmented, connected hand writing recognition, speech recognition and anomaly detection in network traffic or IDS's (intrusion detection systems).

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three *gates* regulate the flow of information into and out of the cell. LSTM networks are well suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the exploding and vanishing gradient problems that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications.



The Long Short-Term Memory (LSTM) cell can process data sequentially and keep its hidden state through time.

WORKING IDEA: In theory, classic (or "vanilla") RNNs can keep track of arbitrary long term dependencies in the input sequences. problem of vanilla RNNs is computational (or practical) in nature: when training a vanilla RNN using back propagation, the gradients which are back propagated can can't "explode" (that is, they can tend to infinity), because of the computations involved in the process, which use finite-precision numbers. RNNs using LSTM units partially solve the vanishing gradient problem, because LSTM units allow gradients to also flow *unchanged*. However, LSTM networks can still suffer from the exploding gradient problem. There are several architectures of LSTM units. A common architecture is composed of a cell (the memory part of the LSTM unit) and three "regulators", usually called gates, of the flow of information inside the LSTM unit: an input gate, an output gate and a forget gate. Some variations of the LSTM unit do not have one or more of these gates or maybe have other gates. For example, gated recurrent units(GRUs) do not have an output gate.

Intuitively, the *cell* is responsible for keeping track of the dependencies between the elements in the input sequence. The *input gate* controls the extent to which a new value flows into the cell, the *forget gate* controls the extent to which a value remains in the cell and the *output gate* controls the extent to which the value in the cell is used to compute the output activation of the LSTM unit. The activation function of the LSTM *gates* is often thelogistic sigmoid function.

There are connections into and out of the LSTM *gates*, a few of which are recurrent. The weights of these connections, which need to be learned during training, determine how the gates operate.

LSTM with a forget gate:
The compact forms of the equations for the forward pass of an LSTM unit with a forget gate are:
$ft = \sigma g(Wfxt + Ufht-1 + bf)$
$it = \sigma g(Wixt + Ujht-1 + bi)$
$ot = \sigma g(Woxt + Uoht-1 + bo)$
$ft = ftoct-1 + itoc\sigma(Wcxt + Ucht-1 + b_c)h_t = o_t o \sigma_h(c_t)$.

Where the initial values are c_o = 0 and h_o= 0 and the operator o denotes the Hadamard product. The subscript t indexes the time step.

Variables:

- $x_t \in \acute{R}^d$: input vector to the LSTM unit.
- $f_t \in \acute{R}^h$: forget gate's activation vector
- $i_t \in \acute{R}^h$: input/update gate's activation vector.
- $o_t \in \acute{R}^h$: output gate's activation vector
- $h_t \in \acute{R}^h$: hidden state vector also known as output vector of the LSTM unit.
- $c_t \in \acute{R}^h$: cell state vector
- $W \in \acute{R}^{h*d}$, $U \in \acute{R}^{h*h}$ and $b \in \acute{R}^h$: weight matrices and bias vector parameters which need to be learned during training.
- Where the superscripts d and h refer to the number of input features and number of hidden units, respectively.

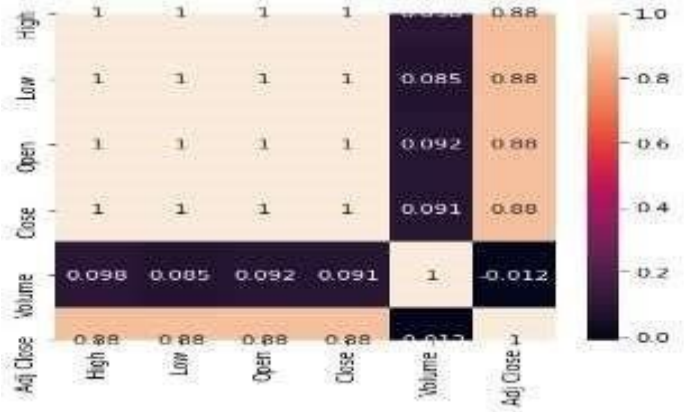
EXPERIMENTAL RESULTS:

We have taken a raw data of Infosys from the yahoo finance market website. Tail data's are shown below.

	Date	High	Low	Open	Close	Volume	Adj Close
4522	2018-12-24	9.24	9.07	9.19	9.08	8590700.0	8.797617
4523	2018-12-26	9.38	9.12	9.15	9.38	9004200.0	9.088287
4524	2018-12-27	9.45	9.28	9.30	9.45	9856500.0	9.156109
4525	2018-12-28	9.50	9.38	9.48	9.43	6818500.0	9.136732
4526	2018-12-31	9.53	9.39	9.47	9.52	7229400.0	9.223932

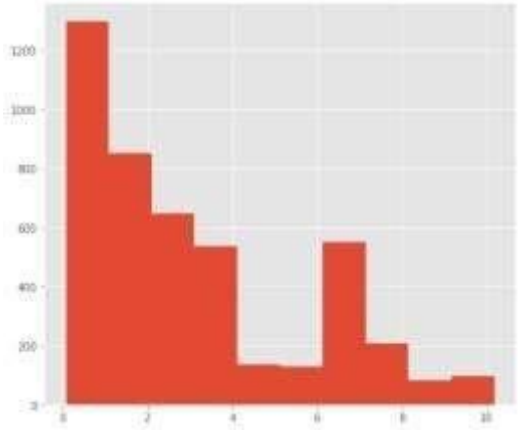
From this, we have a variable's like Date, Low, High, Open, Close and Adj Close. From these variables, we predicted a correct variable for stock value prediction by using feature engineering process. Reason for this variable, is to reduce the verify problem among the variables. See the correlation plot below,

Correlation plot between each and every variables.

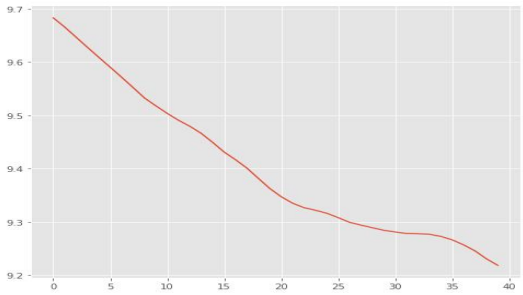


By seeing this, other than ADJACENT CLOSE all values are equal to 1. It might because a over fit problem among the models. Hence we taken a ADJACENT CLOSE for predict of stock values. Plot between ADJ CLOSE and DATE.

From this flow, we have concluded that the company gradually increases their shares and their company value throughout the decades. Histogram of ADJACENT CIOSE data,



After all EDA process, using the neural network libraries the model is predicted. Final results are regressor. count_params() 33601
Final Predicted Plot for last 40 Test values.



Using Confusion Matrix, our model predicted with the accuracy of 83.45%. It is the best accuracy value among all models we have predicted. And finally from this basis, we created a web API for this model using DJANGO framework.

CONCLUSION

The present work has shown how it is possible to perform predictions of future stock market data's using deep learning techniques, specifically machine learning algorithms. From proposed system, we have improved our accuracy using some advanced deep learning techniques. It might be a slower process, but it gives more accuracy than that. On predicted of Stock Values, we considered only depend upon results not on speed and time. That's the reason behind it, to predict the model with advanced algorithms. It should be a good initiative to create an API's using machine learning.

FUTURE ENHANCEMENT

From this model, we have planned to create a best algorithm model from this existing algorithm. By increasing the speed of the model using some big O techniques, and apply in it future. And also adding some public opinions about the model, we have predicted.

REFERENCES

1. Yujie Wang, Hui Liu, Qiang Guo, ShenXiang Xie, "Stock Volatility Prediction by Hybrid Neural Network", IEEE 2019.
2. Ashish Sharma ,Dinesh Bhuriya, Upendra Singh, "Survey of Stock Market Prediction Using Machine Learning Approach", ICECA, 2017.
3. Loke.K.S. "Impact of Financial Ratios and technical Analysis on Stock Price Prediction using Random Forests", IEEE, 2017.
4. SachinSampatPatil, Prof. Megha Jain, "A Survey on Stock Market Prediction Using SVM", IJCTET 2016.
5. RautSushrut Deepak, ShindelshaUday, Dr,D.Malathi, "Machine Learning approach (ICA)", DSJ 2016.
6. Thien Hai Nguyen, Kiyoaki Shirai, and Julien Velcin, "Sentiment analysis on social media for stock movement prediction", 2015.
7. S. Feurriegel and R. Fehrer, "Improving decision analytics with deep learning: The case of financial disclosures, " 2015.
8. Johan Bollen, Huina Mao, and Xiaojun Zeng, "Twitter mood predicts the stock market", 2011.
9. Wettschereck, D., Aha, D.W., Mohri.T, "Areview of empirical evaluation of feature weighting methods for a cass of lazy learning algorithms. Artif. Intell. Rev. 11(1-5), 273-314, 1995.
10. Raja, G P& Mangai, S 2018, 'Investigation On Optimization, Prioritizing and Weight Allocation Techniques for Load Balancing and Controlling Multimedia Traffic in Wireless Mesh Network', International Journal of Business Information Systems, SCOPUS Indexed Journal (Inderscience) - (P ISSN No: 1746-0972). Published Online: 10th Feb 2020, DOI: 10.1504/IJBIS.2020.105161.IF: 0.72.
11. Raja, G P& Mangai, S 2017, 'Firefly Load Balancing Based Energy Optimized Routing for Multimedia Data Delivery in Wireless Mesh Network', Cluster Computing-The Journal of Networks Software Tools and Applications, SCOPUS Indexed Journal (Springer) - (E ISSN No: 1573-7543).Published Online: 27th Dec 2017, https://doi.org/10.1007/s10586-017-1557-1. IF: 2.040.
12. Geetha. E & Nagarajan. C , 2019, 'Stochastic Rule Control Algorithm Based Enlistment of Induction Motor Parameters Monitoring in IoT Applications', Springer, Wireless Personal Communications. October 2018, Volume 102, Issue 4, pp 3629 - 3645.

13. FDTD Modeling and simulation of Microwave Heating for Egg Pasteurization", Archives Des Sciences, ISSN 1661-464X, Vol.65, p.10, Aug.2012.

14. Satish Kumar.R. and Sanavullah, M.Y.," Optimization using Genetic Algorithm for FDTD Modeling and simulation of Microwave Heating for Egg Pasteurization", European Journal of Science Research , ISSN 1450-216X, Vol.84, No.1.pp.81-90,2012.

15. Satish Kumar.R. and Sanavullah, M.Y.," Theoretical and Experimental identification of cooking spot for shell eggs without explosions in a domestic Microwave Oven", Canadian Journal on Electrical and Electronics Engineering, Vol. 1, No. 4, pp.71-78, June 2010.

16. Satish Kumar.R, K.Uma Devi, and Sanavullah, M.Y.," Performance Analysis of using Exterior Rotor Permanent Magnet Brushless DC (ERPMBLDC)Motor", Improvement by a Novel Peak Torque Excitation Technique", International journal of Innovative research in Advanced Engineering, Vol. 1, 2012, pp. 1-7 (Impact factor 1.311).

