BMS COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU

A Technical Seminar Report based on review of Research Publication/Patent

Stock Market Prediction Using Machine Learning Algorithms

Submitted in partial fulfillment for the award of degree of

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by:
SOHAN R KUMAR
1BM19CS159

Work carried out at

Internal Guide

Ms. Sunayana S
Assistant Professor
Department of CSE
BMS College of Engineering

Department of Computer Science and Engineering
BMS College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
2022-2023

BMS COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

I, Sohan R Kumar (1BM19CS159) student of 7th Semester, B.E, Department of Computer Science and Engineering, BMS College of Engineering, Bangalore, hereby declare that, this technical seminar entitled " **Stock Market Prediction Using Machine Learning Algorithms** " has been carried out under the guidance of Ms. Sunayana S**,** Assistant Professor, Department of CSE, BMS College of Engineering, Bangalore during the academic semester October - February 2023. I also declare that to the best of my knowledge and belief, the technical seminar report is not from part of any other report by any other students.

Signature of the Candidate

SOHAN R KUMAR (1BM19CS159)

BMS COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Technical Seminar titled "**Stock Market Prediction Using Machine Learning Algorithms**" has been carried out by SOHAN R KUMAR (1BM19CS159) during the academic year 2022-2023.

Signature of the guide
Ms. Sunayana S
Assistant Professor
Department of Computer Science and Engineering
BMS College of Engineering, Bangalore

Abstract

Investment Bankers, CA's, Hedge Fund / Portfolio Managers, Forex traders, Commodities Analysts. These have been historically considered to be among the most coveted professions of all time. Yet, if one fails to keep up with the demands of the day, one would find one's skills to be obsolete in this era of data analysis. Data Science has inarguably been the hottest domain of the decade, asserting its need in every single sphere of corporate life. It was not long ago when we discovered the massive potential of incorporating ML/AI in the financial world. Now, the very idea of the two being disjointed sounds strange. Data Science has been incremental in providing powerful insights (which people didn't even know existed) and helped massively increase efficiency, helping everyone from a scalp trader to a long-term debt investor. Accurate predictions, unbiased analysis, powerful tools that run through millions of rows of data in the blink of an eye have transformed the industry in ways we could've never imagined.

It is a challenging task to analyze the stock market returns accurately due to nonlinear behavior of the financial market and volatility. With the help of increased computational abilities, machine learning and improved methods of prediction there are efficient systems being developed to predict stock market prices. There are multiple factors that affect the movement of stock prices, as a part of this study essential features that affect the stock market have been considered. The dataset is cleaned prior to processing and is analyzed with a couple of python functions. The model is continuously trained with varied datasets to improve the knowledge base which could provide sufficiently accurate data.

A stock market prediction model is designed to be user friendly within a few characteristics affecting the stock market for the initial prototype of the model. The model is trained to be more effective with the real time data and provides appropriate output. Artificial Intelligence and Machine learning is being used to solve a plethora of problems in the world. The application of Machine Learning for this project is to provide a highly efficient, minimal investment in money, yielding higher income stocks to invest in. In everyday life, it is difficult for a long-term investor to find the correct time to make purchase decisions of certain stocks. While, it is useful for an intraday trader to predict the stock prices according to his requirement, book profits and increase the revenue.

Table of Contents

Chapter 1: Introduction

1.1 Overview

Analysis and prediction of stock prices is important to enable wise investment choices. The primary objective of this paper is to predict the stock prices considering various factors which affect the market value of the concerned share. Since the market changes are very rapid and drastic enough, this model can be used to keenly observe the current market trend and track the price of a share. Trading algorithms are a new buzz in the finance industry and this paper provides the inclusion of machine learning in the stock market to help the investors. Traditionally, a stock is analyzed via Fundamental analysis and Technical Analysis.

The Fundamental analysis is highly focused on the long-term growth of the company and all the financial statements like balance sheet, Profit and Loss and Cash flow statements of a company are studied along with important ratios like price to equity and debt to equity ratios.

While Fundamental analysis focuses on which stock to buy, Technical Analysis focuses on when to buy the stock, this includes traditional methods like candlestick method. The stock market prediction design is more relevant for technical analysis of the stocks and gauges the price movements to invest money in the stock market. Currently, many intraday traders use similar trading algorithms which helps them in faster decision making by simplifying the available real time data. Our solution is intended to predict the stock prices based on the criterions considered. The aim is to improve the model's capability of handling numerous factors that affect the market behavior.

1.2 Motivation

The volatility of the market poses a huge challenge for the analysts to anticipate the rise or fall of the market with precision. With the introduction of machine learning solutions, the algorithm can be trained in a robust manner with recent market updates and tracking the stock movement is made easier. The stock market research involves extensive research and an efficient solution is required to analyze the historic data to make buy or sell decisions. The modules designed in this paper provide a model to analyze data with the help of recurrent neural nets.

1.3 Objective

Artificial Intelligence and Machine learning is being used to solve a plethora of problems in the world. The application of Machine Learning for this project is to provide a highly effective, minimal monetary investment, maximum income yielding stocks to invest in. In everyday life, it is difficult for a long-term investor to find the right time to purchase certain stocks. While, it is useful for an intraday trader to predict the stock prices according to his requirement and book profits. A stock market prediction model is designed to be user friendly within a few features affecting the stock market for the first prototype of the model. The model is trained to be more effective with the real time data and provides appropriate output.

Chapter 2: Literature Survey

Paper 1 - Forecasting stock market trends using support vector regression and perceptually important points.

Citation – Azimifar, M., Araabi, B. N., & Moradi, H. (2020). *Forecasting stock market trends using support vector regression and perceptually important points. 2020 10th International Conference on Computer and Knowledge Engineering (ICCKE).* doi:10.1109/iccke50421.2020.9303667.

Intelligent stock trading systems use soft computing techniques in order to make trading decisions in the stock market. However, the fluctuations of the stock price make it difficult for the trading system to discover the underlying trends. In order to enable the trading system for trend prediction, this paper suggests using perceptually important points as a turning point prediction framework. Perceptually important points are utilized as a highlevel representation for the stock price time series to decompose the price into several segments of uptrends and downtrends and define a trading signal which is an indicator of the current trend. A support vector regression model is trained on this high-level data to make trading decisions based on predicted trading signal. The performance of the proposed trading system is compared with three other trading systems on five of the top performing stocks in Tehran Stock Exchange, and obtained results show a significant improvement.

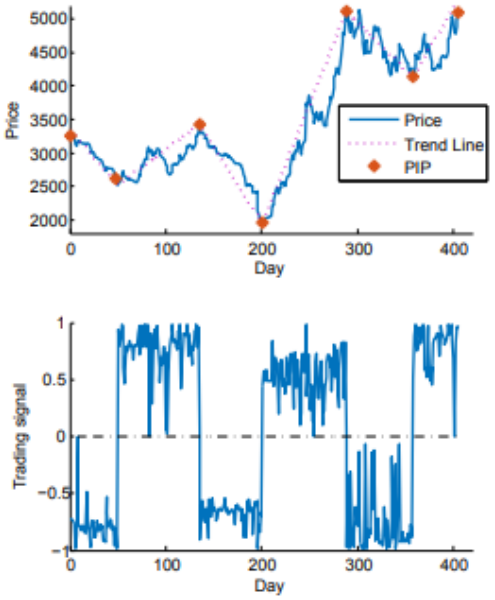


Fig. 1. Trading signal generation: price history, PIPs and the trend lines are shown in the upper figure. The lower figure shows the generated trading signal

TABLE I. TECHNICAL INDICATORS

Technical indicator	Formula
Exponential moving average (EMA)	$\mathrm{EMA}(c_t, n) = \frac{2}{n+1}\{c_t - \mathrm{EMA}(c_{t-1}, n)\} + \mathrm{EMA}(c_{t-1}, n)$
Moving average convergence and divergence (MACD)	$\mathrm{MACD}(c_t, m, n) = \mathrm{EMA}(c_t, n) - \mathrm{EMA}(c_t, m)$
Relative strength index (RSI)	$\mathrm{RSI}(n) = 100 - \frac{100}{1 + \frac{\mathrm{EMA(Up}, n)}{\mathrm{EMA(Down}, n)}}$
Stochastic K%	$K\%(n) = \frac{C_t - L_{t-n}}{HH_{t-n} - LL_{t-n}}$
Stochastic D%	$D\%(n) = \mathrm{EMA}(K, n)$
True strength index (TSI)	$\mathrm{TSI}(n, m) = \frac{\mathrm{EMA}(\mathrm{EMA}(r_t, n), m)}{\mathrm{EMA}(\mathrm{EMA}(\lvert r_t \rvert, n), m)}$

Paper 2 - Stock Market Predication Using A Linear Regression.

Citation - Bhuriya, D., Kaushal, G., Sharma, A., & Singh, U. (2017). *Stock market predication using a linear regression. 2017 International Conference of Electronics, Communication and Aerospace Technology (ICECA).* doi:10.1109/iceca.2017.8212716.

It is a serious challenge for investors and corporate stockholders to forecast the daily behavior of stock market which helps them to invest with more confidence by taking risks and fluctuations into consideration. In this paper, by applying linear regression for forecasting behavior of TCS data set, we prove that our proposed method is best to compare the other regression techniquemethod and the stockholders can invest confidentially based on that. In stock market the most growing topic in research is "Forecasting"because of its importance and popularity among the masses and also small and large companies due to financial benefits and its low risk. Despite the risk of falling too much value per share due to market fluctuations rarely happens, but again, the risk is there. These fluctuations which effect on stock price and trading volume have some difficulties in forecasting. The fluctuations effect on the behavior of people in terms of capital savings or investment, the stock price and the increase or decrease of risk for investors. Therefore, in general, forecasting the stock market behavior through techniques and various methods is a useful tool to assist investors to act with greater certainty and taking the risks and volatility of an investment into consideration and know when to buy the cheapest price and when to sell to highest price [2]. In various fields such as economy, policy and engineering Data mining techniques have a more successful rate compare to traditional statistical method by discovering hidden knowledge of data [3, 4, and 5].Experience has shown that machine learning techniques could be successful in forecasting daily stock price and its trading volume [6]. In this paper, at first review the prior researches done for forecasting stock market and then we describe the importance of trading volume. By using linear regression, we forecastTCS data set [7] behavior and at the end we compared and evaluated the result of our proposed method with other approaches.

Table [1] Dataset Sample (Tcs stock dataset)

Date	Open price	High price	Low price	Close price	Number of trend
8/25/2004	269	270	244	246	26145236
8/26/2004	248	249	249	244	8572064
8/27/2004	245	245	245	239	6013268
8/30/2004	241	247	247	241	4499276

Table 3 Predicted Result

Date	Close price
11/17/2016	2142
11/18/2016	2165
11/19/2016	2178
11/20/2016	2135
11/21/2016	2171

Result analysis of various regression methods

Regression Method	Confidence value
Linear regression	0.9774
Polynomial	0.468
RBF	0.5652

Table 2. Comparison of various regression methods.

Paper 3 - Testing the Application of Support Vector Machine (SVM) to Technical Trading Rules

Citation - Fonseca, A. R., Leles, M. C. R., Moreira, M. G., Vale-Cardoso, A. S., Pereira, M. V. L., Sbruzzi, E. F., & Nascimento, C. L. (2021). *Testing the Application of Support Vector Machine (SVM) to Technical Trading Rules. 2021 IEEE International Systems Conference (SysCon).* doi:10.1109/syscon48628.2021.9447068.

The stock price movements result from many factors that are often difficult to be detected and modelled. The investigation of price trends and the use of the information available to evaluate investments and identify trading opportunities can be promising. However, financial data are non-stationary, i.e., their statistical characteristics constantly change. Therefore, the financial market is a challenging environment for the application of Machine Learning techniques, since they can only make reliable predictions to data consistent with what they have seen before. This paper tests the use of a Machine Learning technique known as Support Vector Machines (SVM) aiming at being a tool to support the decision-making process of trading at the stock market. SVM aggregates some input signals and, based on a set of technical indicators and historical price changes, create buy/sell recommendations of a given security as outputs. The dataset comprises several Brazilian stocks time-series traded on both the Brazilian (B3) and American (NYSE) stock exchange. These time-series belong to various economic sectors and present different market dynamics. The computational simulations are based on a fictitious strategy that does not consider the trading costs and only long positions are allowed. Using two risk-adjusted performance metrics, the results show that strategies based on the SVM model achieve better performance than the Buy & Hold benchmark. Support Vector Machine - SVM SVM is a set of supervised machine learning methods proposed by Cortes and Vapnik [15]. These methods use a linear model to implement a class separation through a nonlinear mapping of the input data sets into a high-dimensional space. It is used in situations that require pattern recognition and classification by data analysis, being considered a learning model based on instances, which seeks to use the similarities between those instances observed in the training data through the features to make a prediction with unseen data model (test data). Hence, this is a memory-based learning method. It can also be extended to regression problems [16, 17] which is based on calculating a regression function. SVM can be explained from the definition of the maximum margin classifier. The data used in the classification problem form a multidimensional space, as the example shown in Equation 1. w0 + (w1X1) + ... + (wnXn) = w0 + N i=1 wixi = y (1) where x = (X1, Xn) is an input data vector, w is the normal vector/weights of the hyperplane, y ∈ {−1,1} is a data set with two classes and N the number of samples used in the model

TABLE I: SVM Model: Features

Feature	Period (days)
RSI	15
SMA_{price}	6; 14; 20
SMA_{volume}	6; 14; 20
SAR	–
ADX	14
r_{log}	1; 5; 10; 20

$$y = \begin{cases} -1, & \text{if price}_{t+1} < \text{price}_t \rightarrow \text{sell} \\ +1, & \text{if price}_{t+1} \geq \text{price}_t \rightarrow \text{buy} \end{cases}$$

Paper 4 – Research on Strategy of HS300 Index Based on Random Forest.

Citation - Huang, X. (202*0). Research on Strategy of HS300 Index Based on Random Forest. 2020 International Conference on Computing and Data Science (CDS). https://doi.org/10.1109/cds49703.2020.00038.*

With the development of China's economy and the continuous improvement of the stock market, stock investment has gradually become one of the ways for people in financial management. It also attracts non-mathematicians to participate in the prediction of stock prices. Models such as machine learning in artificial intelligence are beginning to be widely used in stock forecasting to predict stock trends. This paper calculated the 11 technical indicators of the HS300 index, and used the Random Forest model to predict the movements. The results showed that the accuracy of the model calculation to 0.82 and the F-score is 0.84. Then, the daily income of the model was integrated. Then annualized return reached 28.1%, and the maximum drawdown was 0.42, which was significantly higher than the index return. The results showed that the Random Forest model adopted in this paper is feasible. With the introduction of more and more machine learning methods, artificial intelligence will be further developed in the field of quantitative investment.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (6)$$

The precision measures the classification accuracy of positive samples, and the calculation formula is:

$$Precision = \frac{TP}{TP + FP} \quad (7)$$

The recall indicates the proportion of positive examples in the sample that are correctly predicted. The calculation formula is:

$$Recall = \frac{TP}{TP + FN} \quad (8)$$

F-score is the harmonic average of the recall rate and accuracy. The formula is:

$$F_1 = \frac{2PR}{P + R} = \frac{2TP}{2TP + FP + TN} \quad (9)$$

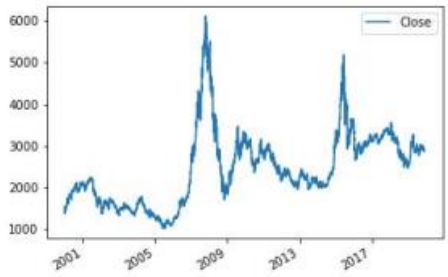


Fig. 1. Changes in the HS300 Index in the past two decades

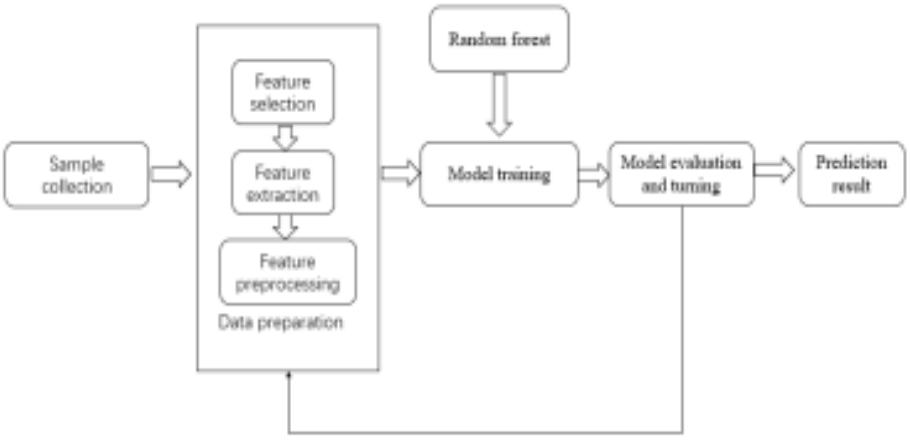


Fig. 3. Strategy diagram

Paper 5 - CUDA parallel computing framework for stock market prediction using K-means clustering.

Citation - Kumari, S., Patil, N., Nankar, P., & Kulkarni, M. (2020). *CUDA parallel computing framework for stock market prediction using K-means clustering. 2020 International Conference on Smart Electronics and Communication (ICOSEC).* doi:10.1109/icosec49089.2020.9215377.

With the recent advancement of machine learning algorithms over past two decades, it has gained a lot of attention of academicians and researchers community and also found prominent application in multiple domain including finance. Our proposed model uses Machine Learning algorithms along with Parallel Computing processing to provide a new trading technique for non-stationary and multidimensional financial data of Reliance Industries retrieved from Dematerialized account from upSTOX Application Programming Interface (API) which extract data at regular interval of 10 minutes. This proposed model uses Kmeans machine learning technique to models the gathered stock data and predicts the upcoming stock values well in advance with parallel processing techniques. This paper provides the analysis of previous year's stock market pricing data and interpret results after performing intensive training based on machine learning algorithm on Compute Unified Device Architecture (CUDA) considering the time constraint of real time trading. The performance of the system is improved drastically with the help of machine learning techniques and to accelerate the process of generating the results, a technique of parallel computing is used in this paper. The performance time is significantly reduced because of high performance speed using CUDA parallel computing technology compared to traditional methods of single Central Processing Unit (CPU). It helped in reduction of calculation time by large margin and hence to gain book profit which is the ultimate goal of trading by predicting the stock values well in advance. Investors can decide whether to keep that stock, sell it or buy some other new stocks or neutral decision on basis of 3 clusters as predicted k means algorithm. Neutral decision means if user owns stock then he should hold that stock with him and if he does not possess stock then he should not buy it. This proposed method is suitable for intraday trading as the stocks value are taken for each 10 minutes on the basis of that model can take decision what should do.Trend Indicators Every trend indicator tells us in which direction the market is moving, if there is a trend in the market. And by using those indicators, trends in the market can be easily detected. There are many trend indicators but in this paper's scope consider only the below 4 indicators. i) Average Directional Index (ADX) ii) Simple Moving Average (SMA) iii) Weighted Moving Average (WMA) iv) Exponential Moving Average (EMA).

b. Simple Moving Average:

Simple Moving Average (SMA) is a technical indicator that tells us whether stock price will continue or reverse the bear or bull trend. For calculation of SMA, the closing prices of stock are generally considered. To classify the trend, k-means clustering algorithm is used.

$$SMA_n = \frac{\sum closing\ price\ in\ past\ n\ days}{n} \quad (5)$$

Where n is number of days. The "n" is considered as any value, but generally it is considered as 14 as default value.

d. Exponential Moving Average(EMA) :

Exponential Moving Average (EMA) also places more weight and significance on recent values like WMA.

$$EMA_{(t1)} = EMA_{(t0)} + \alpha \cdot (p - EMA_{(t0)}) \quad (7)$$

$$\alpha = \frac{2}{(N+1)} \quad (8)$$

Where EMA $_{(t0)}$ is Exponential Moving Average of previous time period, α is Exponential smoothing constant, p is Current price. This is an index smooth moving average price over a period of time.

Paper 6 - Optimization of SV-kNNC using Silhouette Coefficient and LMKNN for Stock Price Prediction.

Citation - Mikael Sinaga, F., Sirait, P., & Halim, A. (2020). *Optimization of SV-kNNC using Silhouette Coefficient and LMKNN for Stock Price Prediction. 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI).* doi:10.1109/isriti51436.2020.9315516.

A broker often finds it difficult to decide to buy or sell shares. Due to the large amount of stock data and errors in analyzing stock indicators, it can cause the broker to suffer losses. Predicting stock prices with high accuracy becomes onerous. Although unnecessary variables have been excluded but there are still outliers. It is still difficult to attain the most optimal class value at the data clustering stage. The objective of this research is to propose a model to refine the accuracy of stock prediction and to produce the best class at the data clustering stage. The proposed model is named SV-kNNC - Silhouette Coefficient - SOM (Self Organizing Map) - LMKNN (Local Mean-based K Nearest Neighbor). First, the best class value is obtained using the Silhouette Coefficient, then the data is weighted, finally the stock data is classified using the LMKNN to secure the prediction results. In the testing section, SV-kNNC - SOM + Silhouette Coefficient - LMKNN is compared against SV-kNNC - SOM - KNN (K Nearest Neighbor). Confusion Matrix is used during the test to get the predictive accuracy value. It can be seen from the results of the prediction accuracy of BBCA's shares is 86.99% with the best K is 3, ASII company is 78.30% with the best K is 3, TLKM company is 85.10% with the best K is 5, BBRI company is 84.73% with the best K is 6, PGAS company is 71.36 % with the best K is 5. SV-KNNC is an algorithm consisting of SVM, K-Means, and KNN. The clustering output in this research still needs improvement because the performance of K-Means is unable to produce efficient data clusters which can be seen from the poor accuracy value obtained from several tests. SV-KNNC uses the Support Vector Machine (SVM) to obtain support vector (SV) from training data located near the hyperplane to classify test data. Therefore, the SV obtained is an ideal representation of training data [6]. A re-examination of the SV contribution on the classification is done in the initial dimensions before using it to perform the test data. Weights are used to determine the contribution of each SV. The instance (SV) with a higher weight is more reliable and has a greater influence on the KNN [10]. In the SV-kNNC and SOM models, comparison is carried out at the data clustering stage. From the results of this research, SOM is better for data clustering. The weakness of this research is that it is still difficult to determine the best class at the data clustering stage. Additionally, the prediction results produced are not optimal [6]. Before Support Vector is clustered, the best class is determined using the Silhouette Coefficient. Then this best class is used for the SOM process in data clustering. After the data is weighted, LMKNN is used for clustering in order to increase the accuracy of stock predictions. The proposed model can obtain the best K parameter at the

data clustering stage. Therefore, the stock prediction results can be better.

Step 1: Determination of K Value

Step 2: Calculate the distance between the test data and eac of the training data using the Euclidean distance

$$D(x,y) = ||x-y||_2 = \sqrt{\sum_{j=1}^{N} |x-y|^2} \quad (7)$$

Step 3: Sort data distances from smallest to largest as many a K for each data class

Step 4: Calculate the local mean vector of each class

$$w_c = argmin_{wj}\left(x, m_{w_j}^k\right), j = 1,2,\dots,M \quad (8)$$

Step 5: Determine the test data class by calculating the closes distance to the local mean vector of each data class

$$m_{w_j}^k = \frac{i}{k}\sum_{i=1}^{k} yi^N, j^N \quad (9)$$

Paper 7 – Stock Market Prediction Using Linear Regression and SVM.

Citation - Panwar, B., Dhuriya, G., Johri, P., Singh Yadav, S., & Gaur, N. (2021). *Stock Market Prediction Using Linear Regression and SVM. 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE).* doi:10.1109/icacite51222.2021.9404733.

In Stock Market is the financial epitome of financial business and trading since it came into existence it has shown the impact of hits low and similarly when it is high. The stock market crash in 2008 showed the world that the business hit the low when the Dow Jones Industrial Average fell 777.68%. Several machine learning algorithms have shown that these stock prices can be predicted and these algorithms can be implemented using the approach of supervised learning. In Supervised Learning, we have test data using this we train the models. Although the results obtained after training the model may differ from the actual but it has been observed that in many cases accuracy is satisfactory. In this paper, the first task is to use web scrapping to collect datasets from stock data. Then we plot the data on the graph, from the graph we can analyze the stock prices going high or low. After this, we will predict stock prices using SVM and Linear Regression, that Linear Regression for stock market analysis is better than the SVM for the same.

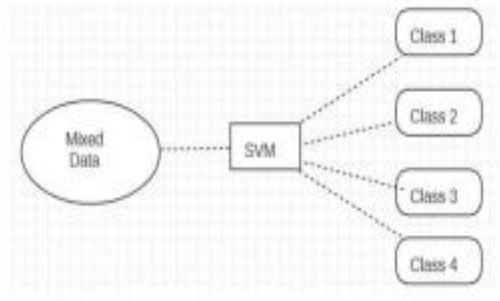


Fig. 2: A SVM Categorization

For SVM we first draw the planes intersecting values of the circles and rectangles and using those we develop the median of both the planes this results in another name for SVM that is the widest street approach.[3] The figure below describes hyperplane[fig3]

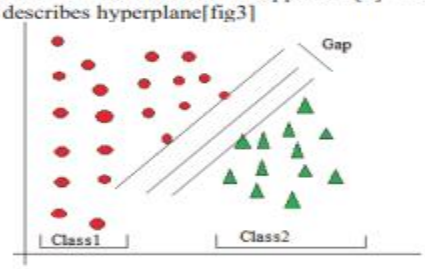


Fig. 3: Hyperplane dividing classes

A. SVM Architecture

In SVM we analyze the two classes for dividing them with a hyperplane, it is a line that splits the data of the two classes such that value between the last points of the classes is maximum it is called maximum margin.[5] Now using the hyperplane and the support vectors we achieve the widest street, which is in further enhance towards SVM approach [fig4].

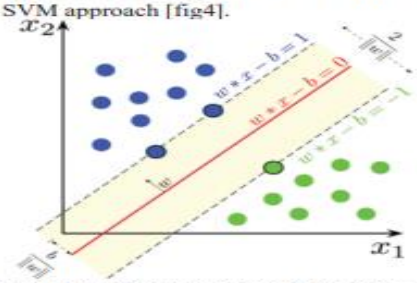


Fig. 4: SVM Architecture In the above figure

Paper 8 - Prediction of Stock Prices using Machine Learning (Regression,Classification) Algorithms.

Citation - Ravikumar, S., & Saraf, P. (2020). *Prediction of Stock Prices using Machine Learning (Regression, Classification) Algorithms. 2020 International Conference for Emerging Technology (INCET).* doi:10.1109/incet49848.2020.9154061.

The stock market is an interesting industry to study. There are various variations present in it. Many experts have been studying and researching on the various trends that the stock market goes through. One of the major studies has been the attempt to predict the stock prices of various companies based on historical data. Prediction of stock prices will greatly help people to understand where and how to invest so that the risk of losing money is minimized. This application can also be used by companies during their Initial Public Offering (IPO) to know what value to target for and how many shares they should release. So far there have been significant developments in this field. Many researchers are looking at machine learning and deep learning as possible ways to predict stock prices. The proposed system works in two methods – Regression and Classification. In regression, the system predicts the closing price of stock of a company, and in classification, the system predicts whether the closing price of stock will increase or decrease the next day.

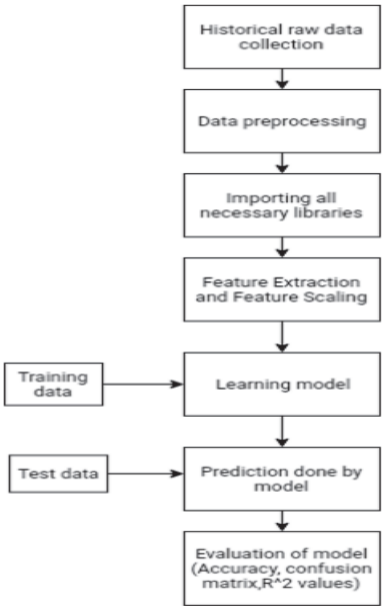
TABLE I. RESULT ANALYSIS OF REGRESSION MODELS

Model	Accuracy	Time(in seconds)
Simple Linear Regression	81.52	0.77
Polynomial Regression	91.45	0.98
Support Vector Regression (SVR)	87.41	1.16
Decision Tree Regression	98.09	0.79
Random Forest Regression	99.57	1.06

B. Classification

TABLE II. RESULT ANALYSIS OF CLASSIFICATION MODELS

Model	Acc.	Time(s)
Support Vector Machine (linear)	68.41	158.48
Support Vector Machine (poly)	64.80	195.38
Support Vector Machine (rbf)	67.86	201.15
Support Vector Machine (sigmoid)	58.65	160.81
K – Nearest Neighbors	61.50	19.02
Logistic Regression	68.27	10.51
Naïve Bayes	67.10	10.14
Decision Tree Classification	57.99	198.57
Random Forest Classification	63.33	202.54



Paper 9 - Research and Comparison of Random Forests and Neural Networks in Shanghai and Shenzhen Financial 20 Index Prediction.

Citation - Su, C.-Y., Lei, Y., & Wang, M.-X. (2021). *Research and Comparison of Random Forests and Neural Networks in Shanghai and Shenzhen Financial 20 Index Prediction. 2021 World Conference on Computing and Communication Technologies (WCCCT).* doi:10.1109/wccct52091.2021.00023.

Machine learning has made great achievements in the field o f a rtificial in telligence, es pecially in th e financial industry, which has shown great potential and attracted the attention of the academia and the industry. With the opening of the capital market to the outside world, how to achieve more accurate forecast of the trend of stock index and timely carry out reasonable risk control is one of the important issues concerned by investors. On the basis of literature and theoretical analysis, the public data sets of a well-known securities firm i s selected to make predictions of the Shanghai and Shenzhen CSI 20 index based on neural networks and random forest models. The average absolute error (MAE) was then used as a metric to compare the performance of the two machine learning algorithms on the stock index prediction problem. The results show that the model constructed by the random forest algorithm performs better on this problem. Therefore, we believe that making investment decisions based on the results can help investors make effective investment decisions under certain risks.

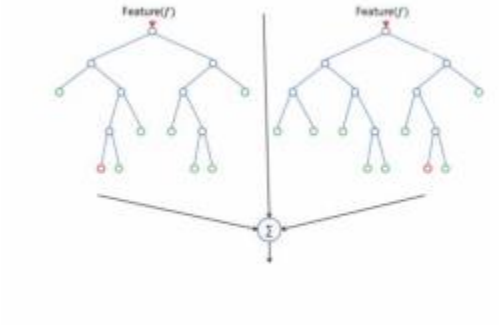


Fig. 1: Random forests model

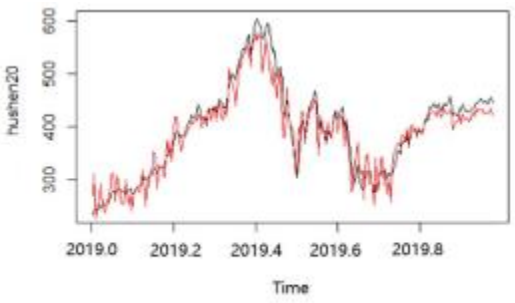


Fig. 5: Random forest results

Random forests Show in figure 1,if each ensemble classifier hk(x) is a decision tree, then the collection is a random forest. We define hk(x) the parameters of the decision tree for the classifier Θk = (θk1, θk2,...,θkp)(these parameters include the structure of the tree, the nodes to which variables are split, etc.). We can also write it as hk(x) = h (x | Θk). Fig. 1: Random forests model The question is, how do we choose which features appear in which nodes of the tree kth.In a random forest, factors Θk are chosen randomly based on the variables Θ being modeled. A random forest is a family of classifiers h (x | Θ1),...,h (x | ΘK) based on a classification tree Θk with one parameter, chosen randomly from the model random vectors Θ. For the final classification f(x)(combined with the classifier {hk(x)}), the random forest takes the plural or average of these results as the output of this new data. The formula is as follows D = {(xi, yi)} n i=1. The process of building a random forest is shown in the figure. Compared with many algorithms, the learning process for random forests is very fast.

Paper 10 - Research and prediction of Shanghai-Shenzhen 20 Index Based on the Support Vector Machine Model and Gradient Boosting Regression Tree.

Citation - Pingwen Xue, Yuan Lei, Yanxing Li. (2020). *Research and prediction of Shanghai-Shenzhen 20 Index Based on the Support Vector Machine Model and Gradient Boosting Regression Tree. 2020 International Conference on Intelligent Computing, Automation and Systems (ICICAS).* doi: 10.1109/icicas51530.2020.00019.

The non-stationarity, non-linearity and multi-factor influence of financial index series makes it more difficult to predict. In order to achieve a more accurate stock index prediction, this paper selects a public data set provided by a well-known securities company to predict the CSI 20 Index (CSI20) based on the gradient-enhancing regression tree model and support vector machine model. RMSE, MAE, and MAPE are selected as the indicators to evaluate the differences in the forecasting ability of the two models in the financial sector. The results of this study can help investors to adopt effective investment strategies and reduce investment risks. Support vector machine According to the calculation rules of stock index mentioned above, the calculation of stock index is a regression problem. However, after literature review, we find that there are many factors that need to be considered in the calculation of stock index, so the problem of stock index calculation is actually a nonlinear regression problem. The

following will introduce the theory of support vector machine model under nonlinear conditions. For nonlinear problems, support vector machine (SVM) maps the sample space into the high-dimensional feature space (Hilbert space) by selecting appropriate nonlinear mapping, and carries out linear regression in this space. Thus, the nonlinear regression problem in the low dimensional space is transformed into the linear regression problem in the high dimensional space. Support vector machine (SVM) is as follows.

$$\min_{w,b,\zeta} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i$$
$$st \quad y_i\left(w\cdot\phi\left(x_i\right)+b\right) \geq 1-\xi_i \quad \xi_i \geq 0 \quad i=1,2,\ldots,N$$

According to the calculation, we can get the classification decision function in the following form.

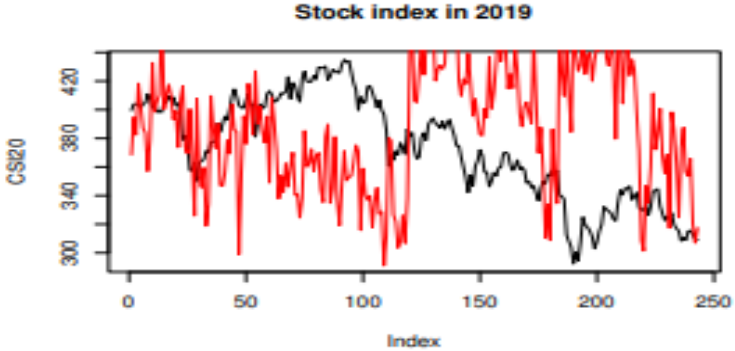
$$f(x) = \operatorname{sign}\left(\sum_{i=1}^{N} a_i^* y_i \left(\phi(x)\cdot\phi\left(x_i\right)\right) + b^*\right)$$

The common kernel functions include polynomial kernel function, Gaussian radial basis function (RBF) kernel function and sigmoid kernel function.

$$K(x,z) = (< x,z > +R)^d$$

$$K(x,z) = \exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right)$$

$$\mathrm{K}(x,z) = \tanh(v(x,z)+c)$$



Paper 11 - The Public Sentiment analysis within Big data Distributed system for Stock market prediction– A case study on Colombo Stock Exchange.

Citation - Malawana, M. V. D. H. ., & Rathnayaka, R. M. K. T. (2020). *The Public Sentiment analysis within Big data Distributed system for Stock market prediction– A case study on Colombo Stock Exchange. 2020 5th International Conference on Information Technology Research (ICITR).* doi:10.1109/icitr51448.2020.9310871.

Stock price prediction plays an important role on the journey of investors on the stock market. The prices of the company stocks on the market are performed by different deliverables. Social media data sets, news sites, feedback and reviews are some kind of online tools that can affect the stock market. It is often worth using this context to predict the performance of market shares. We take the advantage of Sentiment analysis on Market related announcement and respective public opinions for stock market trend predictions for more accurate recommendations. Sentiment Analysis is a machine learning program for extracting opinions from a text section that is designed to support any product, company, individual or other entity (positive, negatively, neutral). In this research calculations and data processing were performed within machine learning approach with use of Spark model on Google cloud platform. Among most of the stock prediction researches, only few researchers have done their researches on sentiment analysis within big data distributed environment. Logistic Regression and Naïve Bayes perform well in sentiment classification. Main finding of this research is that public opinion significantly influences the fluctuations of market forces and economic factors such as monetarism, government reforms, unforeseen pandemics, interest rates, public trust, and faith in bond market trust. The detection of the feelings pattern will enhance the market prediction as it ensures the consistency of decision.

Chapter 3: Methodology/Techniques or Algorithm Used

- After analyzing the Nifty50 dataset and datasets containing the details of stocks listed in Nifty50 which are segregated based on their market capital, exhaustive data cleaning is carried out with the help of python libraries like NumPy, Pandas, Matplotlib.
- We convert categorical to numerical attributes using Scikit Learn The output sparse matrix, we convert this sparse matrix to NumPy array. After the data cleaning part of work, we ended up with results.
- Rescaling of the records is an important element to reduce the error. We split the dataset into training and testing dataset by using Sklearn's package.
- Here we are using,
    a. Pandas.
    b. Linear Regression.
    c. Naive bayes (Linear Classifier)
    d. Logistic Regression (Linear Classifier)
    e. Support Vector Machine
    f. K-NEAREST NEIGHBOUR (KNN)
    g. Random Forest
    h. K-means clustering (Elbow curve method)

  Using the above mentions algorithms and specified techniques involving regression, classification, clustering and machine learning packages, we deduce and infer the predictions of particular stock's performance in the stock market.

Chapter 4: Tools Used

1. Python :
   Python programming language to analyze the stock market interactively. Stock Market Analysis means analyzing the current and historical trends in the stock market to make future buying and selling decisions.

2. Sklearn :
   Scikit-learn is the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

3. Pandas :
   Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with relational or labeled data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python.

4. Matplotlib :
   Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy.

5. Jupyter Notebook :
   Jupyter notebook is a very popular and flexible development environment which lets us write and execute python code, display the output and any kind of visualization or plot, etc. in the same document. however, it is always advisable to install Python with Anaconda environment on your system as well.

6. Seaborn :
   Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures. Seaborn helps you explore and understand your data.

7. Numpy :
   NumPy is the package for scientific computing in Python. NumPy arrays eases advanced mathematical and other types of operations on large numbers of data.

Chapter 5: Modules Implemented and Output

Module 1 – Pandas (Data Preprocessing – Cleaning and Transformation).

In Module 1, you are going to get familiar with pandas, the python module which is used to process and analyse data. Processing could include removing unknown values from the data or replacing unknown values with values which make sense, maybe 0. Analysing the data could include finding out the trend of a stock price, e.g. how the stock price changes with respect to the Nifty 50 basket of stocks. This modules involves importing CSV file using pd.read_csv(), Check whether csv file has data on more than one category. Remove all the rows where 'Series' column is NOT 'EQ'. Analyze and understand stock_data DataFrame properly. Calculate the maximum, minimum and mean price for the last 90 days. (price=Closing Price unless stated otherwise). Analyse the data types for each column of the "stock_data". Changing the date column from 'object' type to 'datetime64(ns)'.

1.1 Subtract the minimum value of the date column from the maximum value.

Here

- "`Total Traded Quantity`" => Volume
- "`Average Price`" => Price
- "`Turnover`" => Price * Volume
  - Product of total volume traded to the average price of a stock for the day.

Let implement VWAP formula

$$VWAP = \frac{\sum price * volume}{\sum volume}$$

$$VWAP = \frac{\sum Turnover}{\sum TotalTradedQuantity}$$

1.2 Write a second function to calculate the profit/loss percentage over the last N days.

$$Profit/LossPercentage = \frac{ClosePrice - PrevClose}{PrevClose} * 100$$

1.3 Calculate the Average Price and Profit/Loss Percentages over the course of last :

- 1 week,
- 2 weeks,
- 1 month,
- 3 months,

- 6 months,
- 1 year.

1.4 Add a column 'Day_Perc_Change' where the values are the daily change in percentages i.e. the percentage change between 2 consecutive day's closing prices. Instead of using the basic mathematical formula for computing the same, use 'pct_change()' function provided by Pandas for dataframes. You will note that the first entry of the column will have a 'Nan' value. Why does this happen? Either remove the first row, or set the entry to 0 before proceeding.

1.5 Add another column 'Trend' whose values are:

- **'Slight or No change'** for 'Day_Perc_Change' in between -0.5 and 0.5
- **'Slight positive'** for 'Day_Perc_Change' in between 0.5 and 1
- **'Slight negative'** for 'Day_Perc_Change' in between -0.5 and -1
- **'Positive'** for 'Day_Perc_Change' in between 1 and 3
- **'Negative'** for 'Day_Perc_Change' in between -1 and -3
- **'Among top gainers'** for 'Day_Perc_Change' in between 3 and 7
- **'Among top losers'** for 'Day_Perc_Change' in between -3 and -7
- **'Bull run'** for 'Day_Perc_Change' >7
- **'Bear drop'** for 'Day_Perc_Change' <-7

Out[48]:

	Day_Perc_Change	Trend
0	0.000000	Slight or No change
1	0.362566	Slight or No change
2	-0.230366	Slight or No change
3	0.939337	Slight positive
4	-0.395113	Slight or No change
5	0.365364	Slight or No change
6	-0.431640	Slight or No change
7	-0.261151	Slight or No change
8	2.984918	Positive
9	1.261060	Positive

1.6 Find the average and median values of the column 'Total Traded Quantity' for each of the types of 'Trend'.
{Hint : use 'groupby()' on the 'Trend' column and then calculate the average and median values of the column 'Total Traded Quantity'}

```
avg_median_trendwise = stock_data.groupby(by="Trend")["Total Traded Quantity"].agg(['mean','median']).rename(columns={'mean':'average'})
avg_median_trendwise
```

Trend	average	median
Bear drop	4.878630e+07	48786302.0
Among top losers	1.463854e+07	10985897.0
Negative	5.634151e+06	5088554.0
Slight negative	5.454672e+06	4616803.0
Slight or No change	4.647983e+06	4043539.5
Slight positive	5.848807e+06	5301784.0
Positive	6.655530e+06	5523811.5
Among top gainers	1.259063e+07	12042170.5
Bull run	NaN	NaN

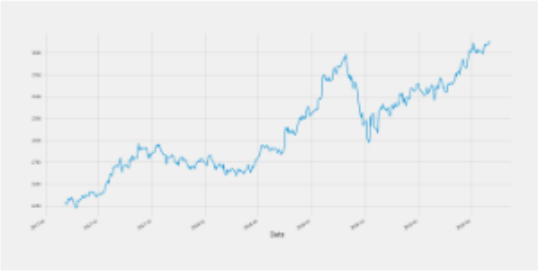
Module 2 – Plotting in Financial Markets (Data Visualization & Technical Analysis).

'A picture speaks a thousand words' has never been truer in financial markets. Absolutely no one goes through the millions of rows of numbers, we always prefer the data in a plotted form to draw better inferences. This module would cover the plotting, basic technical indicators and our own customisation, and making our own trade calls! You should target to finish module 2, including the prerequisites, in 1-2 weeks.

Module 3 – Regression & Beta Calculation.

This module would introduce us to the Regression related inferences to be drawn from the data. Regression is basically a statistical approach to find the relationship between variables. In machine learning, this is used to predict the outcome of an event based on the relationship between variables obtained from the data-set. More often than not, we utilize linear regression to come up with an ideal inference.

- Look out for drastic changes in this stock, you have the exact date when these took place, try to fetch the news for this day of this stock
- This would be helpful if we are to train our model to take NLP inputs.



3.1 Linear Regression

Using linear regression, find the coefficients of the inputs and using the same trained model, complete the entire column.

In [5]:
```
# Missing values in Pred Columns
Pred_NaN_count = len(gold_data[gold_data.Pred.isna()])

# Creating suitable dataset for regression model
dataset = gold_data[['Open', 'High', 'Low', 'Price', 'Pred']].iloc[:-Pred_NaN_count]
X = dataset.iloc[:, :-1]
y = dataset.iloc[:, -1:]

# divding dataset into training and testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# Linear regression model
regressor = LinearRegression()
# Finding best fit line
regressor.fit(X_train, y_train)
```

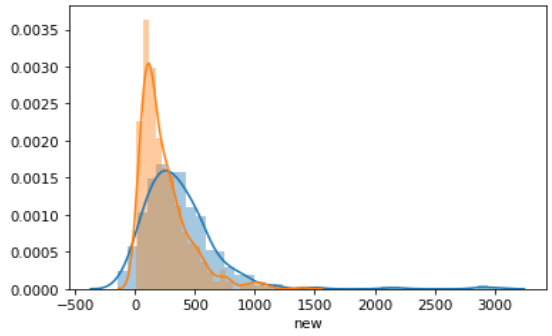
Out[5]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

3.2 Prediction Accuracy

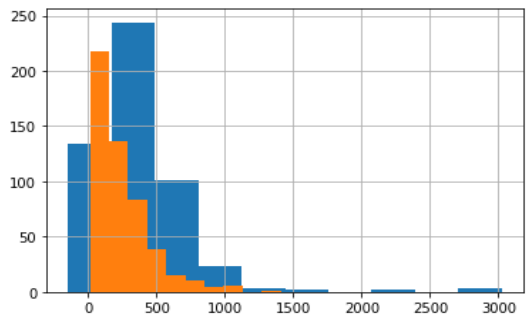
In [10]:
```
y_pred = new_regressor.predict(X_test)
score_new_regressor = r2_score(y_test.values, y_pred)
print("Accuracy Score in case of 'new' column: ", score_new_regressor)
```

```
Accuracy Score in case of 'new' column:  0.9999901742208236
```

In [11]:
```
sns.distplot(gold_data.Pred)
sns.distplot(gold_data.new)
plt.show()
```



In [12]:
```
gold_data.Pred.hist()
gold_data.new.hist()
plt.show()
```



3.3 CAPM Analysis and Beta Calculation using regression

CAPM(Capital Asset Pricing Model) attempts to price securities by examining the relationship that exists between expected returns and risk. The Beta of an asset is a measure of the sensitivity of its returns relative to a market benchmark (usually a market index). How sensitive/insensitive is the returns of an asset to the overall market returns (usually a market index like S&P 500 index).

Import the stock of your choosing AND the Nifty index. Using linear regression (OLS), calculate -

- The daily Beta value for the past 3 months. (Daily= Daily returns)
- The monthly Beta value. (Monthly= Monthly returns)

Refrain from using the (covariance(x,y)/variance(x)) formula. Attempt the question using regression.(Regression Reference) Were the Beta values more or less than 1 ? What if it was negative ? Discuss. Include a brief writeup in the bottom of your jupyter notebook with your inferences from the Beta values and regression results

Steps to Calculate Beta

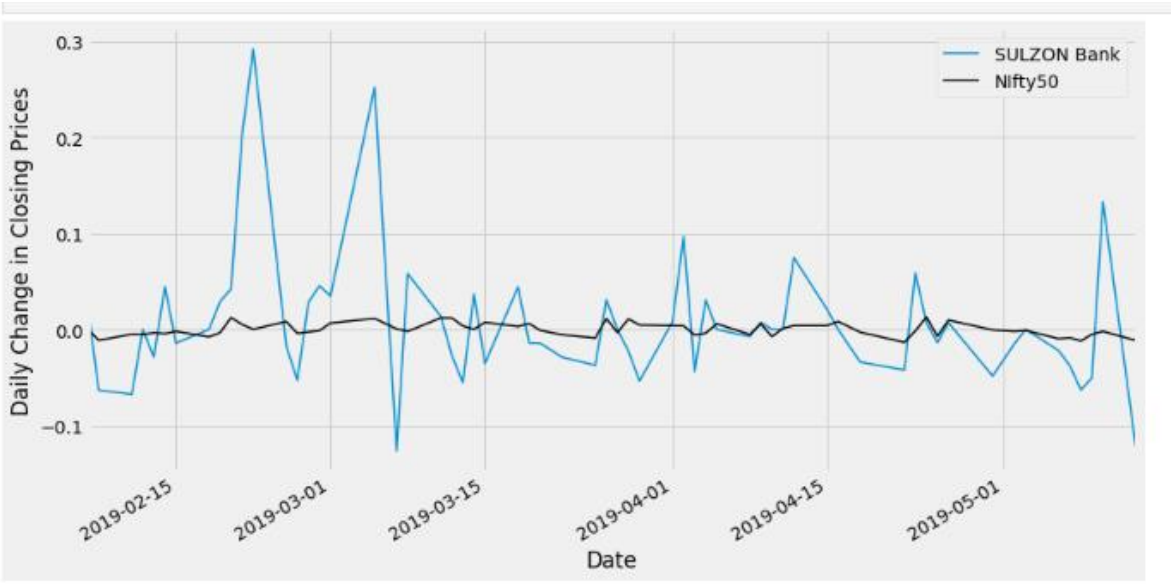
1. Calcute Daily returns of Gold
2. Calcute Daily returns of Nifty

$$DailyReturns = \frac{ClosePrice - Prevprice}{PrevPrice}$$

3.2.1 The daily Beta value for the past 3 months. (Daily= Daily returns)

In [14]:
```
months = 3
days = months * 21

# Reading sulzon Stock and nifty index data
sulzon = pd.read_csv("./stock_data/SUZLON.csv")
nifty = pd.read_csv("./Nifty50.csv")

# Taking SULZON Stock of past 3 months
sulzon_stock = sulzon.tail(days)
sulzon_stock.Date = pd.to_datetime(sulzon_stock.Date)
sulzon_stock = sulzon_stock.set_index('Date')

# Taking Nifty50 index of past 3 months
nifty_stock = nifty.tail(days)
nifty_stock.Date = pd.to_datetime(nifty_stock.Date)
nifty_stock = nifty_stock.set_index('Date')
```

In [15]:
```
# Daily Returns of SULZON stock for last 3 months
sulzon_daily_return = sulzon_stock['Close Price'].pct_change().dropna()

# Daily Returns of Nifty50 index for last 3 months
nifty_daily_return = nifty_stock.Close.pct_change().dropna()
```




```
In [17]: # Reshaping for Regression Purpose
sulzon_daily_return = sulzon_daily_return.ravel().reshape(-1,1)
nifty_daily_return = nifty_daily_return.ravel().reshape(-1,1)
```

3.4 Conclusion

- **Were the Beta values more or less than 1 ?**

Beta Values is more than 1.

1. **What if it was negative ?**

A negative beta correlation would mean an investment that moves in the opposite direction from the stock market. When the market rises, then a negative-beta investment generally falls. When the market falls, then the negative-beta investment will tend to rise.

- **Inferences from the Beta Values and Regression results**
  - Beta measures the volatility of a stock compared with the volatility of the market as a whole.
  - A high beta means the stock price will move faster than a stock with low beta. High beta means high volatility, but also the possibility of high returns.
  - β=0: indicates no correlation with NIFTY or some chosen Index/Benchmark.
  - β=1: shows a stock has equally sensitive as the market.
  - β> 1: indicates a stock that's more volatile/unstable than NIFTY.
  - β< 1: shows less sensitive than NIFTY
  - 1.45 is 45% more sensitive than NIFTY
- **Daily Beta for last 3 months:**

β = 3.4868

Beta is greater than one and greater than zero which means SULZON Stocks are moving in the same direction as NIFTY but at a very high rate. It moving 3 times fast than the market.

- **Monthly Beta Value:**

beta=2.7005 (β>1)
A beta is greater than one which means SULZON Stocks are moving in the same direction as NIFTY and at a higher rate. It moving 2 times fast than the market.
It shows SULZON stock is very volatile.
It very nice option for Intraday trading but not for long term trading.
Instead of investing all money in one stock we can diversify the stocks.
Due to high volatility we can get a very big return on investment by using the CAPM model.

Module 4 – Algorithm Trading using Classifications (Trade Call Prediction)

In this module, we'd be covering the concept of classification and utilize our skills to solve the following queries – (Stock Price = Close Price)

4.1 Create new column “Call” and adding Bollinger Bands Columns
Create a new column 'Call' , whose entries are -

- 'Buy' if the stock price is below the lower Bollinger band
- 'Hold Buy/ Liquidate Short' if the stock price is between the lower and middle Bollinger band
- 'Hold Short/ Liquidate Buy' if the stock price is between the middle and upper Bollinger band
- 'Short' if the stock price is above the upper Bollinger band

In [2]:
```
stock_data = pd.read_csv("./week2.csv",index_col=0)
stock_data.Date = pd.to_datetime(stock_data.Date)
stock_data = stock_data.set_index('Date')
stock_data.head()
```

Out[2]:

Date	Symbol	Series	Prev Close	Open Price	High Price	Low Price	Last Price	Close Price	Average Price	Total Traded Quantity	Turnover	No. of Trades	Deliverable Qty	% Dly Qt to Traded Qty	Year	Month	Day_Perc_Change	Tre
2017-05-15	INFY	EQ	964.25	963.5	963.50	949.10	953.25	951.55	951.49	3648582	3.471580e+09	75335	3052819	83.67	2017	5	0.000000	Slig or l chan
2017-05-16	INFY	EQ	951.55	953.1	960.15	946.95	956.00	955.00	952.92	3065084	2.920775e+09	71808	1858063	60.62	2017	5	0.362566	Slig or l chan
2017-05-17	INFY	EQ	955.00	951.6	958.45	943.85	952.90	952.80	949.48	1457754	1.384110e+09	75429	792251	54.35	2017	5	-0.230366	Slig or l chan
2017-05-18	INFY	EQ	952.80	943.0	973.90	942.85	960.25	961.75	962.61	4028924	3.878282e+09	120990	2309450	57.32	2017	5	0.939337	Slig posit
2017-05-19	INFY	EQ	961.75	961.5	962.70	947.85	957.40	957.95	954.18	2128698	2.031155e+09	88897	1457747	68.48	2017	5	-0.395113	Slig or l chan

In [3]:
```
rolling_avg = stock_data['Close Price'].rolling(window=14).mean()
rolling_std = stock_data['Close Price'].rolling(window=14).std()
```

In [4]:
```
stock_data['lower'] = rolling_avg-2*rolling_std
stock_data['avg'] = rolling_avg
stock_data['upper'] = rolling_avg+2*rolling_std
stock_data.dropna(inplace=True)
stock_data.head()
```

Out[4]:

Date	Symbol	Series	Prev Close	Open Price	High Price	Low Price	Last Price	Close Price	Average Price	Total Traded Quantity	...	No. of Trades	Deliverable Qty	% Dly Qt to Traded Qty	Year	Month	Day_Perc_Change	Trend	I
2017-06-01	INFY	EQ	977.05	969.3	979.70	958.55	971.25	971.40	969.49	2754303	...	81421	1822506	66.17	2017	6	-0.578271	Slight negative	936.70
2017-06-02	INFY	EQ	971.40	973.4	975.45	964.20	966.00	969.45	968.76	1958983	...	48927	1396644	71.29	2017	6	-0.200741	Slight or No change	939.54
2017-06-05	INFY	EQ	969.45	970.0	972.00	957.05	959.00	958.75	961.71	2731349	...	83794	1941199	71.07	2017	6	-1.103719	Negative	940.31
2017-06-06	INFY	EQ	958.75	965.0	987.50	964.40	980.00	979.35	980.08	3504343	...	83178	2018029	57.59	2017	6	2.148631	Positive	943.63
2017-06-07	INFY	EQ	979.35	985.0	988.90	945.00	959.30	961.30	963.75	6227523	...	139990	2519343	40.45	2017	6	-1.843059	Negative	943.55

5 rows × 21 columns

4.2 Training different Models –
Different Classification Algorithm :

1. Naive bayes (Linear Classifier)
2. Logistic Regression (Linear Classifier)
3. Support Vector Machine
4. K-NEAREST NEIGHBOUR (KNN)
5. Random Forest

Naive bayes

In [15]:
```
from sklearn.naive_bayes import GaussianNB
gaussian_classifier = GaussianNB()
gaussian_classifier.fit(X_train, y_train)
```

Out[15]:
```
GaussianNB(priors=None, var_smoothing=1e-09)
```

In [16]:
```
training_accuracy = gaussian_classifier.score(X_train, y_train).round(2)
training_accuracy
```

Out[16]:
```
0.58
```

In [17]:
```
testing_accuracy = gaussian_classifier.score(X_test, y_test).round(2)
testing_accuracy
```

Out[17]:
```
0.52
```

In [18]:
```
classifers_train_test.loc[0] = ["Naive Bayes", training_accuracy, testing_accuracy]
```

Logistic Regression

In [19]:
```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(max_iter=10000,class_weight='balanced', multi_class="ovr", solver='liblinear')
lr.fit(X_train, y_train)
```

Out[19]:
```
LogisticRegression(C=1.0, class_weight='balanced', dual=False,
                   fit_intercept=True, intercept_scaling=1, l1_ratio=None,
                   max_iter=10000, multi_class='ovr', n_jobs=None, penalty='l2',
                   random_state=None, solver='liblinear', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [20]:
```
training_accuracy = lr.score(X_train, y_train).round(2)
training_accuracy
```

Out[20]: 0.97

In [21]:
```
testing_accuracy = lr.score(X_test, y_test).round(2)
testing_accuracy
```

Out[21]: 0.98

In [22]:
```
classifers_train_test.loc[1] = ["Logistic Regression", training_accuracy, testing_accuracy]
```

SVM

In [23]:
```
from sklearn.svm import SVC
svm_classifier = SVC(verbose=True, gamma='scale',class_weight='balanced')
svm_classifier.fit(X_train, y_train)
```

```
[LibSVM]
```

Out[23]:
```
SVC(C=1.0, cache_size=200, class_weight='balanced', coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=True)
```

In [24]:
```
training_accuracy = svm_classifier.score(X_train, y_train).round(2)
training_accuracy
```

Out[24]: 0.25

In [25]:
```
testing_accuracy = svm_classifier.score(X_test, y_test).round(2)
testing_accuracy
```

Out[25]: 0.19

In [26]:
```
classifers_train_test.loc[2] = ["SVM", training_accuracy, testing_accuracy]
```

KNN

In [27]:
```
from sklearn.neighbors import KNeighborsClassifier
knn_classifier = KNeighborsClassifier(n_neighbors=3)
knn_classifier.fit(X_train, y_train)
```

Out[27]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=3, p=2,
                     weights='uniform')
```

In [28]:
```
training_accuracy = knn_classifier.score(X_train, y_train).round(2)
training_accuracy
```

Out[28]: 0.89

In [29]:
```
testing_accuracy = knn_classifier.score(X_test, y_test).round(2)
testing_accuracy
```

Out[29]: 0.85

In [30]:
```
classifers_train_test.loc[3] = ["KNN", training_accuracy, testing_accuracy]
```

Random Forest Classifier

```
In [31]: from sklearn.ensemble import RandomForestClassifier
         rf_classifier = RandomForestClassifier(max_depth =3, n_estimators = 10,class_weight='balanced')
         rf_classifier.fit(X_train, y_train)

Out[31]: RandomForestClassifier(bootstrap=True, class_weight='balanced',
                                criterion='gini', max_depth=3, 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=0, warm_start=False)

In [32]: training_accuracy = rf_classifier.score(X_train, y_train).round(2)
         training_accuracy

Out[32]: 0.55

In [33]: testing_accuracy = rf_classifier.score(X_test, y_test).round(2)
         testing_accuracy

Out[33]: 0.38

In [34]: classifers_train_test.loc[4] = ["Random Forest", training_accuracy, testing_accuracy]
```

4.3 Accuracy of models

```
In [35]: classifers_train_test
```

Out[35]:

	Classifier	Train_AUC	Test_AUC
0	Naive Bayes	0.58	0.52
1	Logistic Regression	0.97	0.98
2	SVM	0.25	0.19
3	KNN	0.89	0.85
4	Random Forest	0.55	0.38

Logistic Regression gives the best accuracy and performance among all five models.

Support Vector Machine Classifier gives worst accuracy and performance.

Module 5 – Clustering for Diverse portfolio analysis

Clustering is a method of unsupervised learning and is a common technique for statistical data analysis used in many fields.

Clustering is a Machine Learning technique that involves the grouping of data points. Given a set of data points, we can use a clustering algorithm to classify each data point into a specific group. In theory, data points that are in the same group should have similar properties and/or features, while data points in different groups should have highly dissimilar properties and/or features.

In financial Markets, Cluster analysis is a technique used to group sets of objects that share similar characteristics. It is common in statistics, but investors will use the approach to build a diversified portfolio. Stocks that exhibit high correlations in returns fall into one basket, those slightly less correlated in another, and so on, until each stock is placed into a category.

5.1 Create a table/data frame with the closing prices of 30 different stocks, with 10 from each of the caps.

In [1]:
```
import pandas as pd
close_price_30 = pd.DataFrame([])
```

In [2]:
```
import os
# Iterate over all 30 stocks
for i in os.listdir("./30_Stock_files/"):
    current_df = pd.read_csv("./30_Stock_files/"+str(i))
    # Take rows only with Series 'EQ'
    current_df = current_df[current_df.Series == 'EQ'].reset_index(drop=True)
    # Take Closing Price column from each stock
    close_price_30[current_df.Symbol[0]] = current_df["Close Price"]

print(f"Shape of close_price_30 : {close_price_30.shape[0]} Rows & {close_price_30.shape[1]} Columns")
close_price_30.head()
```

```
Shape of close_price_30 : 494 Rows & 30 Columns
```

Out[2]:

	APOLLOTYRE	ASHOKA	ASIANPAINT	BAJAJELEC	BAJFINANCE	CENTURYPLY	DHFL	HDFCBANK	HEROMOTOCO	HINDUNILVR	...	RAYMOND	RBLBANK	RCOM
0	231.90	216.05	1148.05	341.15	1332.95	266.65	431.40	1553.40	3515.45	983.25	...	772.80	563.90	32.00
1	234.40	214.90	1142.85	347.00	1347.75	266.10	424.45	1559.65	3619.40	1000.40	...	785.00	562.60	32.10
2	237.35	217.00	1154.95	349.85	1324.80	264.85	429.00	1557.15	3645.95	1009.40	...	783.65	564.25	31.95
3	232.65	209.65	1151.35	334.10	1314.55	260.35	417.95	1557.10	3592.05	990.25	...	746.95	552.15	31.05
4	234.65	206.25	1123.15	336.20	1289.15	254.05	404.20	1561.25	3571.65	1008.00	...	723.10	555.45	30.55

5 rows × 30 columns

5.2 Calculate average annual percentage return and volatility of all 30 stocks over a theoretical one year period.

6.2.1 Average Annual Percentage Return

In [5]:
```
annual_perc_return = daily_perc_return.mean()
annual_perc_return.head()
```

Out[5]:
```
APOLLOTYRE   -0.152627
ASHOKA       -0.291083
ASIANPAINT    0.055553
BAJAJELEC    -0.037507
BAJFINANCE    0.206037
dtype: float64
```

6.2.2 Annual Percentage Volatility

In [6]:
```
annual_volatility = daily_perc_return.std()
annual_volatility.head()
```

Out[6]:
```
APOLLOTYRE    1.946514
ASHOKA        3.242312
ASIANPAINT    1.463245
BAJAJELEC     2.367102
BAJFINANCE    2.206760
dtype: float64
```

5.3 Cluster the 30 stocks according to their mean annual Volatilities and Returns using K-means clustering. Identify the optimum number of clusters using the Elbow curve method.

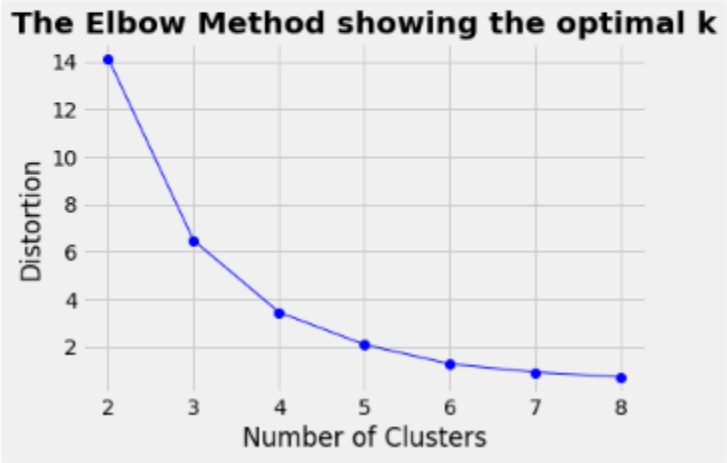
In [7]:
```
# Dataframe with column annual returns and annual volatilities
clustering_data = pd.DataFrame({'Returns':annual_perc_return, 'Volatilities':annual_volatility})
```

6.3.1 Elbow Curve Method Procedure

In [8]:
```
from sklearn.cluster import KMeans

# define how many cluster we want to test
cluster_num = 8

# define the dataframe that will contains all our relevant information for each cluster size.
diff_cluster_result = pd.DataFrame(index=pd.np.arange(2,cluster_num+1),
                                   columns=['loss', 'score','center_returns', 'center_volatility', 'cluster_label', 'model'])

for k in range(2, cluster_num+1):
    # create an instance of the model, and fit the training data to it.
    kmeans = KMeans(n_clusters=k, random_state=0).fit(clustering_data)

    # Inertia/Loss/Distortion: Sum of distances of samples to their closest cluster center.
    diff_cluster_result['loss'][k] = kmeans.inertia_
    diff_cluster_result['center_returns'][k] = kmeans.cluster_centers_[:,0]
    diff_cluster_result['center_volatility'][k] = kmeans.cluster_centers_[:,1]
    diff_cluster_result['cluster_label'][k] = kmeans.labels_.astype(float)
    diff_cluster_result['score'][k] = kmeans.score
    diff_cluster_result['model'][k] = kmeans
```

5.4 Plotting of Elbow Curve Graph and performing cluster visualization.

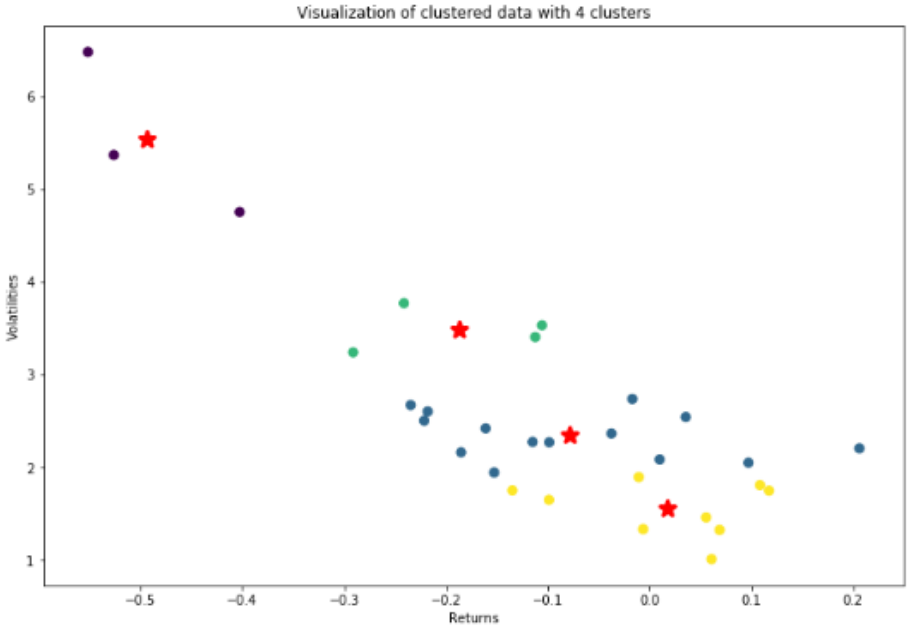
In [9]:
```
import matplotlib.pyplot as plt
%matplotlib inline
with plt.style.context('fivethirtyeight'):
    plt.plot(diff_cluster_result.index.values, diff_cluster_result.loss, 'bo-', linewidth=1)
    plt.xlabel("Number of Clusters")
    plt.ylabel("Distortion")
    plt.title('The Elbow Method showing the optimal k', fontweight='bold')
    plt.show()
```



Optimum Number of clusters = 4

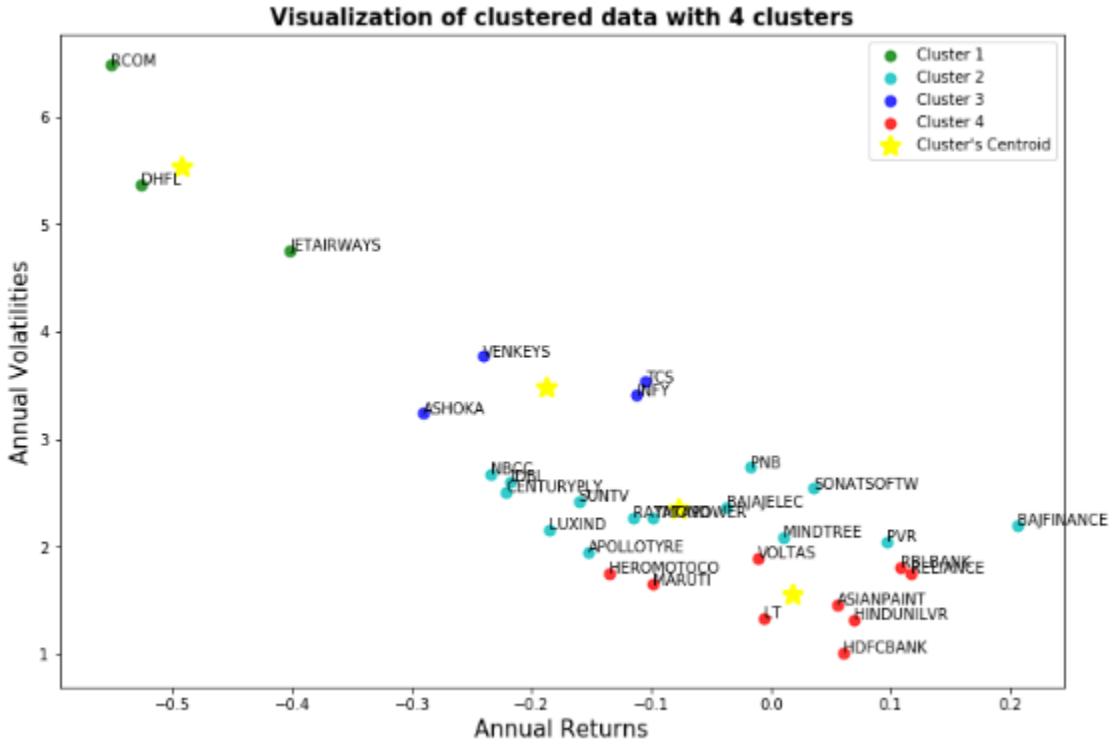
In [11]:
```
k_optimized = 4
kmeans_optimized = diff_cluster_result.loc[k_optimized]
```

In [12]:
```
clustering_data.plot(kind='scatter', x='Returns', y='Volatilities',
                     c=kmeans_optimized.cluster_label, s=50,
                     cmap='viridis', colorbar=False, figsize=(12,8),
                     title='Visualization of clustered data with {} clusters'.format(k_optimized))
plt.scatter(kmeans_optimized.center_returns, kmeans_optimized.center_volatility, marker='*', c='r', s=150, linewidths=3)
plt.show()
```



5.5 Plot Clustering of Stocks

In [14]:
```
plt.figure(figsize=(12,8))

# Plot all clusters
for i in range(len(all_clusters)):
    plt.scatter(all_clusters[i].Returns, all_clusters[i].Volatilities, s=50, label=f"Cluster {i+1}", c=list('gcbr')[i], alpha=0.8)

# Annotate stock name to each data point
for i in range(len(clustering_data)):
    plt.annotate(clustering_data.index[i], (clustering_data.Returns[i], clustering_data.Volatilities[i]))

# Plot Centriod of each cluster
plt.scatter(kmeans_optimized.center_returns, kmeans_optimized.center_volatility, marker='*', c='yellow', s=150, linewidth=3, label="Cluster

plt.title("Visualization of clustered data with 4 clusters", fontweight='bold', fontsize=15)
plt.xlabel("Annual Returns", fontsize=15)
plt.ylabel("Annual Volatilities", fontsize=15)
plt.legend()
plt.show()
```



6.4.3 Separate Data frame of Each Cluster

In [15]:
```
cluster1
```

Out[15]:

	Returns	Volatilities	label
DHFL	-0.526120	5.368415	1
JETAIRWAYS	-0.402479	4.754339	1
RCOM	-0.551513	6.478758	1

In [16]:
```
cluster2
```

Out[16]:

	Returns	Volatilities	label
APOLLOTYRE	-0.152627	1.946514	2
BAJAJELEC	-0.037507	2.367102	2
BAJFINANCE	0.206037	2.206760	2
CENTURYPLY	-0.221293	2.505610	2
IDBI	-0.217885	2.604046	2
LUXIND	-0.185099	2.164111	2
MINDTREE	0.009951	2.086993	2
NBCC	-0.234637	2.673751	2
PNB	-0.016995	2.739882	2
PVR	0.097040	2.052217	2
RAYMOND	-0.114845	2.276097	2
SONATSOFTW	0.035575	2.544301	2
SUNTV	-0.160940	2.422537	2
TATAPOWER	-0.098589	2.272462	2

In [17]:
```
cluster3
```

Out[17]:

	Returns	Volatilities	label
ASHOKA	-0.291083	3.242312	3
INFY	-0.112286	3.407303	3
TCS	-0.105631	3.533329	3
VENKEYS	-0.241175	3.771359	3

In [18]:
```
cluster4
```

Out[18]:

	Returns	Volatilities	label
ASIANPAINT	0.055553	1.463245	4
HDFCBANK	0.060881	1.013472	4
HEROMOTOCO	-0.134744	1.754052	4
HINDUNILVR	0.068739	1.327639	4
LT	-0.006306	1.336219	4
MARUTI	-0.098799	1.651774	4
RBLBANK	0.108374	1.810614	4
RELIANCE	0.117284	1.751214	4
VOLTAS	-0.010604	1.897297	4

Chapter 6: Learnings and Takeaways from the Study

Predicting the stock market was a time-consuming and laborious procedure a few years or even a decade ago. However, with the application of machine learning for stock market forecasts, the procedure has become much simpler. Machine learning not only saves time and resources but also outperforms people in terms of performance.

The Stock market jargon seems to be more familiar than before and the model that we trained provided with increasing accuracy with every single improvisation made, which made the whole model training work fulfilling.

I came across multiple stock market terms like open, high, low, close, market capitalization and much more which helped me to understand the market in a much better way. This project upheld the importance of cohesion in a team as well, supporting each other. We grew together developing the concerned project. I learnt to work efficiently and consider a multidimensional point of view about the overall work.

All together it has been a satisfying work experience with a lot of new lessons.

REFERENCES AND ANNEXURES

[1] Azimifar, M., Araabi, B. N., & Moradi, H. (2020). *Forecasting stock market trends using support vector regression and perceptually important points. 2020 10th International Conference on Computer and Knowledge Engineering (ICCKE).* doi:10.1109/iccke50421.2020.9303667.

[2] Bhuriya, D., Kaushal, G., Sharma, A., & Singh, U. (2017). *Stock market predication using a linear regression. 2017 International Conference of Electronics, Communication and Aerospace Technology (ICECA).* doi:10.1109/iceca.2017.8212716.

[3] Fonseca, A. R., Leles, M. C. R., Moreira, M. G., Vale-Cardoso, A. S., Pereira, M. V. L., Sbruzzi, E. F., & Nascimento, C. L. (2021). *Testing the Application of Support Vector Machine (SVM) to Technical Trading Rules. 2021 IEEE International Systems Conference (SysCon).* doi:10.1109/syscon48628.2021.9447068.

[4] Huang, X. (202*0). Research on Strategy of HS300 Index Based on Random Forest. 2020 International Conference on Computing and Data Science (CDS). https://doi.org/10.1109/cds49703.2020.00038.*

[5] Kumari, S., Patil, N., Nankar, P., & Kulkarni, M. (2020). *CUDA parallel computing framework for stock market prediction using K-means clustering. 2020 International Conference on Smart Electronics and Communication (ICOSEC).* doi:10.1109/icosec49089.2020.9215377.

[6] Malawana, M. V. D. H. ., & Rathnayaka, R. M. K. T. (2020). *The Public Sentiment analysis within Big data Distributed system for Stock market prediction– A case study on Colombo Stock Exchange. 2020 5th International Conference on Information Technology Research (ICITR).* doi:10.1109/icitr51448.2020.9310871.

[7] Mikael Sinaga, F., Sirait, P., & Halim, A. (2020). *Optimization of SV-kNNC using Silhouette Coefficient and LMKNN for Stock Price Prediction. 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI).* doi:10.1109/isriti51436.2020.9315516.

[8] Panwar, B., Dhuriya, G., Johri, P., Singh Yadav, S., & Gaur, N. (2021). *Stock Market Prediction Using Linear Regression and SVM. 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE).* doi:10.1109/icacite51222.2021.9404733.

[9] Ravikumar, S., & Saraf, P. (2020). *Prediction of Stock Prices using Machine Learning (Regression, Classification) Algorithms. 2020 International Conference for Emerging Technology (INCET).* doi:10.1109/incet49848.2020.9154061.

[10] Su, C.-Y., Lei, Y., & Wang, M.-X. (2021). *Research and Comparison of Random Forests and Neural Networks in Shanghai and Shenzhen Financial 20 Index Prediction. 2021 World Conference on Computing and Communication Technologies (WCCCT).* doi:10.1109/wccct52091.2021.00023.

[11] Pingwen Xue, Yuan Lei, Yanxing Li. (2020). *Research and prediction of Shanghai-Shenzhen 20 Index Based on the Support Vector Machine Model and Gradient Boosting Regression Tree. 2020 International Conference on Intelligent Computing, Automation and Systems (ICICAS).* doi: 10.1109/icicas51530.2020.00019.