

Stock Market Trend Prediction Using K-Nearest Neighbor (KNN) Algorithm

ABSTRACT

This paper examines a hybrid model which combines a K-Nearest Neighbors (KNN) approach with a probabilistic method for the prediction of stock price trends. One of the main problems of KNN classification is the assumptions implied by distance functions. The assumptions focus on the nearest neighbors which are at the centroid of data points for test instances. This approach excludes the non-centric data points which can be statistically significant in the problem of predicting the stock price trends. For this it is necessary to construct an enhanced model that integrates KNN with a probabilistic method which utilizes both centric and non-centric data points in the computations of probabilities for the target instances. The embedded probabilistic method is derived from Bayes' theorem. The prediction outcome is based on a joint probability where the likelihood of the event of the nearest neighbors and the event of prior probability occurring together and at the same point in time where they are calculated. The proposed hybrid KNN Probabilistic model was compared with the standard classifiers that include KNN, Naive Bayes, One Rule (OneR) and Zero Rule (ZeroR). The test results showed that the proposed model outperformed the standard classifiers which were used for the comparisons.

Keywords: Stock Price Prediction, K-Nearest Neighbors, Bayes' Theorem, Naive Bayes, Probabilistic Method

1. INTRODUCTION

Analyzing financial data in securities has been an important and challenging issue in the investment community. Stock price efficiency for public listed firms is difficult to achieve due to the opposing effects of information competition among major investors and the adverse selection costs imposed by their information advantage.

There are two main schools of thought in analyzing the financial markets. The first approach is known as fundamental analysis. The methodology used in fundamental analysis evaluates a stock by measuring its intrinsic value through qualitative and quantitative

analysis. This approach examines a company's financial reports, management, industry, micro and macro-economic factors.

The second approach is known as technical analysis. The methodology used in technical analysis for forecasting the direction of prices is through the study of historical market data. Technical analysis uses a variety of charts to anticipate what are likely to happen. The stock charts include candlestick charts, line charts, bar charts, point and figure charts, OHLC (open-high-low-close) charts and mountain charts. The charts are viewable in different time frames with price and volume. There are many types of indicators used in the charts, including resistance, support, breakout, trending and momentum.

Several alternatives to approach this type of problem have been proposed, which range from traditional statistical modelling to methods based on computational intelligence and machine learning. Vanstone and Tan surveyed the works in the domain of applying soft computing to financial trading and investment. They categorized the papers reviewed in the following areas: time series, optimization, hybrid methods, pattern recognition and classification. Within the context of financial trading discipline, the survey showed that most of the research was being conducted in the field of technical analysis. An integrated fundamental and technical analysis model was examined to evaluate the stock price trends by focusing on macro-economic analysis. It also analyzed the company behaviour and the associated industry in relation to the economy which in turn provide more information for investors in their investment decisions.

A nearest neighbor search (NNS) method produced an intended result by the use of KNN technique with technical analysis. This model applied technical analysis on stock market data which include historical price and trading volume. It applied technical indicators made up of stop loss, stop gain and RSI filters. The KNN algorithm part applied the distance function on the collected data. This model was compared with the buy-and-hold strategy by using the fundamental analysis approach.

Fast Library for Approximate Nearest Neighbors (FLANN) is used to perform the searches for choosing the best algorithm found to work best among a collection of algorithms in its library. Majhi et al. examined the FLANN model to predict the S&P 500 indices, and the FLANN model was established by performing fast approximate nearest neighbor searches in high dimensional spaces.

Artificial neural networks (ANN) exhibit high generalization power as compared to conventional statistical tools. ANN is able to infer from historical data to identify the characteristics of performing stocks. The information is reflected in technical and financial variables. As a result, ANN is used as a statistical tool to explore the intricate relationships between the related financial and technical variables and the performance of stocks.

Neural network modelling can decode nonlinear regularities in asset price movements. Statistical inference and modifications to standard learning techniques prove useful in dealing with the salient features of economic data.

Some research has been carried out through the use of both qualitative and quantitative analysis. Shynkevich et al. studied how the performance of a financial forecasting model was improved by the use of a concurrent, and appropriately weighted news articles, having different degrees of relevance to the target stock. The financial model supported the decision-making process of investors and traders. Textual pre-processing techniques were utilized for the predictive system. A multiple kernel learning technique was applied to predict the stock price movements. The technique integrated information extracted from multiple news categories while separate kernels were used to analyze each category. The news articles were partitioned according to some factors from the industries and their relevance to the target stock. The experiments were performed on stocks from the health care sector. The results showed that the financial forecasting model had achieved better performance when data sources contain increased categories of the number of relevant news. An enhanced model for this study incorporated additional data source using historical prices and made predictions based on both textual and time series data. Additional kernels can be employed for different data sources. The use of new categorical features was to improve the forecasting performance.

Linear regression is commonly used in financial analysis and forecasting. Many regression classifiers had demonstrated their usefulness to analyze quantitative data to make forecast by estimating the model parameters.

A regression driven fuzzy transform (RDFT) distributes a smoothing approximation of time series with a smaller delay as compared with moving average. This feature is important for forecasting tool where time plays a key role.

In high dimensional data, not all features are relevant and have an influence on the outputs. An Enhanced Feature Representation Based on Linear Regression Model for Stock Market Prediction was evaluated to investigate the statistical metrics used in feature selection that extracts the most relevant features to reduce the high dimensionality of the data. The statistical metrics include Information Gain, Term Frequency-Invert Document Frequency and the Document Frequency. The study illustrated that the identification of the relevant feature representations produced better result in the prediction output.

Volatility indicates the risk of a security. The Generalized Autoregressive Conditional Heteroscedasticity (GARCH) process is an approach used to estimate volatility in financial markets. The Seemingly Unrelated Regressions (SUR) is a generalization of a linear regression model that comprises several indicator relationships that are linked by the fact that their volatilities are correlated. A GARCH-SUR model was evaluated and demonstrated that the existence of a significant relationship between the volatility of macroeconomic variables and the stock market volatility in the financial markets.

As a company raises investment capital by offering its security to the public for the first time in an Initial Public Offering (IPO), there is a lot of volatilities in the absence of IPO lock-up period which is a cooling-off period that allows for the newly issued securities to stabilize without additional selling pressures from insiders. A study was conducted to investigate the lock-up provisions of IPOs and their effect on price changes around the lock-up expiry periods by analyzing the Efficient Market Hypothesis (EMH) in relation to the lockup provision by using the standard Event Study Methodology and the Comparison Period Returns Approach (CPRA). The results showed that regardless of whether the market risks were incorporated in the analysis, the financial markets remained in semi-strong form around the lock-up expiry date. The results of the CPRA and the event study methodology also showed a positive abnormal trading volume at lock-up expiry date for most of the sectors in the IPO market.

Maetal. proposed a hybrid financial time series model by combining Support Vector Regression (SVR), Trend model and Maximum Entropy (ME) based on ANN for forecasting trends in fund index. The study showed that the hybrid model extracted the financial features characteristics to formulate an improved predictive model.

A hybrid intelligent data mining methodology based on Genetic Algorithm - Support Vector Machine Model was reviewed to explore stock market tendency. This approach makes

use of the genetic algorithm for variable selection in order to improve the speed of support vector machine by reducing the model complexity, and then the historical data is used to identify stock market trends. Hybrid techniques can be used to improve the existing forecasting models due to the limitation of ANN like black box technique. A combination of methods such as fuzzy rule-based system, fuzzy neural network and Kalman filter with hybrid neuro-fuzzy architecture have been developed to predict financial time series data.

This research studies a hybrid approach through the use of KNN algorithm and a probabilistic method for predicting the stock price trends.

1.1 Comparison of various learning classifiers

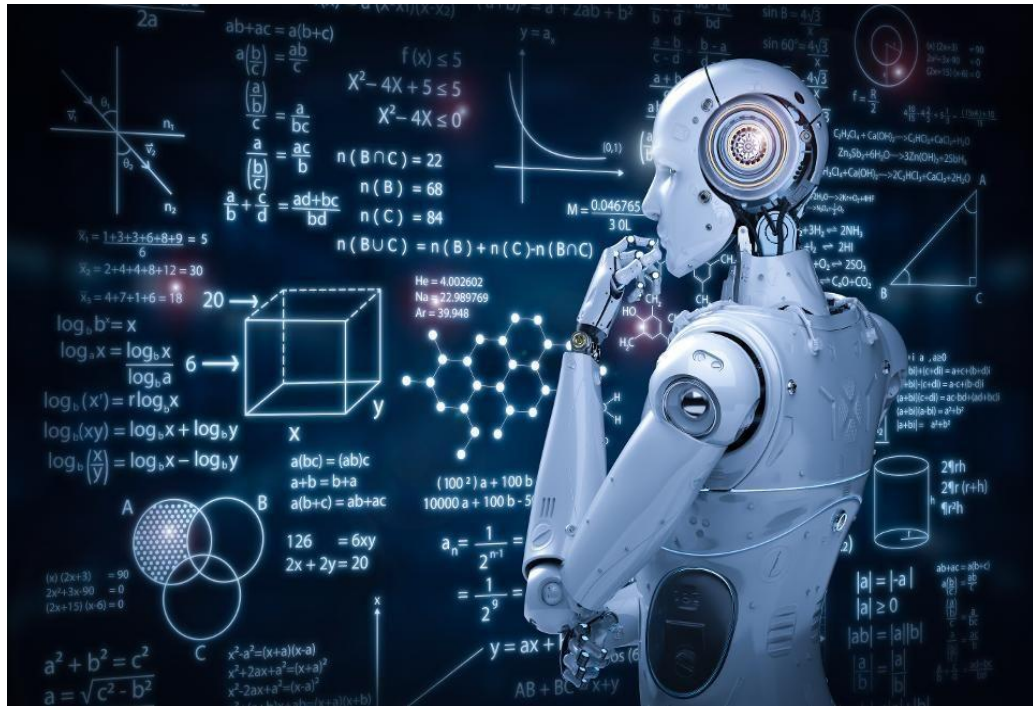
When developing a classifier using various functions from different classifiers, it is important to compare the performances of the classifiers. Simulation results can provide us with direct comparison results for the classifiers with a statistical analysis of the objective functions. The hybrid KNN-Probabilistic model was compared with the supervised learning and classification algorithms, including ‘KNN’, ‘Naïve Bayes’, ‘OneR’ and ‘ZeroR’.

Machine Learning (ML):

Introduction

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people. Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve.

Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs.



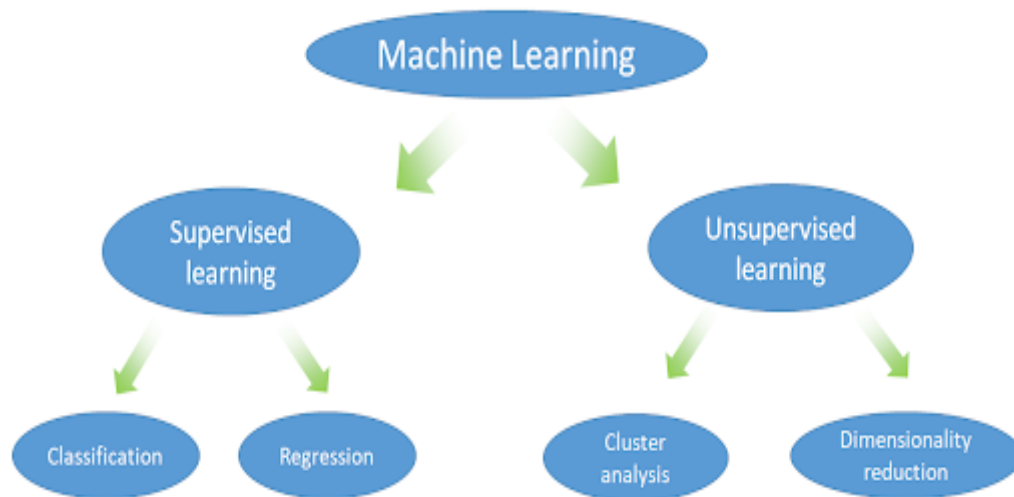
Any technology user today has benefitted from machine learning. Facial recognition technology allows social media platforms to help users tag and share photos of friends. Optical character recognition (OCR) technology converts images of text into movable type.

Machine learning is a continuously developing field. Because of this, there are some considerations to keep in mind as you work with machine learning methodologies, or analyze the impact of machine learning processes. We'll look into the common machine learning methods of supervised and unsupervised learning, and common algorithmic approaches in machine learning, including the k-nearest neighbor algorithm, decision tree learning, and deep learning.

Machine Learning Methods:

Two of the most widely adopted machine learning methods are

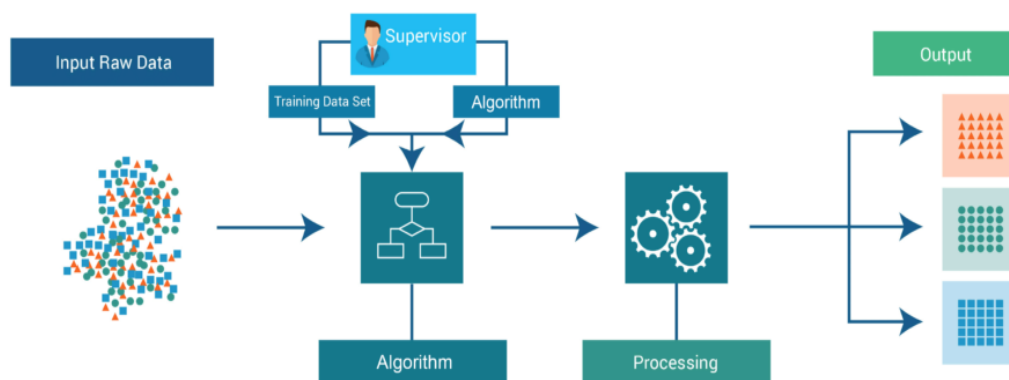
1. Supervised learning
2. Unsupervised learning



1. Supervised learning:

In supervised learning, the computer is provided with example inputs that are labeled with their desired outputs. The purpose of this method is for the algorithm to be able to “learn” by comparing its actual output with the “taught” outputs to find errors, and modify the model accordingly. Supervised learning therefore uses patterns to predict label values on additional unlabeled data.

Supervised Learning

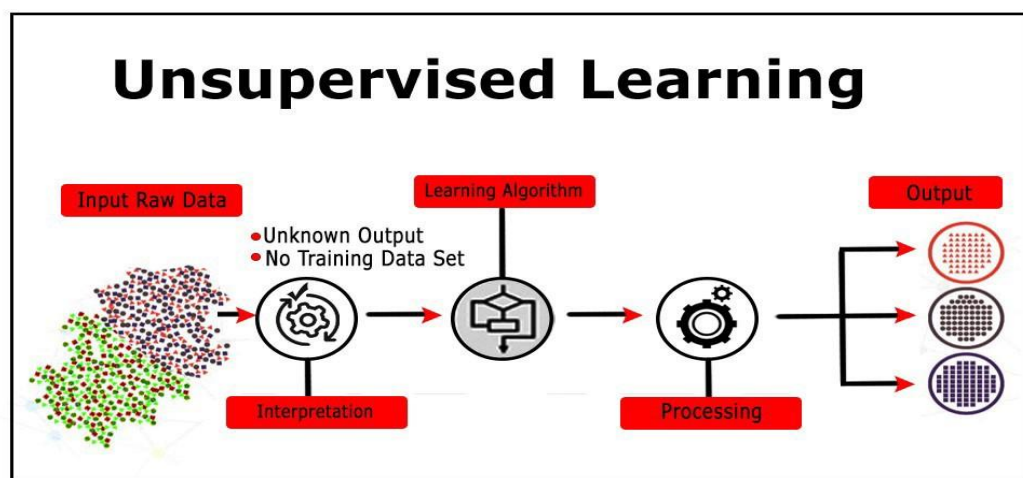


A common use case of supervised learning is to use historical data to predict statistically likely future events. It may use historical stock market information to anticipate upcoming fluctuations, or be employed to filter out spam emails. In supervised learning, tagged photos of dogs can be used as input data to classify untagged photos of dogs.

2. Unsupervised Learning:

In unsupervised learning, data is unlabeled, so the learning algorithm is left to find commonalities among its input data. As unlabeled data are more abundant than labeled data, machine learning methods that facilitate unsupervised learning are particularly valuable.

The goal of unsupervised learning may be as straightforward as discovering hidden patterns within a dataset, but it may also have a goal of feature learning, which allows the computational machine to automatically discover the representations that are needed to classify raw data. Unsupervised learning is commonly used for transactional data.

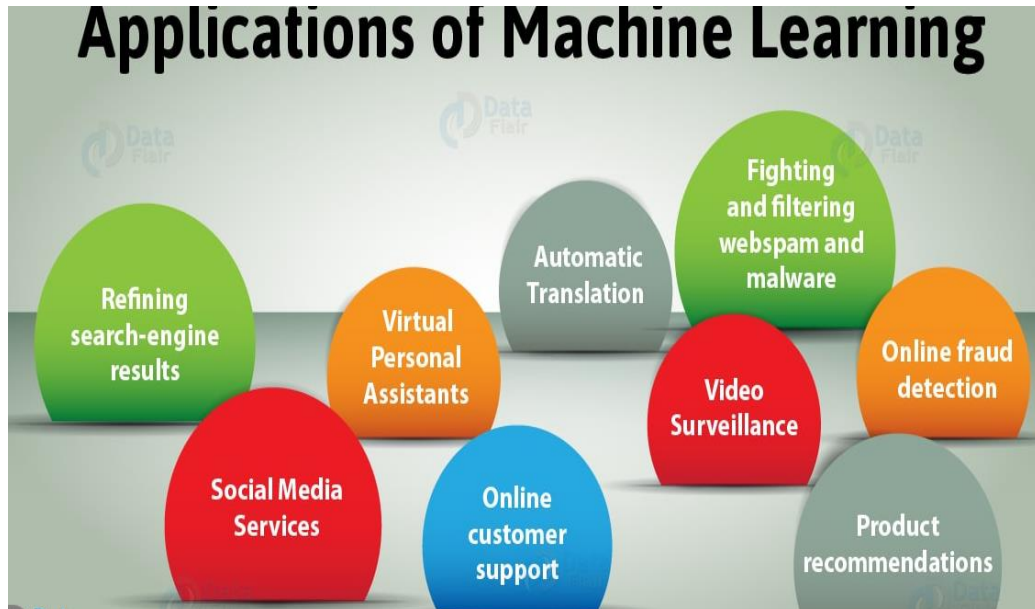


Without being told a “correct” answer, unsupervised learning methods can look at complex data that is more expansive and seemingly unrelated in order to organize it in potentially meaningful ways. Unsupervised learning is often used for anomaly detection including for fraudulent credit card purchases, and recommender systems that recommend what products to buy next.

Application of Machine Learning:

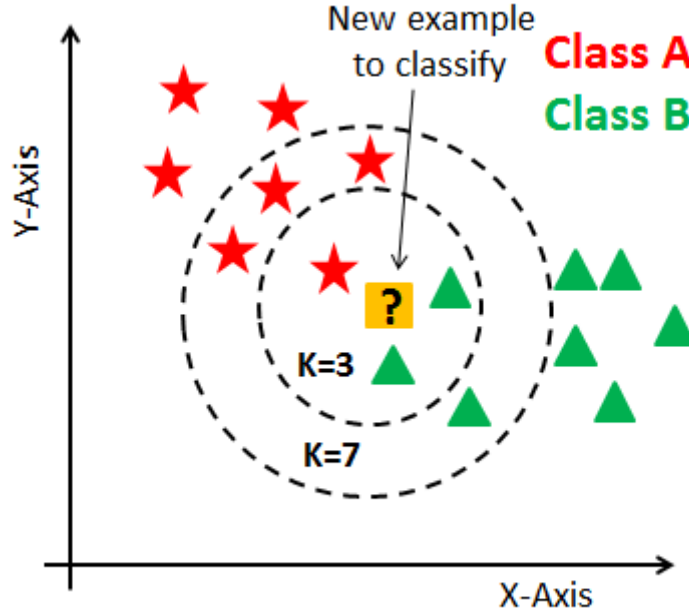
- Face Recognition
- Social Media Services
- Virtual Personal Assistants
- Online Fraud Detection
- Product Recommendations
- Autonomous

- Self-Driving Cars
- Stock Market Trading



1.1.1 K-Nearest Neighbors Classifier

The KNN algorithm is used to measure the distance between the given test instance and all the instances in the data set, this is done by choosing the 'k' closest instances and then predict the class value based on these nearest neighbors. The 'k' is assigned as number of neighbors voting on the test instance. As such KNN is often referred to as case based learning or an instance-based learning where each training instance is a case from the problem domain. KNN is also referred to as a lazy learning algorithm due to the fact that there is no learning of the model required and all of the computation works happen at the time a prediction is requested. KNN is a non-parametric machine learning algorithm as it makes no assumptions about the functional form of the problem being solved. Each prediction is made for a new instance (x) by searching through the entire training set for the 'k' most nearest instances and applying majority voting rule to determine the prediction outcome.



A variety of distance functions are available in KNN which include Euclidean, Manhattan, Minkowski and Hamming. The Euclidean distance function is probably the most commonly used in any distance-based algorithm. It is defined as

$$d(x,y) = \sqrt{\sum_{i=1}^k (x_i - y_i)^2}$$

- (1) Where, x and y are two data vectors and k is the number of attributes. The Manhattan distance function is defined as

$$d(x,y) = (\sum_{i=1}^k |x_i - y_i|)$$

- (2) The Minkowski distance function is defined as

$$d(x,y) = (\sum_{i=1}^k (|x_i - y_i|^p))^{1/p}$$

- (3) The distance functions for Euclidean, Manhattan and Minkowski are used for numerical attributes. The Hamming distance function is defined as

$$d(x,y) = \sum_{i=1}^k |x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

- (4) Hamming distance is usually used for categorical attributes. The Hamming distance between two data vectors is the number of attributes in which they differ.

Advantages of KNN Algorithm:

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

Disadvantages of KNN Algorithm:

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

1.1.2 Bayes' Theorem

The Bayes' theorem plays an important role in probabilistic learning and classification. The Bayesian classification represents a supervised learning method as well as a statistical method for classification. It has learning and classification methods based on probability theory.

The Bayesian classification is named after Thomas Bayes, who proposed the Bayes' theorem. Bayes' theorem is often called Bayes' rule. The Bayes' rule uses prior probability of each category given no information about an item. Bayesian classification provides probabilistic methods where prior knowledge and observed data can be combined. It has a useful perspective for evaluating a variety of learning algorithms.

The Bayesian classification calculates explicit probabilities for hypothesis and it is also robust to noise in input data. Given a hypothesis h and data D which bears on the hypothesis, Bayes' theorem is stated as

$$P(h/D) = \frac{P(h/D)P(h)}{P(D)}$$

Where:

$P(D)$: independent probability of D

$P(h)$: independent probability of h: prior probability

$P(h|D)$: conditional probability of h given D: posterior probability

$P(D|h)$: conditional probability of D given h: likelihood

a. Naïve Bayes Classifier

The Naïve Bayes classifier is a classification method based on Bayes' theorem with independence assumptions among predictors. A Naive Bayes classifier assumes that the presence of a particular attribute in a class is unrelated to the presence of any other attribute. A Naive Bayes model is easy to build, with no complicated iterative parameter estimation that makes it particularly useful for very large data sets. In spite of its simplicity, the Naïve Bayes classifier often does surprisingly well and is widely used due to the fact that it often outperforms other more sophisticated classification techniques. The Bayes' rule from (5) can also be expressed as

$$P(h/D) = P(d_1|h) \times P(d_2|h) \times \dots \times P(d_n|h) \times P(h)$$

Where:

h_1, h_2, \dots, h_n be a set of mutually exclusive events that together form the sample space S.

D be any event from the same sample space, such that $P(D) > 0$.

Advantages of Naïve Bayes Classifier:

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for **text classification problems**.

Disadvantages of Naïve Bayes Classifier:

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

b. Bayesian Network Classifier

A Bayesian network is a type of graph which is used to model events for probabilistic relationships among a set of random variables. This can then be used for inference. The graph is called a directed acyclic graph (DAG). DAG contains nodes and edges. A node is also called a vertex. The nodes of the graph represent random variables. Edges represent the connections between the nodes. If two nodes are connected by an edge, it has an associated probability that it will transmit from one node to the other. The graph is directed and does not contain any cycles. A directed graph has ordered pairs of vertices and it consists of a set of vertices and a set of arcs. In a DAG diagram, a vertex is represented by a circle with a label, and an edge is represented by an arrow or line extending from one vertex to another. Bayesian Network provides a representation of the joint probability distribution over the variables. A problem domain is modeled by a list of variables X_1, \dots, X_n . Knowledge about the problem domain is represented by a joint probability $P(X_1, \dots, X_n)$.

A joint probability refers to the probability of more than one variable occurring together, such as the probability of event A and event B. Notation for joint probability of two events takes the form $P(A \cap B)$ which denotes the probability of the intersection of A and B. Joint probability for dependent events of event A and event B can be expressed as

$$P(A \cap B) = P(A) \times P(B | A)$$

Where:

“ \cap ” denotes the point where A and B intersect.

$P(A)$ is the probability of event A occurs

$P(B | A)$ is the conditional probability of B given A

Joint probability for independent events of event A and event B can be expressed as

$$P(A \cap B) = P(A) \times P(B)$$

In a joint distribution for independent variables, two discrete random variables X and Y are independent if the joint probability mass function conforms to

$$P(X = x \text{ and } Y = y) = P(X = x) \times P(Y = y) \text{ for all } x \text{ and } y.$$

1.1.3 Decision Tree Learning

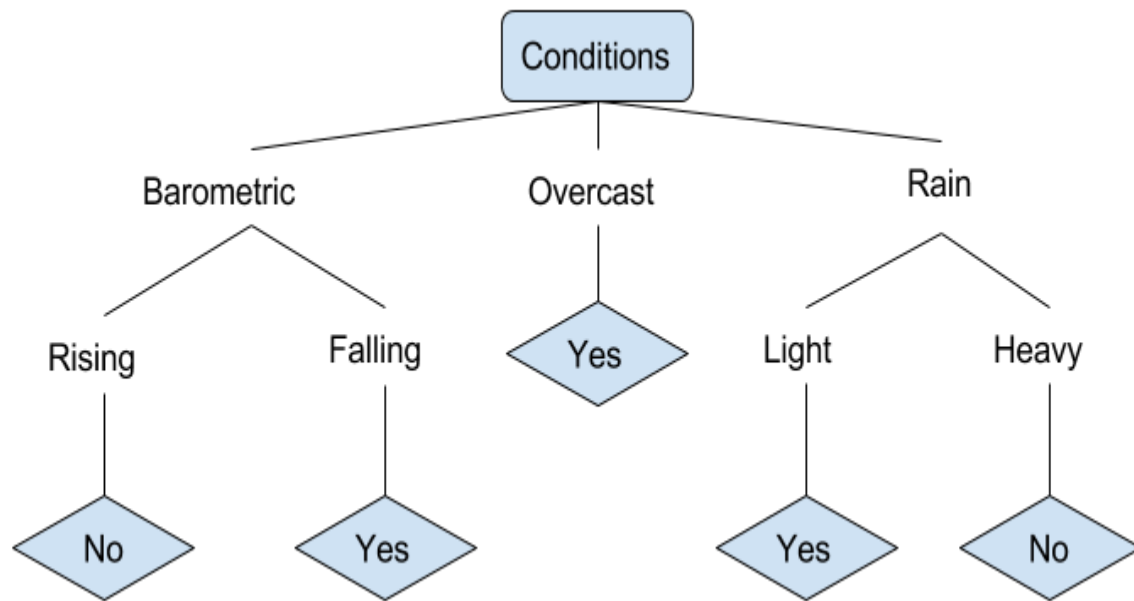
For general use, decision trees are employed to visually represent decisions and show or inform decision making. When working with machine learning and data mining, decision trees are used as a predictive model. These models map observations about data to conclusions about the data's target value.

The goal of decision tree learning is to create a model that will predict the value of a target based on input variables.

In the predictive model, the data's attributes that are determined through observation are represented by the branches, while the conclusions about the data's target value are represented in the leaves.

When "learning" a tree, the source data is divided into subsets based on an attribute value test, which is repeated on each of the derived subsets recursively. Once the subset at a node has the equivalent value as its target value has, the recursion process will be complete.

Let's look at an example of various conditions that can determine whether or not someone should go fishing. This includes weather conditions as well as barometric pressure conditions.



In the simplified decision tree above, an example is classified by sorting it through the tree to the appropriate leaf node. This then returns the classification associated with the particular leaf, which in this case is either a `Yes` or a `No`. The tree classifies a day's conditions based on whether or not it is suitable for going fishing.

1.2 Related Works

A study on a KNN approach that made use of economic indicators and classification techniques to predict the stock price trends yielded considerable precision. Four indicators were identified and calculated based on the technical indicators and their formulas. The values of the indicators were normalized in the range between -1 and +1. Accuracy and F-measure were calculated to evaluate the performance of the model. Calculation of these evaluation measures required estimating Recall and Precision which were assessed from True Negative (TN), False Negative (FN), True Positive (TP) and False Positive (FP). The performance of the KNN model was improved by using the optimal value for the k parameter. The evaluation of the KNN model and its performance is illustrated in Table 1.

Table 1: KNN Model Performance.

Measurement	K Parameter		
	25	45	70
Accuracy	0.8138	0.8059	0.8132
F-measure	0.8202	0.8135	0.8190

Another study examined a model named Integrated Multiple Linear Regression-One Rule Classification Model (Regression-OneR) that predicts the stock class outputs in classification form based on the initial predicted outputs in regression form. The regression classifiers were used to predict the outputs in continuous values as opposed to the classification classifiers which were used to predict the outputs in categorical values. As illustrated in Table 2, the result showed that the hybrid regression-classification approach produced better accuracy rate and lower Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) as compared with the model which incorporated only a standard classification algorithm.

Table 2: Prediction Results of Classifiers.

Classifier	Accuracy	MAE	RMSE
Regression-OneR	85.0746	0.1493	0.3863
OneR	71.6418	0.2836	0.5325
ZeroR	64.1791	0.4619	0.4805
J48 Decision Tree	82.0896	0.2121	0.3796
REP Tree	76.1194	0.2764	0.4207

2. MATERIALS AND METHODS

Data were collected from Bursa Malaysia which is the main stock exchange in Malaysia. The data sources are corporate annual reports which include income statements, cash flow and balance sheet. The features in the data set were formulated based on fundamental analysis. The features were financial ratios which are indicators of the companies' financial health. Table 3 and Table 4 illustrate the structure of data set 1 and data set 2. Data set 1 contains original data from heterogeneous sources with different data types used for currency values and financial ratios. All of the variables in data set 1 have numerical data type. The data in data set 1 was transformed into data set 2 with variables in categorical data type. The transformed data set provides a way to measure rankings of stock price performance in a standardized data format. The transformation process is shown in Figure 1.

Figure 1, illustrates that initially the numerical data from corporate reports were used as inputs. At this stage the data are raw facts. The raw data are then used for calculations of the financial ratios. The values in financial ratios are fractional type. At this stage the data are semantic but diverse. After that, the fractional data are then converted into standardized percentage values based on performance interpretations of the features. At this stage the data are standardized. A ranking table is set up to group performance categories based on ranges of percentage values. The table is then used in data mapping to categorize the data from the percentage data format into the categorical data format. At this stage the data are interpretable.

2.1 Data Sets

The naming conventions for the features in the data sets are closely resembled to the financial ratio terms which include debt/equity, asset turnover, cash flow and return on equity. The original source of data in data set 1 contains numeric values which were then converted into categorical values in data set 2.

Debt_Equity, Asset_Turnover, Cash_Flow, Return_on_Equity are the independent variables. The values of the independent variables are used to predict the value of the class variable named Price_Trend. The variable named Price in data set 1 contains numerical values which do not provide semantic interpretations of the stocks since various stocks have various prices. Whereas the variable named Price_Trend in data set 2 contains standardized categorical values which indicate positive or negative price trend.

Table 3: Data set 1.

Feature	Data Type
Debt_Equity	Numeric
Asset_Turnover	Numeric
Cash_Flow	Numeric
Return_On_Equity	Numeric
Price	Numeric

Table 4: Data set 2.

Feature	Data Type
Debt_Equity	Categorical
Asset_Turnover	Categorical
Cash_Flow	Categorical
Return_On_Equity	Categorical
Price_Trend	Categorical

Debt_Equity (D/E):

The debt-to-equity (D/E) ratio is calculated by dividing a company's total liabilities by its shareholder equity. These numbers are available on the balance sheet of a company's financial statements. The debt-to-equity ratio measures a company's debt relative to the value of its net assets, a high debt/equity ratio is often associated with high risk; it means that a company has been aggressive in financing its growth with debt.

$$\text{Debt/Equity} = \text{Total Liabilities} / \text{Total shareholder's Equity}$$



Cash Flow:

Cash flow is a measure of a company's financial health in terms of incomings and outgoings of cash, representing the operating activities of a company.



Cash from Operating Activities:

The operating activities on the CFS include any sources and uses of cash from business activities. In other words, it reflects how much cash is generated from a company's products or services.

Cash Flow from Investing Activities:

Cash flow from investing activities is one of the sections on the cash flow statement that reports how much cash has been generated or spent from various investment-related activities in a specific period. Investing activities include purchases of physical assets, investments in securities, or the sale of securities or assets.

Negative cash flow is often indicative of a company's poor performance. However, negative cash flow from investing activities might be due to significant amounts of cash being invested in the long-term health of the company, such as research and development.

Before analysing the different types of positive and negative cash flows from investing activities, it's important to review where a company's investment activity falls within its financial statements.

Cash Flow from Financing Activities:

Cash flow from financing activities (CFF) is a section of a company's cash flow statement, which shows the net flows of cash that are used to fund the company. Financing activities include transactions involving debt, equity, and dividends.

Cash flow from financing activities provides investors with insight into a company's financial strength and how well a company's capital structure is managed.

Formula:

$$\text{CFF} = \text{CED} - (\text{CD} + \text{RP})$$

Where:

CED = Cash in flows from issuing equity or debt

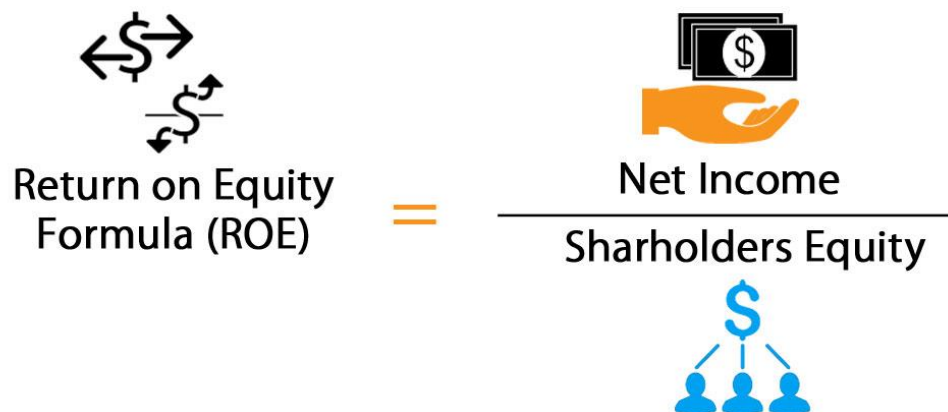
CD = Cash paid as dividends

RP = Repurchase of debt and equity

Return_on_Equity (ROE):

Return on equity is a measure of profitability based on how much profit a company generates with each dollar of stockholders' equity. ROE is considered a measure of how effectively management is using a company's assets to create profits.

$$\text{Return on Equity} = \text{Net Income} / \text{Average Shareholders' Equity}$$



$$\text{Return on Equity Formula (ROE)} = \frac{\text{Net Income}}{\text{Sharholders Equity}}$$

Price_Trend:

Price trend is the general direction and momentum of a market or of the price of security. If the price of security is going mainly upward, it is said to be on an upward price trend. The values of the dependable feature named "Price_Trend" consists of Profit and Loss labels. Profit indicates an upward price trend, whereas Loss indicates a downward price trend. The data set 2 is structured to suit the approach of the KNN-Probabilistic model.



Asset_Turnover:

The asset turnover ratio measures the value of a company's sales or revenues relative to the value of its assets. The asset turnover ratio can be used as an indicator of the efficiency with which a company is using its assets to generate revenue.

The higher the asset turnover ratio, the more efficient a company is at generating revenue from its assets. Conversely, if a company has a low asset turnover ratio, it indicates it is not efficiently using its assets to generate sales.



$$\text{Asset Turnover} = \frac{\text{Total Sales}}{\text{Beginning Assets} + \text{Ending Assets} / 2}$$

Where:

Total Sales=Annual sales total

Beginning Assets=Assets at start of year

Ending Assets=Assets at end of year

$$\text{Asset Turnover Ratio Formula} = \frac{\text{Net Sales}}{\text{Average Total Assets}}$$


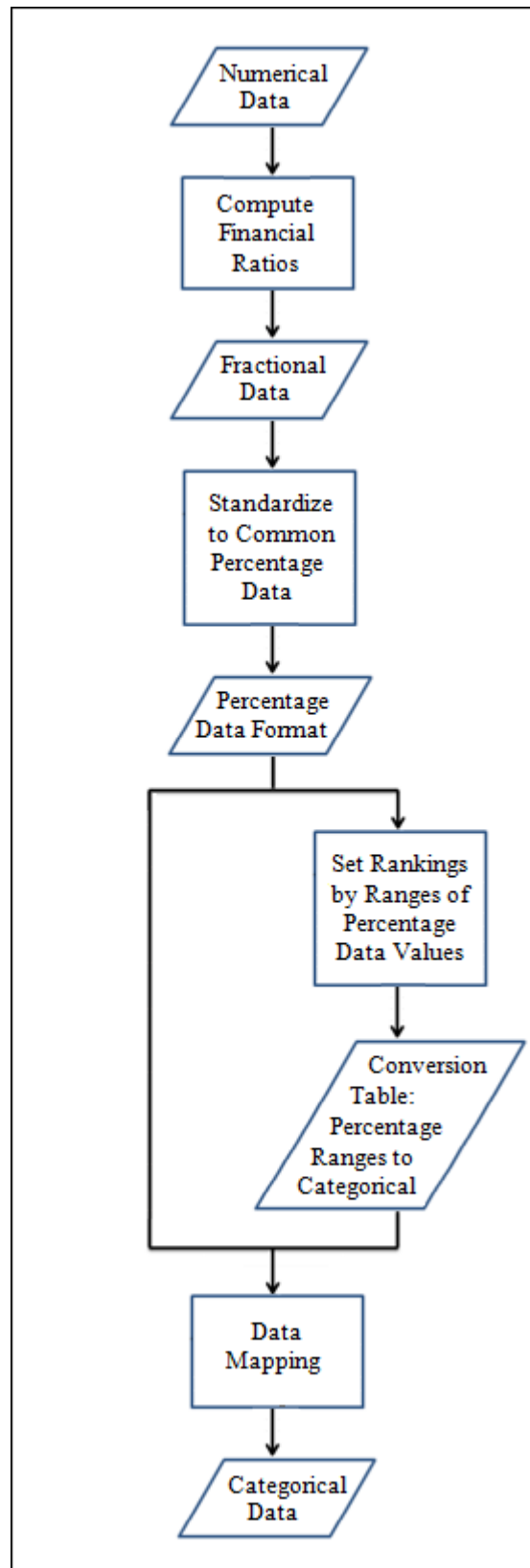


Figure 1: Flowchart of Data Transformation

2.2 Methods

The flow chart in Figure 2 illustrates the process flow of the proposed model. Using the parallel approach, the model starts with computing the prior probabilities and the probabilities based on KNN approach simultaneously on both the Profit class and Loss class. KNN initialization process involves the use of the k value for the nearest neighbors of test instances. KNN then calculates the number of Profit class and Loss class instances based on the k number of nearest neighbors in the vicinity of each test instance. The outcome generated from KNN is then used by the probabilistic method for further classification.

The probabilistic method calculates the prior probabilities of Profit class and Loss class based on the number of instances in the data set. The outcome from the earlier KNN approach is used as an input as the probabilities of Profit class and Loss class by the nearest neighbors method. The joint probabilities of Profit class and Loss class can then be calculated using the outcomes of the prior probabilities and the calculated KNN's probabilities. Finally, the predictive decision is made by comparing the joint probabilities of Profit class and Loss class.

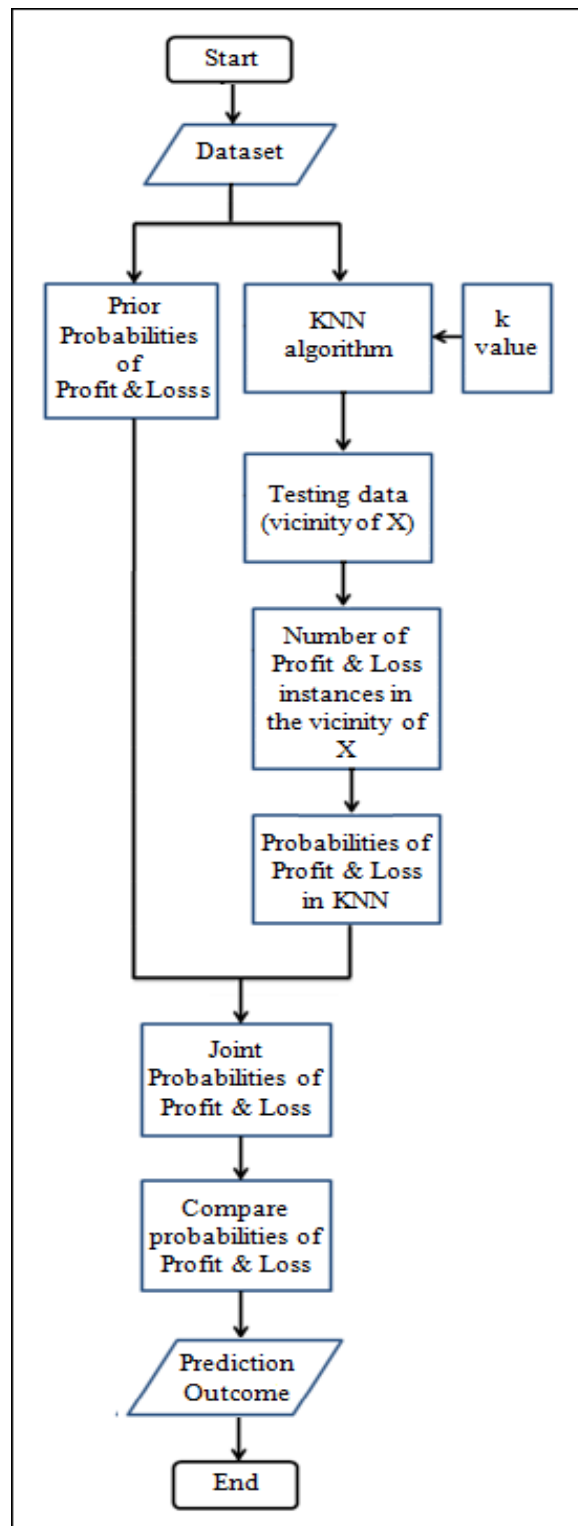


Figure 2: Flowchart of KNN-Probabilistic Model

The steps adopted for classification by KNN are illustrated as follows:

Steps:

- Classification: KNN
- Initialization of k value on nearest neighbors
- Compute the distance between the X query instance and all the training samples.
- Sort the distance values
- Determine the nearest neighbors to the query instance based on the k value
- Calculate the number of Profit instances of the nearest neighbors in the vicinity of X query instance
- Calculate the number of Loss instances of the nearest neighbors in the vicinity of X query instance

The steps adopted for classification by the probabilistic method is illustrated as follows:

Steps:

- Classification: Probabilistic method
- Calculate the prior probabilities of Profit class and Loss class from the data set
- Calculate the KNN's probabilities of Profit class and Loss class based on the number of Profit nearest neighbors and the number of Loss nearest neighbors.
- Calculate the joint probabilities from the prior probabilities and KNN's probabilities on Profit class and Loss class
- Compare the joint probabilities of Profit class and Loss class
- Select the predictive value from the class values with the highest joint probability

3. System Study

3.1 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are,

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**

3.1.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.1.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.1.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

4. SYSTEM REQUIREMENT SPECIFICATION

4.1 A Software Requirements Specification (SRS)

A requirements specification for a software system – is a complete description of the behaviour of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification

A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development life cycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- **Business requirements** describe in business terms *what* must be delivered or accomplished to provide value.
- **Product requirements** describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
- **Process requirements** describe activities performed by the developing organization. For instance, process requirements could specify specific methodologies that must be followed, and constraints that the organization must obey.

Product and process requirements are closely linked. Process requirements often specify the activities that will be performed to satisfy a product requirement. For example, a maximum development cost requirement (a process requirement) may be imposed to help achieve a maximum sales price requirement (a product requirement); a requirement that the product be maintainable (a Product requirement) often is addressed by imposing requirements to follow particular development styles.

4.2 Purpose

A systems engineering, a requirement can be a description of *what* a system must do, referred to as a Functional Requirement. This type of requirement specifies something that the delivered system must be able to do. Another type of requirement specifies something about the system itself, and how well it performs its functions. Such requirements are often called Non-functional requirements, or 'performance requirements' or 'quality of service requirements.' Examples of such requirements include usability, availability, reliability, supportability, testability and maintainability.

A collection of requirements define the characteristics or features of the desired system. A 'good' list of requirements as far as possible avoids saying *how* the system should implement the requirements, leaving such decisions to the system designer. Specifying how the system should be implemented is called "implementation bias" or "solution engineering". However, implementation constraints on the solution may validly be expressed by the future owner, for example for required interfaces to external systems; for interoperability with other systems; and for commonality (e.g. of user interfaces) with other owned products.

In software engineering, the same meanings of requirements apply, except that the focus of interest is the software itself.

4.3 REQUIREMENT ANALYSIS

The project involved analyzing the design of few applications so as to make the application more users friendly. To do so, it was really important to keep the navigations from one screen to the other well-ordered and at the same time reducing the amount of typing the user needs to do. In order to make the application more accessible, the browser version had to be chosen so that it is compatible with most of the Browsers.

4.3.1 REQUIREMENT SPECIFICATION

➤ **Functional Requirements**

- Graphical User interface with the User.

➤ **Non Functional Requirements**

The major non-functional Requirements of the system are as follows

- **Usability**

The system is designed with completely automated process hence there is no or less user intervention.

- **Reliability**

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using java is more reliable.

- **Performance**

This system is developing in the high level languages and using the advanced front-end and back-end technologies it will give response to the end user on client system with in very less time.

4.4 Software and Hardware Requirements

➤ **Software Requirements**

For developing the application the following are the Software Requirements:

- Python

➤ **Operating Systems supported**

- Windows

➤ **Technologies and Languages used to Develop**

- Python

➤ **Hardware Requirements**

For developing the application the following are the Hardware Requirements:

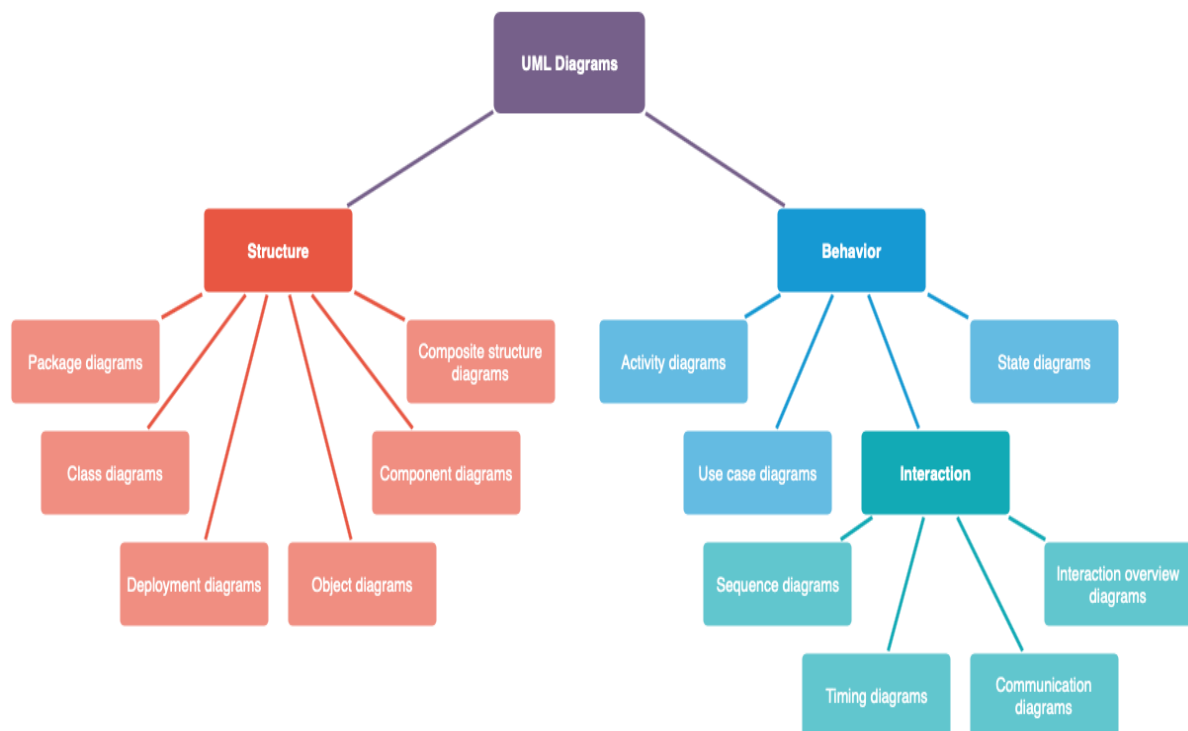
- Processor: Pentium IV or higher
- RAM: 2GB
- Space on Hard Disk: minimum 10GB

4.5 UML Diagrams:

A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. Use Case diagrams are formally included in two modelling languages defined by the OMG: the Unified Modelling Language (UML) and the Systems Modelling Language (Sys ML).

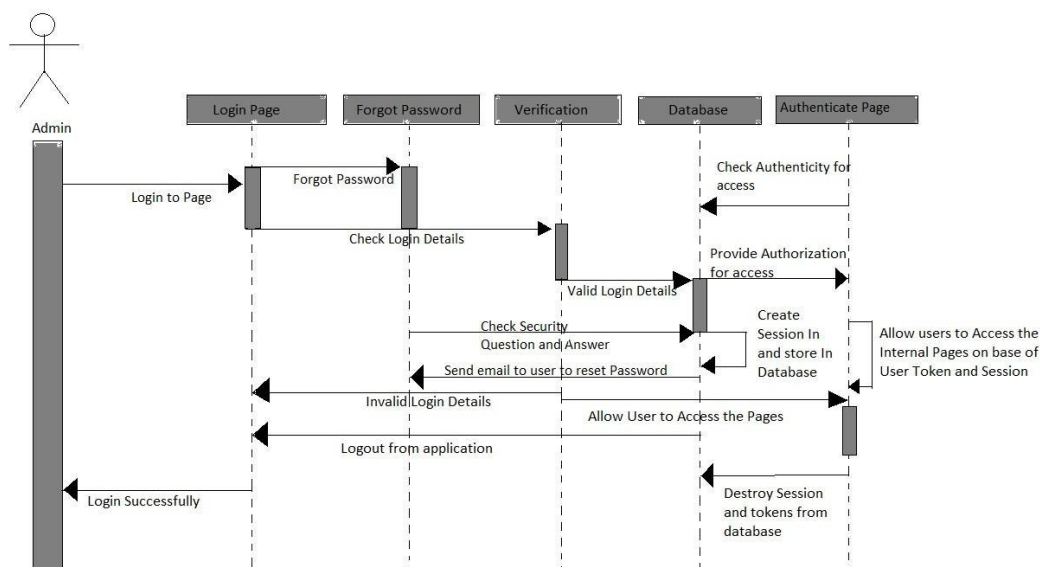
Types of UML Diagrams

There are several types of UML diagrams and each one of them serves a different purpose. The two most broad categories that encompass all other types are **Behavioral** UML diagram and **Structural** UML diagram. As the name suggests, some UML diagrams try to analyze and depict the structure of a system or process, whereas other describe the behavior of the system, its actors, and its building components. The different types are broken down as follows:



Activity Diagram:

An activity diagram is essentially a flowchart, showing flow of controls from one activity to another. Unlike a traditional flowchart, it can model the dynamic functional view of a system. An activity diagram represents an operation on some classes in the system that results to changes in the state of the system.



From the diagram, the client is expected to provide the input dataset and the required output. The required output is used in back propagation, for the system uses it to compare its predicted value from time to time in order to get the optimal prediction. The client selects a threshold constant and looking at the input value the neural network initializes the weight for the input layer and the hidden layer. Then the calculation for the activation function of both the input and the output layer is done and the system calculates the error. If the error is large then the system adjusts the weight and backpropagates it to the input layer for further calculation to be done. The system outputs its prediction whenever the error is small.

Use Case Diagram:

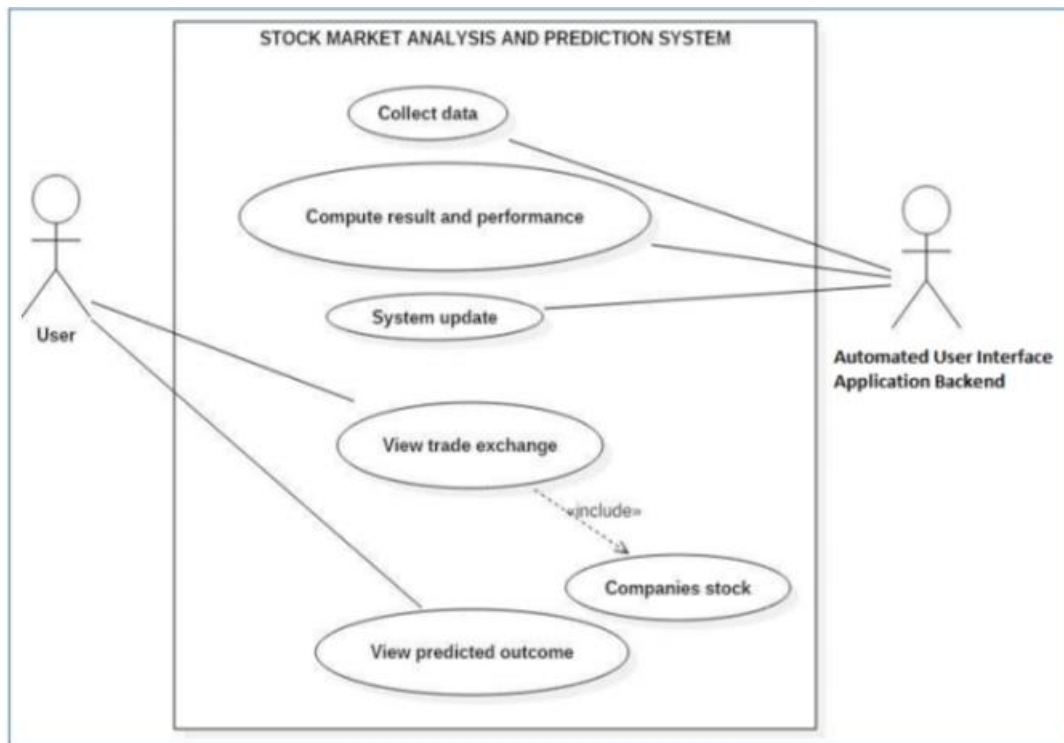
A cornerstone part of the system is the functional requirements that the system fulfills. Use Case diagrams are used to analyze the system's high-level requirements. These requirements are expressed through different use cases. We notice three main components of this UML diagram:

Functional requirements – represented as use cases; a verb describing an action

Actors – they interact with the system; an actor can be a human being, an organization or an internal or external application

Relationships between actors and use cases – represented using straight arrows

The Case Diagram of the Proposed System The case diagram of the proposed system is shown below. The proposed system allows the user to provide the training data and the threshold constant. Training Neural network requires initialization of input weights and the first input weight will be adjusted until the optimal prediction is achieved.



Use Case ID	Use Case Name	Primary actor	Scope	Complexity	Priority
1	Collect data	Admin	In	High	1
2	Compute result and performance	Admin	In	High	1
3	System update	Admin	In	High	1
4	View traded exchange	User	In	Medium	2
5	Company Stock	User	In	Medium	2
6	View predicted outcome	User	In	High	1

Within the circular containers, we express the actions that the actors perform. Such actions are: purchasing and paying for the stock, checking stock quality, returning the stock or distributing it. As you might have noticed, use case UML diagrams are good for showing dynamic behaviours between actors within a system, by simplifying the view of the system and not reflecting the details of implementation.

Use case description:

Use case ID:1

Use case name: Collect data

Description: Every required data will be available in Yahoo stock exchange. Admin will be able to collect the data for system.

Use case ID:2

Use case name: Compute result and performance

Description: Prediction result will be handled and generated by admin. The system will be built, through which the result of prediction and system performance will be analyzed.

Use case ID: 3

Use case name: System update

Description: With the change of market and technology regular update of system is required. Beside there the predict result of stock exchange and their actual price will be updated by admin in regular basis.

Use case ID: 4

Use case name: View traded exchange

Description: Company trading which is held at Yahoo can be viewed by user.

Use Case ID: 5

Use Case Name: Company Stock

Description: It is extended feature of view traded exchange. This includes the stock value of particular company.

Use Case ID: 6

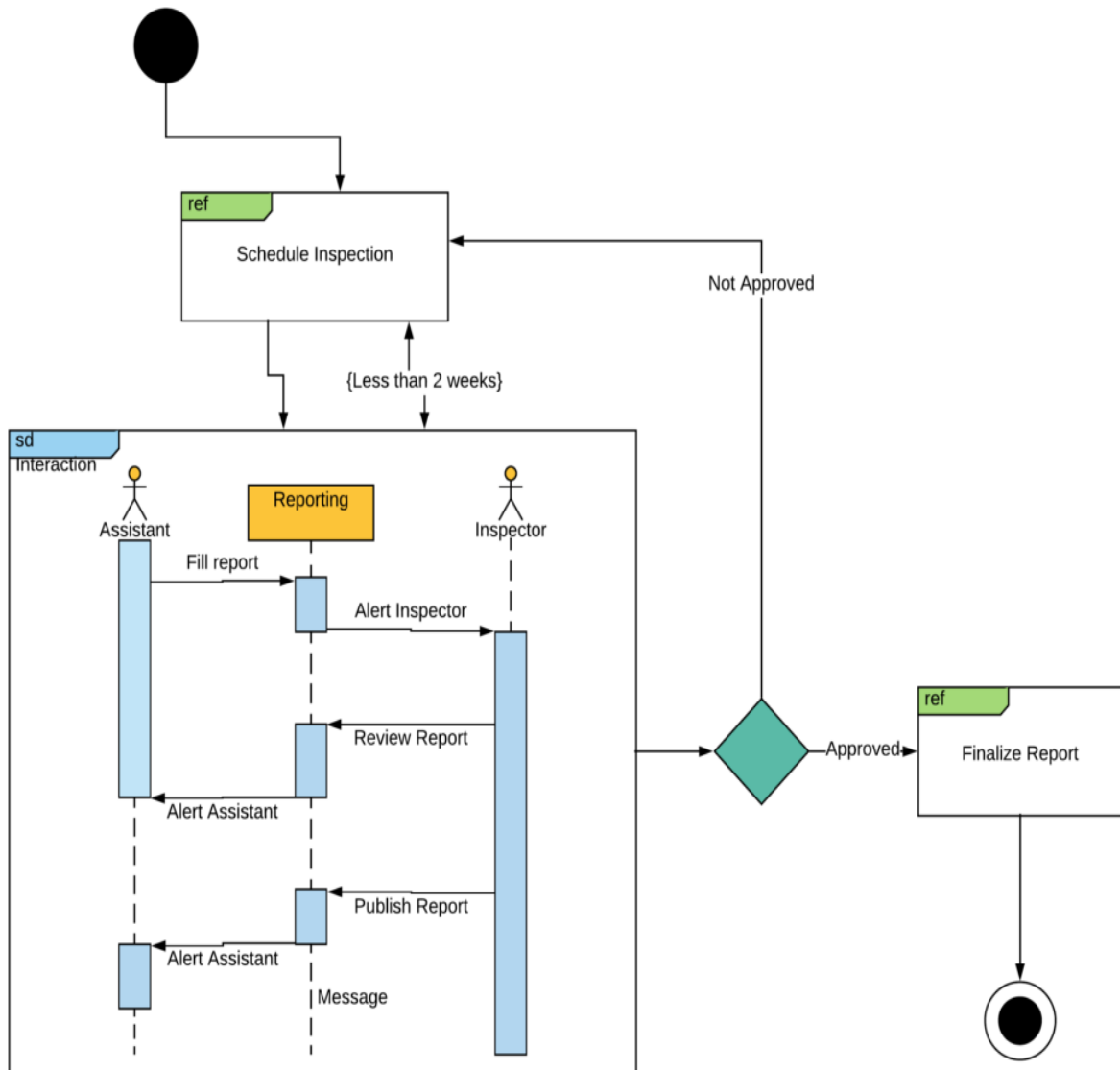
Use Case Name: View predicted outcome

Description: This use case is must important in whole project. The key feature of this project is to predict the stock value of hydropower companies. Thus, this will be available in user interface and viewer can observe them.

Interaction Overview Diagram:

Interaction Overview UML diagrams are probably some of the most complex ones. So far we have explained what an activity diagram is. Additionally, within the set of behavioral diagrams, we have a subset made of four diagrams, called Interaction Diagrams:

- Interaction Overview Diagram
- Timing Diagram
- Sequence Diagram
- Communication Diagram



Timing diagram:

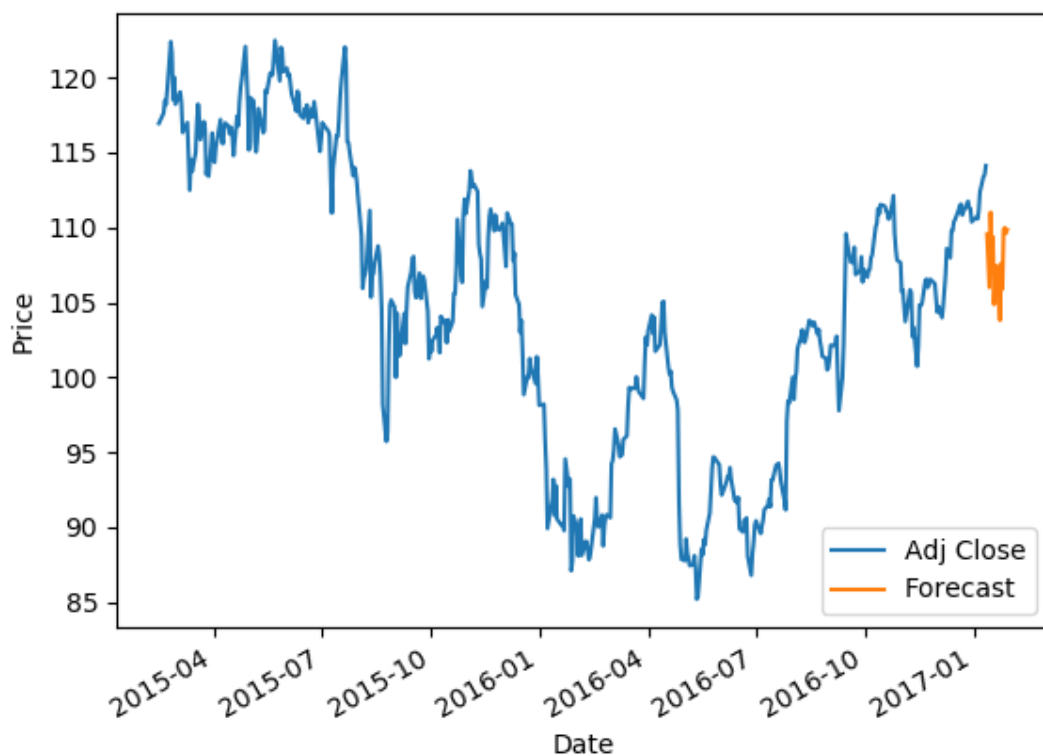
Timing UML diagrams are used to represent the relations of objects when the center of attention rests on time. We are not interested in how the objects interact or change each other, but rather we want to represent how objects and actors act along a linear time axis.

Each individual participant is represented through a lifeline, which is essentially a line forming steps since the individual participant transits from one stage to another. The main focus is on time duration of events and the changes that occur depending on the duration constraints.

The main components of a timing UML diagram are:

- **Lifeline** – individual participant
- **State timeline** – a single lifeline can go through different states within a pipeline
- **Duration constraint** – a time interval constraint that represents the duration of necessary for a constraint to be fulfilled
- **Time constraint** – a time interval constraint during which something needs to be fulfilled by the participant
- **Destruction occurrence** – a message occurrence that destroys the individual participant and depicts the end of that participant's lifeline

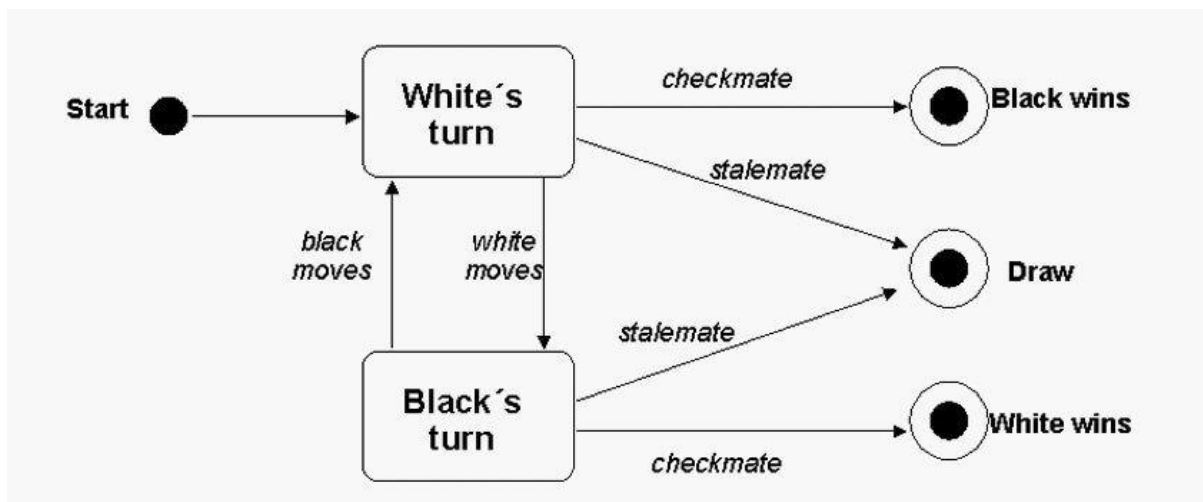
An example of a simplified timing UML diagram is given below.



State Machine UML diagram:

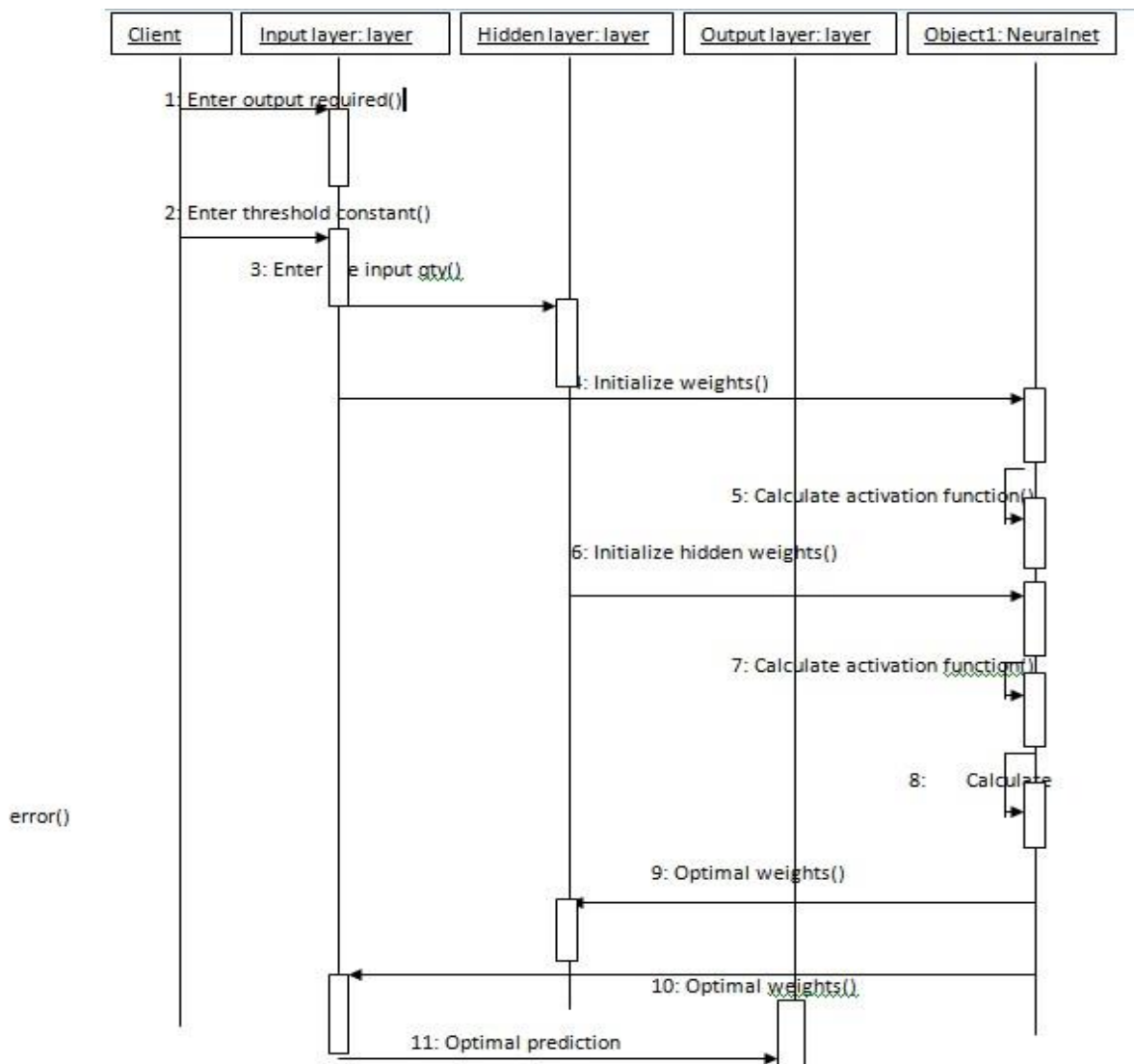
State machine UML diagrams, also referred to as State chart diagrams, are used to describe the different states of a component within a system. It takes the name state machine because the diagram is essentially a machine that describes the several states of an object and how it changes based on internal and external events.

A very simple state machine diagram would be that of a chess game. A typical chess game consists of moves made by White and moves made by Black. White gets to have the first move and thus initiates the game. The conclusion of the game can occur regardless of whether it is the White's turn or the Black's. The game can end with a checkmate, resignation or in a draw (different states of the machine).



Sequence UML Diagram:

Sequence diagrams are probably the most important UML diagrams among not only the computer science community but also as design-level models for business application development. The sequence diagram numbers the actions starting from the data input to the optimal prediction. There is an arrow direction to show the sequence of flow for the action taking to arrive at the optimal prediction.

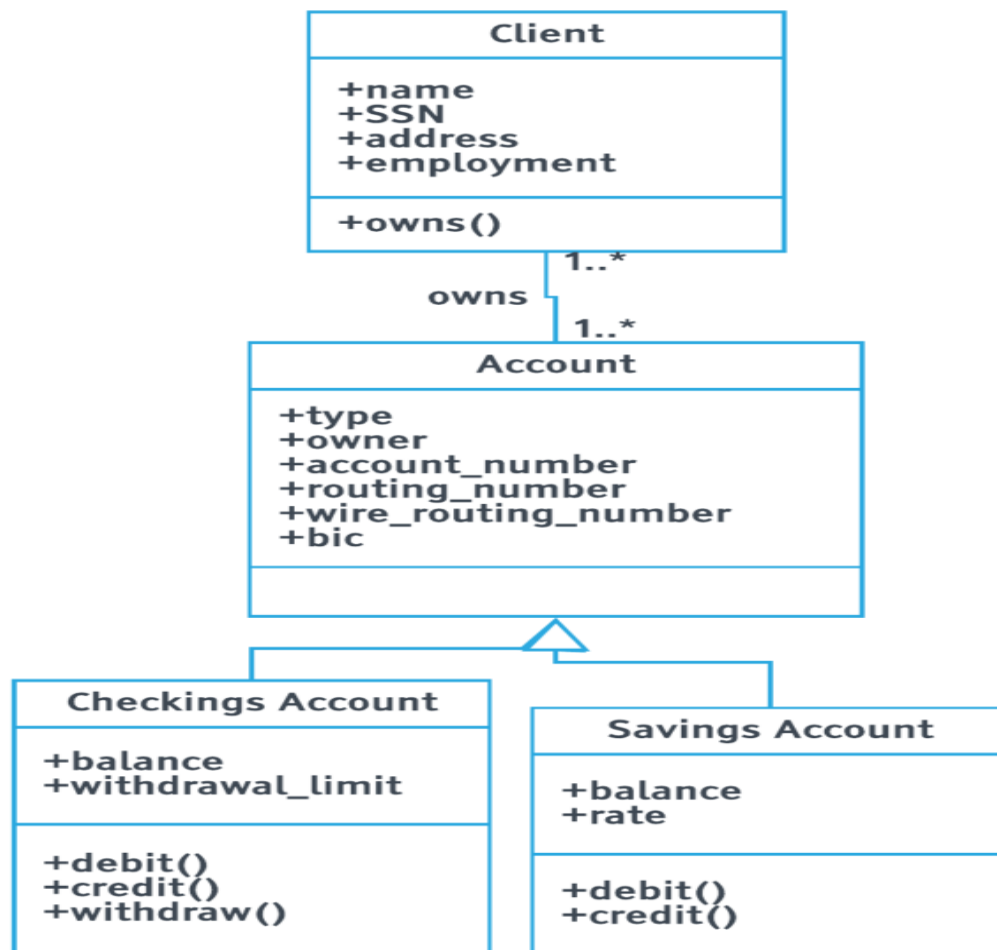


Class Diagram:

Class UML diagram is the most common diagram type for software documentation. Since most software being created nowadays is still based on the Object-Oriented Programming paradigm, using class diagrams to document the software turns out to be a common-sense solution. This happens because OOP is based on classes and the relations between them.

In a nutshell, class diagrams contain classes, alongside with their attributes (also referred to as data fields) and their behaviors (also referred to as member functions). More specifically, each class has 3 fields: the class name at the top, the class attributes right below

the name, the class operations/behaviors at the bottom. The relation between different classes (represented by a connecting line), makes up a class diagram.



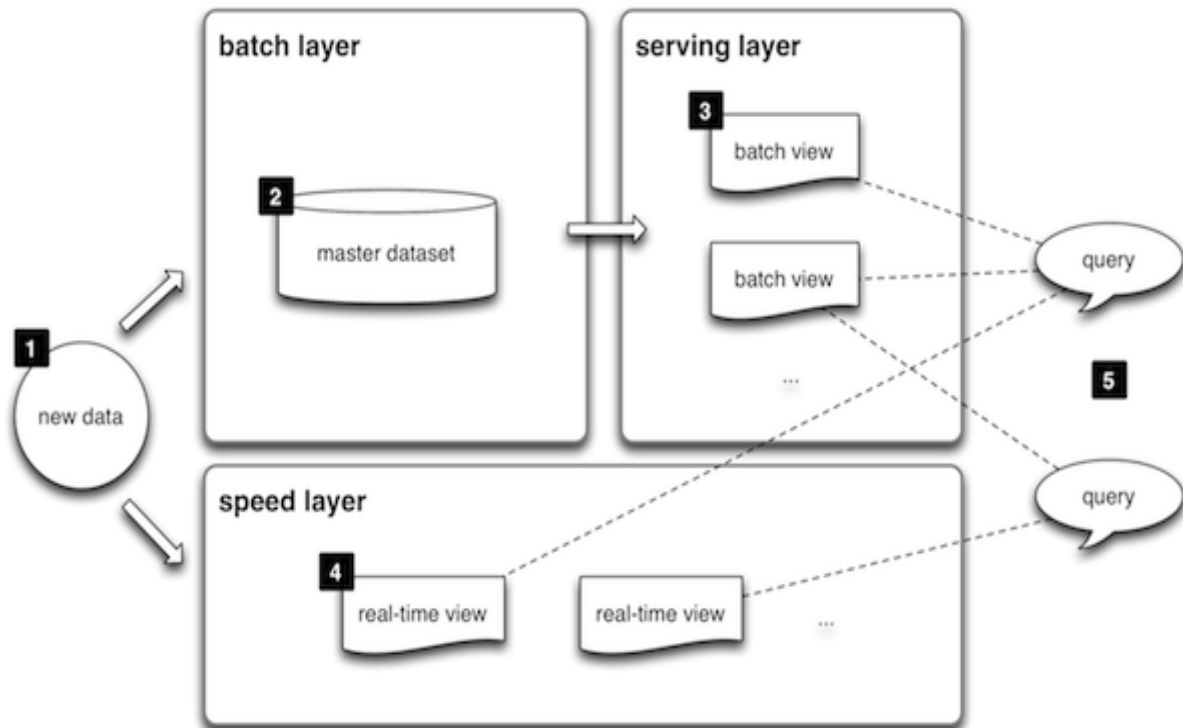
The example above shows a basic class diagram. The ‘Checking’s Account’ class and the ‘Savings Account’ class both inherit from the more general class, ‘Account’. The inheritance is shown using the blank-headed arrow.

Component Diagram:

When dealing with documentation of complex systems, component UML diagrams can help break down the system into smaller components. Sometimes it is hard to depict the architecture of a system because it might encompass several departments or it might employ different technologies.

For example, Lambda architecture is the typical example of a complex architecture that can be represented using a component UML diagram. Lambda architecture is a data-

processing architecture employed by several companies for storing and processing data in a distributed system. It is made up of three different layers: the speed layer, the batch layer and the serving one.



The image above shows how a component diagram can help us get a simplified top-level view of a more complex system. The annotations used here are not tailored according to UML standards.

Deployment Diagram:

Deployment diagrams are used to visualize the relation between software and hardware. To be more specific, with deployment diagrams we can construct a physical model of how software components (artifacts) are deployed on hardware components, known as nodes.

A typical simplified deployment diagram for a web application would include:

- **Nodes** (application server and database server)
- **Artifacts** (application client and database schema)

The nodes host the artifacts. The database schema runs on the database server and the application client runs on the application server.

5. SOFTWARE ENVIRONMENT

5.1 PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating systems. Python, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. Python is managed by the non-profit Python Software Foundation. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

a. Matplotlib:

Matplotlib helps with data analysing, and is a numerical plotting library. We talked about it in Python for Data Science.

b. Pandas:

Like we've said before, Pandas is a must for data-science. It provides fast, expressive, and flexible data structures to easily (and intuitively) work with structured (tabular, multidimensional, potentially heterogeneous) and time-series data.

c. Scrapy:

If your motive is fast, high-level screen scraping and web crawling, go for Scrapy. You can use it for purposes from data mining to monitoring and automated testing.

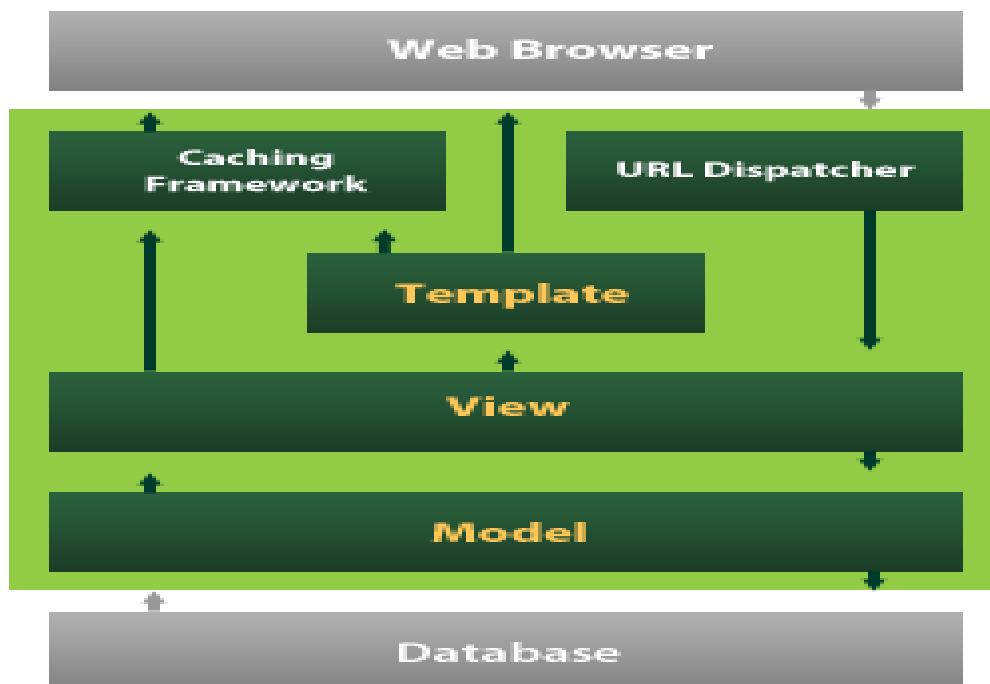
What can Python do

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

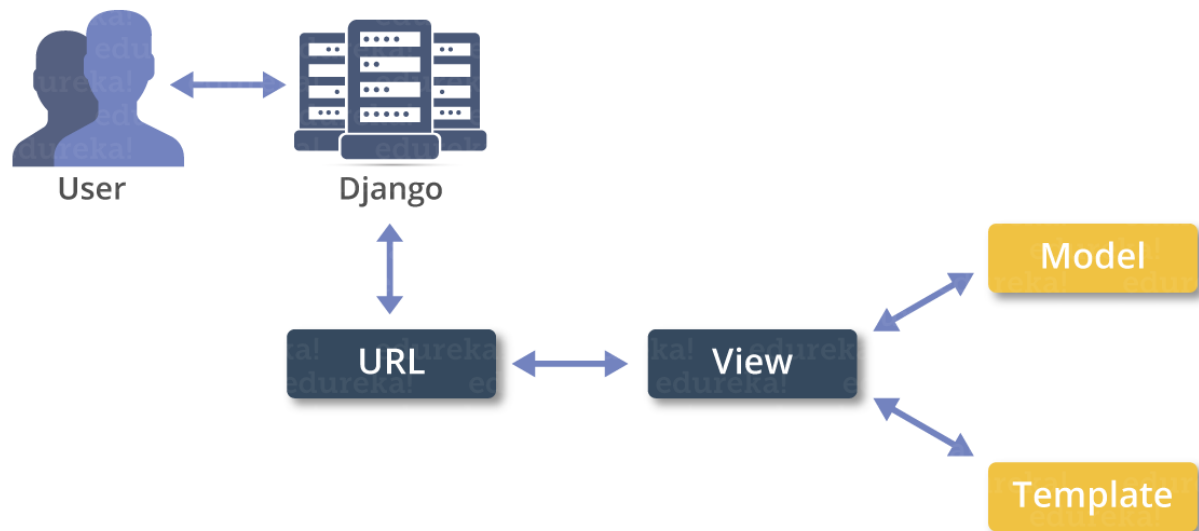
5.2 DJANGO

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes reusability and "pluggability" of components, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models.



Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models



6. INPUT AND OUTPUT DESIGN

6.1 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

What data should be given as input?

How the data should be arranged or coded?

The dialog to guide the operating personnel in providing input.

Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

6.2 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis

design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

7. SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.1 TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a

business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centred on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of

system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

7.2 Verification and Validation

The testing process is a part of broader subject referring to verification and validation. We have to acknowledge the system specifications and try to meet the customer's requirements and for this sole purpose, we have to verify and validate the product to make sure everything is in place. Verification and validation are two different things. One is performed to ensure that the software correctly implements a specific functionality and other is done to ensure if the customer requirements are properly met or not by the end product. Verification of the project was carried out to ensure that the project met all the requirement and specification of our project. We made sure that our project is up to the standard as we planned at the beginning of our project development.

8. RESULTS AND DISCUSSION

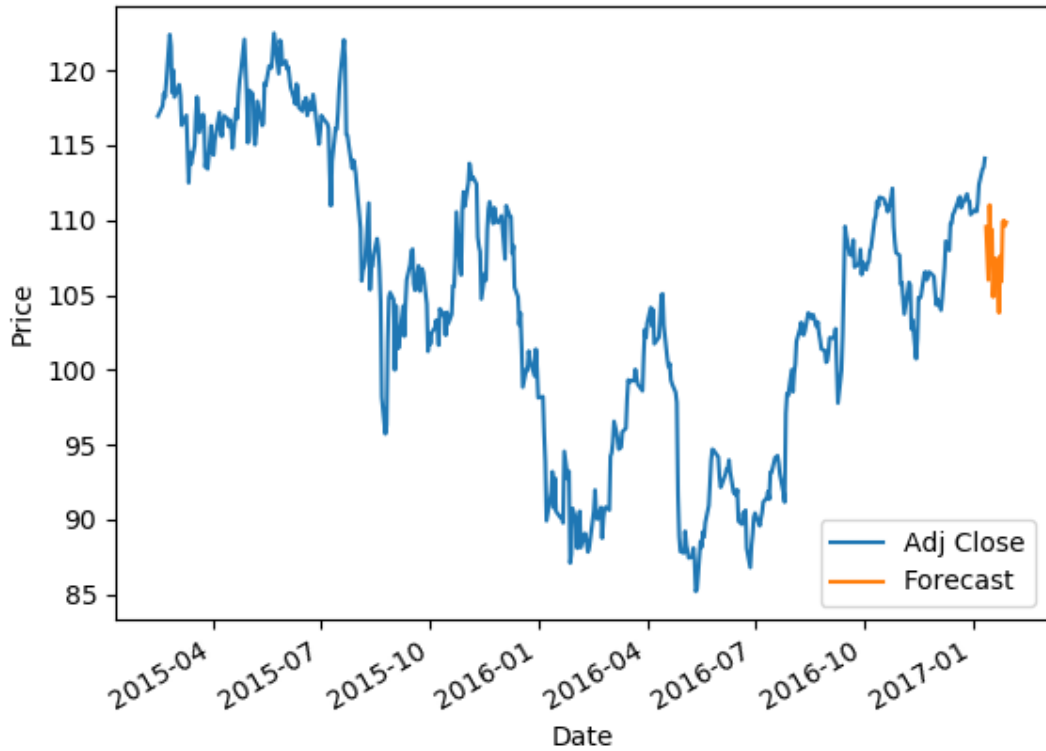
8.1 Results

The proposed model was tested and compared with four other standard algorithms, including KNN, Naïve Bayes, OneR and ZeroR. The test examined how accurate the tested algorithms predict the stock price trends, and evaluated the MAE and RMSE. Table 5 presents the test results. The hybrid KNN-Probabilistic model has allowed us to achieve an estimated accuracy of 89.1725%, exceeding the stand alone KNN reported accuracy of 86.6667% and the Naive Bayes accuracy of 76.1194%. The accuracy rates for OneR and ZeroR classifiers were 71.6418% and 64.1791% respectively. KNN-Probabilistic model has MAE rate of 0.0667% and RMSE rate of 0.2582% which are much lower than the other classifiers.

Table 5: Prediction Results of Classifiers.

Classifier	Accuracy (%)	MAE	RMSE
KNN-Probabilistic	93.3333	0.0667	0.2582
KNN	86.6667	0.1333	0.3651
Naive Bayes	76.1194	0.1726	0.2824
OneR	71.6418	0.5325	0.6139
ZeroR	64.1791	0.4619	0.4805

Overall, KNN-Probabilistic model has better accuracy rate and error rates than the other classifiers used for comparisons. The test demonstrated that the hybrid mechanism of KNN and probabilistic method produced significantly improved results, compared with each of the KNN and Naïve Bayes classifiers.



8.2 Discussion

The proposed method begins with processing the data using data set 2, with each record contains a stock's financial features and the predicted outcomes in a structured categorical format. Using these records as inputs, stock price trends were predicted using the proposed hybrid KNN-Probabilistic model.

For KNN, the features in the data set are the data points in metric space with notion of distance. Each of the data set record contains a set of vectors and class labels associated with each vector. Each class label is either labeled as Profit for positive class or is labeled as Loss for negative class. The k value decides how many neighbors that can influence the classification. Initial step in KNN is to determine the appropriate k value. The k value is very training-data dependent. A small k value means that noise will have a higher influence on the result and a large value creates an overfit model. The use of k -fold cross-validation indicates the k value led to the highest classified generalizability. Typically, odd number is used as k

value when the number of classes is two, so that a decision can be determined based on the class value with the higher number of instances.

For KNN-Probabilistic model, an odd number k value is not required because a decision for prediction is not made at the initial stage of KNN classifier. A ' k ' value of even number is used to prevent unnecessary bias of unequal representations of the two classes at the stage of KNN method. A decision for prediction will be made based on the combined outcomes of the KNN method and the probabilistic method. The KNN method determines the class instances that form the initial probabilities from the nearest neighbors' perspective. The probabilistic method makes use of a combination of probabilities in its decision making. When the inputs for prior probability and probability based on KNN are available, the predictive model can calculate the joint probability and make prediction on the class outcome.

The probabilistic model includes some functional relations between the unknown parameters and the observed data to allow us to make predictions. The goal of this statistical analysis is to estimate the unknown parameters in the proposed model. Initial stage includes identifying the optimal value for the ' k ' parameter. The computations used in the model include prior probability, probability in KNN and joint probability. The model has the following estimating computations.

The probability estimations based on KNN are,

- The probability of Profit = the number of Profit instances of the nearest neighbors in the vicinity of the query instance / Number of ' k ' instances.
- The probability of Loss = the number of Loss instances of the nearest neighbors in the vicinity of the query instance / Number of ' k ' instances.
- The probabilities in KNN measures the support provided by the data for each possible value of the ' k ' parameter of KNN.

The computations of prior probability are,

- The prior probability of Profit = Total number of Profit instances / Total number of all instances

- The prior probability of Loss = Total number of Loss instances / Total number of all instances

The computations of joint probability are,

- Joint probability of Profit = the probability of Profit based on KNN \times the prior probability of Profit.
- Joint probability of Loss = the probability of Loss based on KNN \times the prior probability of Loss.

The steps of the process used for probability comparison is,

- If Joint Probability of Profit $>$ Joint Probability of Loss
Then prediction = Profit
- Else If Joint Probability of Loss $>$ Joint Probability of Profit
Then prediction = Loss
- Else If Joint Probability of Profit $==$ Joint Probability of Loss
Then repeat and re-adjust the k parameter.

9. SCREEN SHOTS

Double click on 'run.bat' file to get below screen

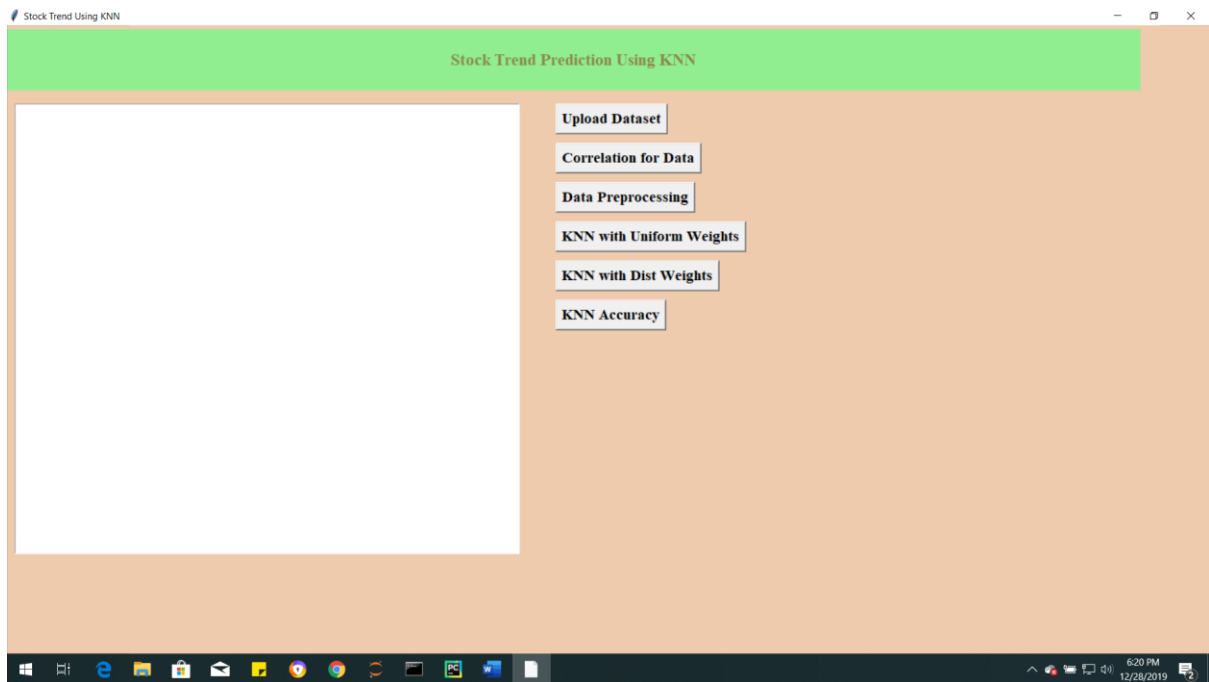


Figure 9.1 Stock Trend Prediction Using KNN

In above screen click on 'Download Button' download the Apple Stock and competitors data from Yahoo Finance Dataset

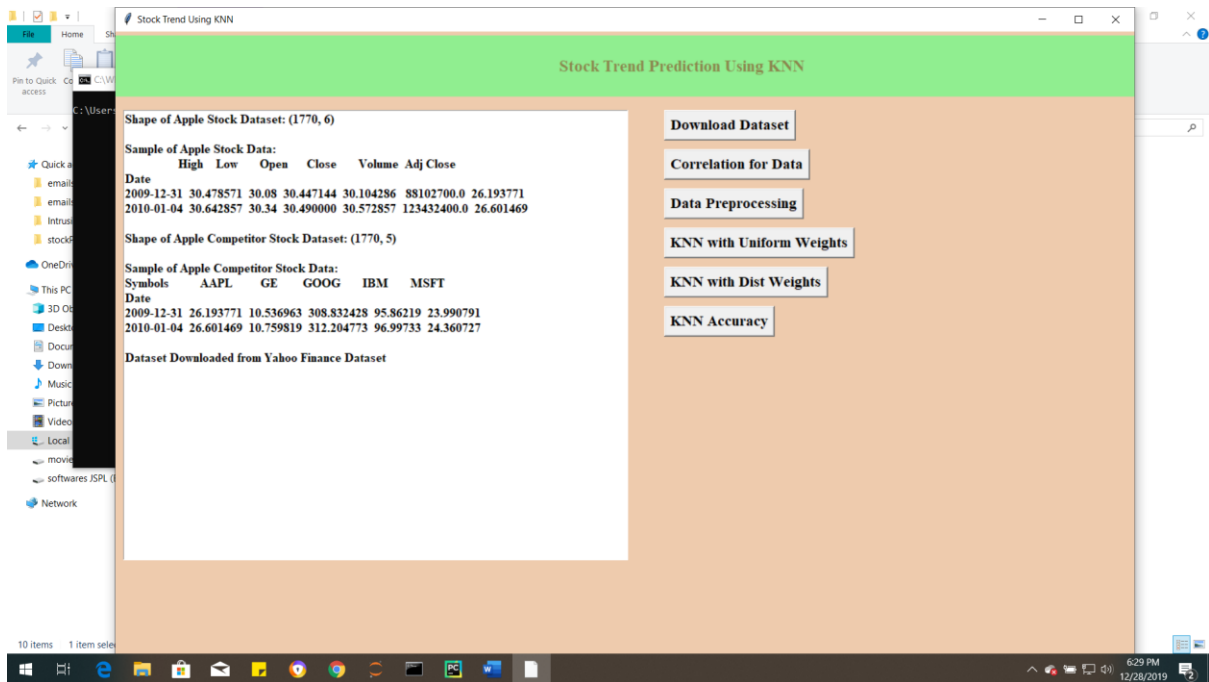


Figure 9.2 Download Dataset

In above screen I am Downloading of Apple Stock and Apple competitor Stock Data from Yahoo Finance Dataset.

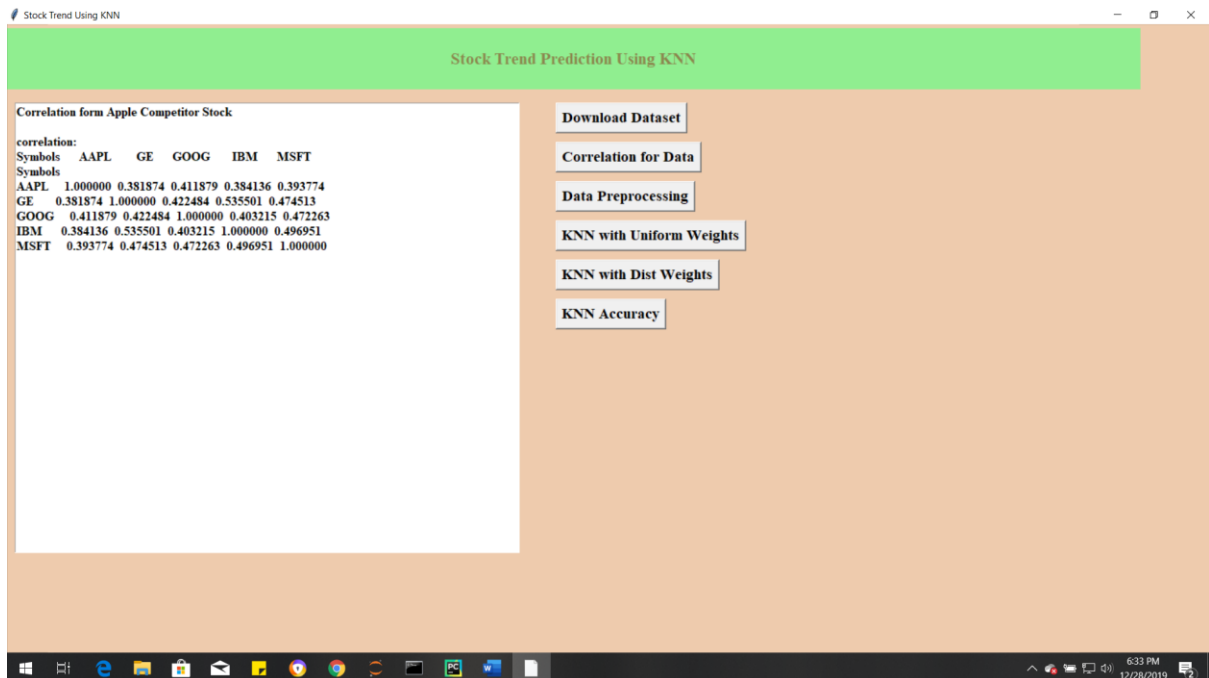


Figure 9.3 Correlations for Data

Now click on ‘Correlation Data’ Button to find the correlation between Apple and Competitor Stock market Dataset.

Show the trend in the technology industry rather than show how competing stocks affect each other.

Now click on ‘Data Pre-processing button to drop missing values, split labels split train and test

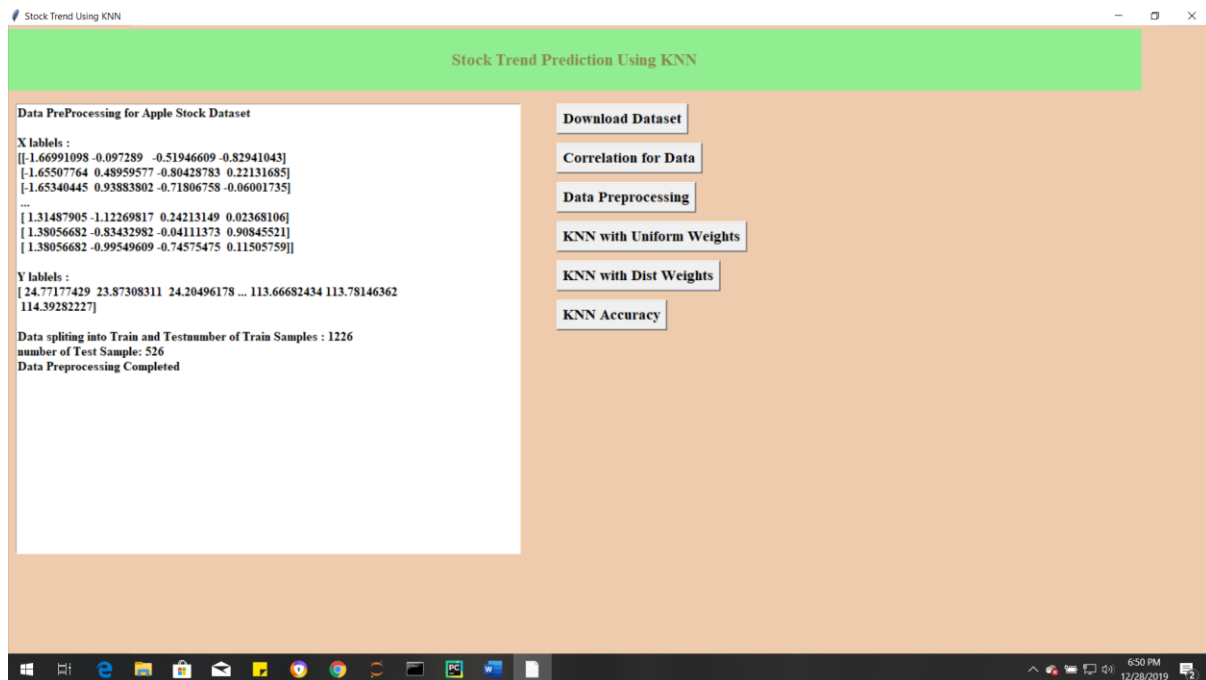


Figure 9.4 Data Pre-processing

After pre-processing all missing values are dropped, Separating the label here, Scalling of X, find Data Series of late X and early X (train) for model generation and evaluation, Separate label and identify it as y and Separation of training and testing of model.

In above screen we can see dataset contains total 1752 records and 1226 used for training and 526 used for testing.

Now click on ‘Run KNN with Uniform Weights’ to generate KNN model with uniform weights and calculate its model accuracy

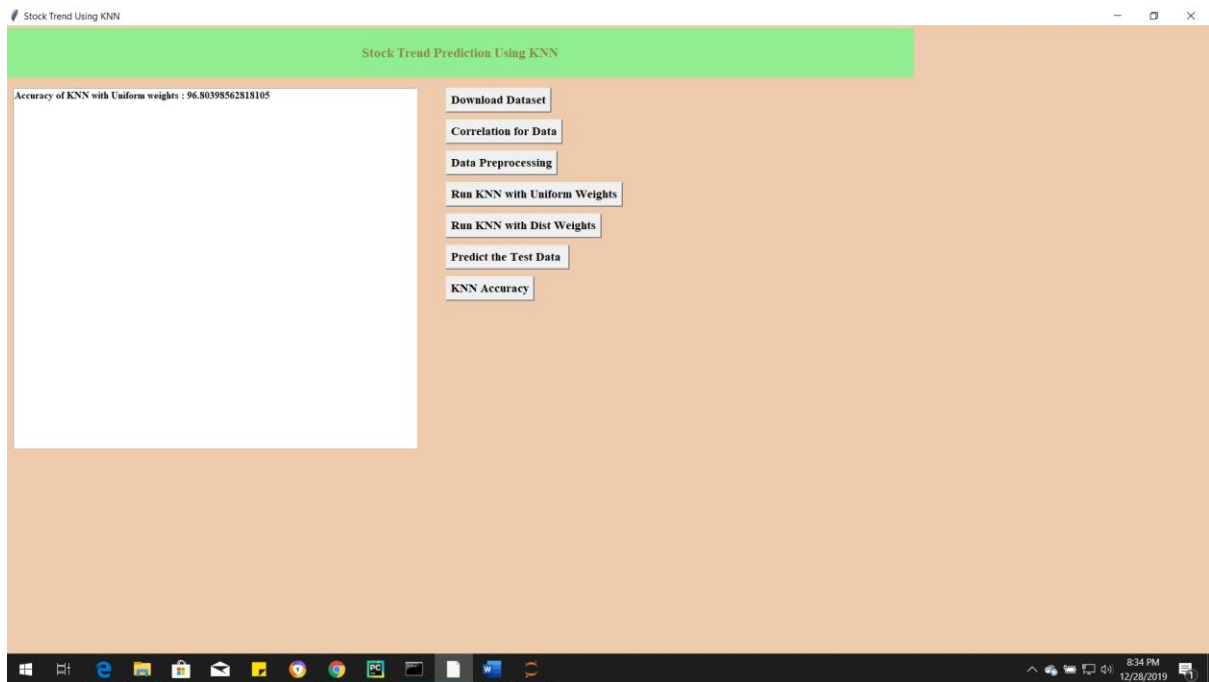


Figure 9.5 Run KNN with Uniform weight

In above screen we can see with KNN with uniform weights got 96.8% accuracy, now click on ‘Run KNN with distance weights’ to calculate accuracy

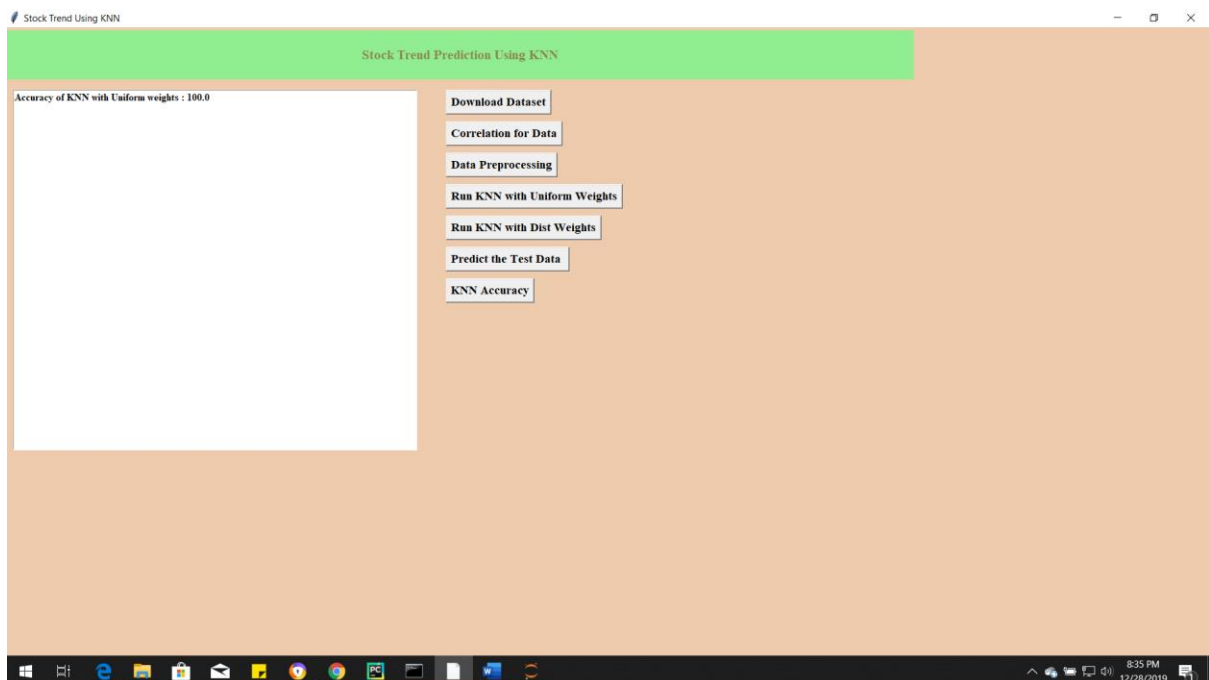


Figure 9.6 Run KNN with Dist. weight

In above screen we got 100% accuracy, now we will click on ‘Predict Test Data’ button to upload test data and to predict whether test data stock market for both models.

Accuracy score (>0.95) for most of the models.

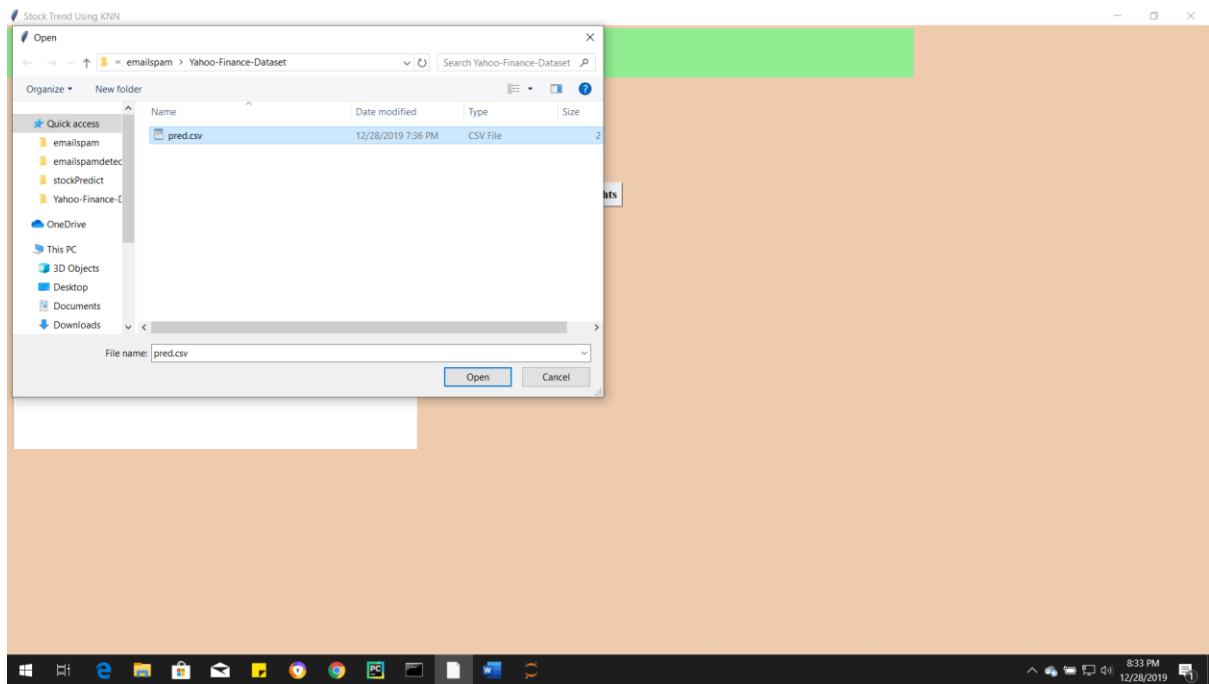


Figure 9.7 Test Data Upload

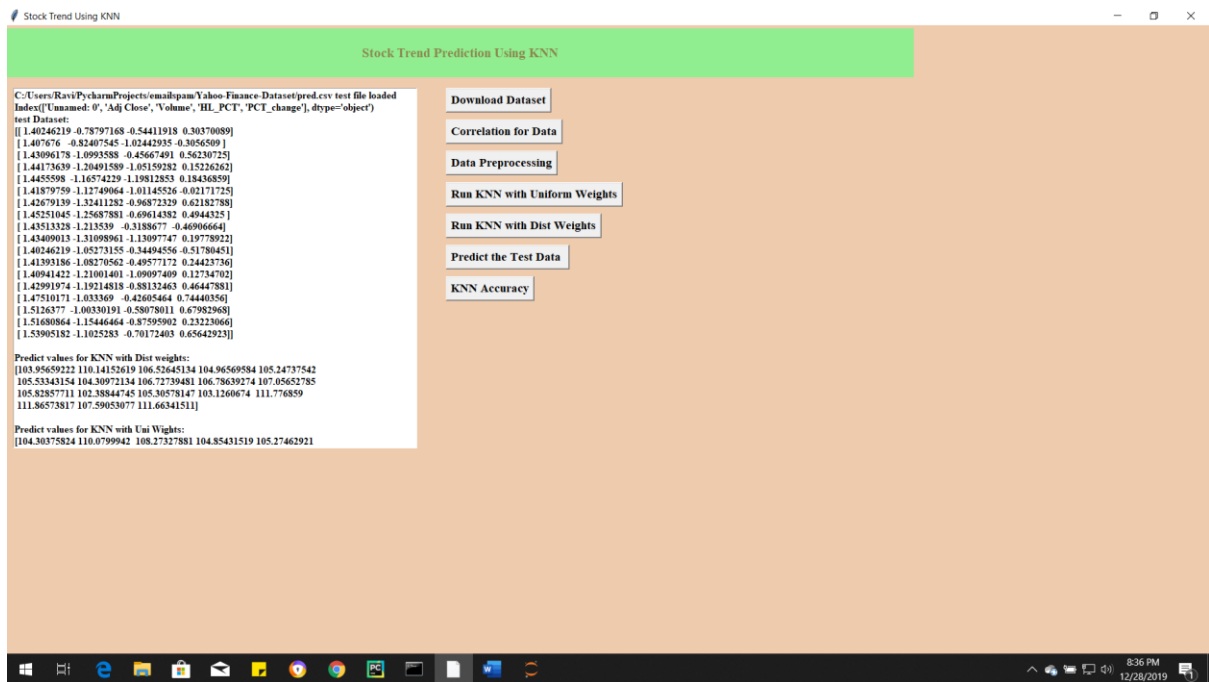


Figure 9.8 KNN Accuracy

In above screen for each test data we got forecast values for Apple Stock for each test record. Now click on ‘KNN Accuracy’ button to save the predicted values for each model save in the local directory and Accuracy comparison to both the models

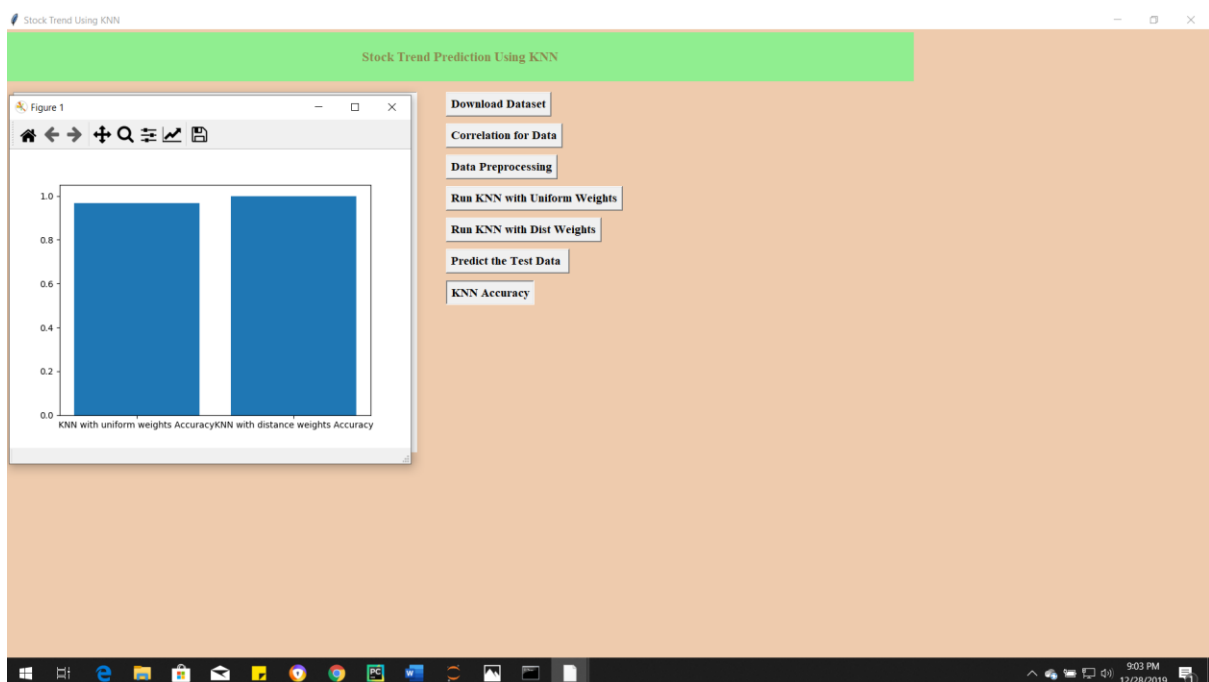
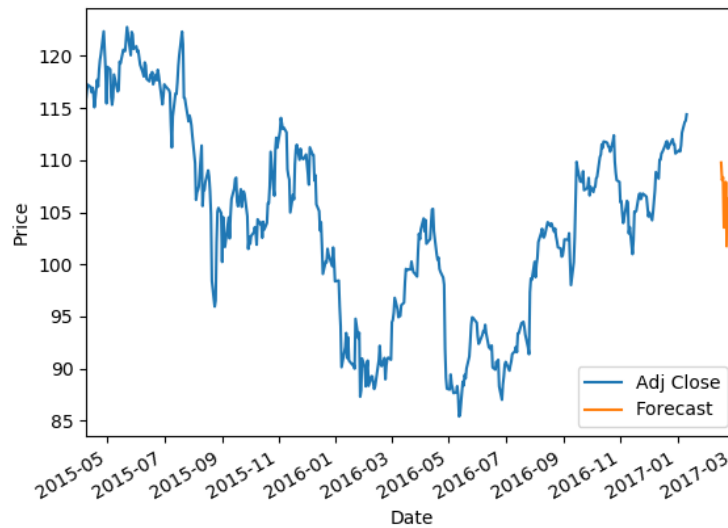


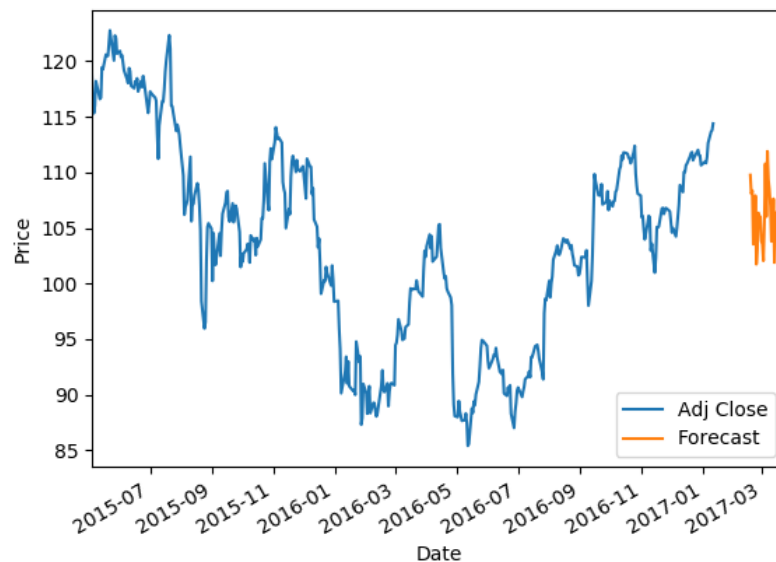
Figure 9.9 Test Data Results

From above graph we can see that distance weights has little bit better accuracy compare to Uniform weights, in above graph x-axis contains algorithm name and y-axis represents accuracy of that algorithms

Plotting the Prediction for KNN with Uniform Weights:



Plotting the Prediction for KNN with Distance Weights:



10. CONCLUSION

The aim of this research is to improve the statistical fitness of the proposed model to overcome a KNN problem due to its computation approach. The KNN classifier can compute the empirical distribution over the Profit and Loss class values in the k number of nearest neighbors. However, the outcome is less than adequate due to sparse data. The KNN classifier has under fitting issue as it does not cater to generalization of sparse data outside the range of nearest neighborhood.

We have compared a hybrid KNN-Probabilistic model with four standard algorithms on the problem of predicting the stock price trends. Our results showed that the proposed KNN-Probabilistic model leads to significantly better results compared to the standard KNN algorithm and the other classification algorithms.

The limitation of the proposed model is that it applies a binary classification technique. The actual output of this binary classification model is a prediction score in two-class. The score indicates the model's certainty that the given observation belongs to either the Profit class or Loss class. For future work, the knowledge component is to transform the binary classification into multiclass classification. The multiclass classification involves observation and analysis of more than the existing two statistical class values. Additional research will include the application of the probabilistic model to multiclass data in order to provide more specific information of each class value. The newly formed multiclass classification will contain five class labels named “Sell”, “Underperform”, “Hold”, “Outperform”, and “Buy”. In numerical values for mapping purpose, we will convert “Sell” to -2 which implies strongly unfavorable; “Underperform” to -1 which implies moderately unfavorable; “Hold” to 0 which implies neutral; “Outperform” to 1 which implies moderately favorable; and “Buy” to 2 which implies strongly favorable.

11. BIBLIOGRAPHY

REFERENCES

- [1] Benjamin Graham, Jason Zweig, and Warren E. Buffett, *The Intelligent Investor*, Publisher: Harper Collins Publishers Inc, 2003.
- [2] Charles D. Kirkpatrick II and Julie R. Dahlquist, *Technical Analysis: The Complete Resource for Financial Market Technicians* (3rd Edition), Pearson Education, Inc., 2015.
- [3] Bruce Vanstone and Clarence Tan, A Survey of the Application of Soft Computing to Investment and Financial Trading, *Proceedings of the Australian and New Zealand Intelligent Information Systems Conference*, Vol. 1, Issue 1, http://epublications.bond.edu.au/infotech_pubs/ 13/, 2003, pp. 211–216.
- [4] Monica Tirea and Viorel Negru, *Intelligent Stock Market Analysis System - A Fundamental and Macro-economical Analysis Approach*, IEEE, 2014.
- [5] Kian-Ping Lim, Chee-Wooi Hooy, Kwok-Boon Chang, and Robert Brooks, Foreign investors and stock price efficiency: Thresholds, underlying channels and investor heterogeneity, *The North American Journal of Economics and Finance*. Vol. 36, <http://linkinghub.elsevier.com/retrieve/pii/S1062940815001230>, 2016, pp. 1–28.
- [6] Lamartine Almeida Teixeira and Adriano Lorena Inácio de Oliveira, A method for automatic stock trading combining technical analysis and nearest neighbor classification, *Expert Systems with Applications*, <http://linkinghub.elsevier.com/retrieve/pii/S0957417410002149>, 2010, pp. 6885–6890.
- [7] Banshidhar Majhi, Hasan Shalabi, and Mowafak Fathi, FLANN Based Forecasting of S&P 500 Index. *Information Technology Journal*, Vol. 4, Issue 3, <http://www.scialert.net/abstract/?doi=itj.2005.289.292>, 2005, pp. 289–292.
- [8] Ritanjali Majhi, G. Panda, and G. Sahoo, Development and performance evaluation of FLANN based model for forecasting of stock markets, *Expert Systems with Applications*, Vol. 36, Issue 3, <http://linkinghub.elsevier.com/retrieve/pii/S0957417408005526>, 2009, pp. 6800–6808.

- [9] Tong-Seng Quah and Bobby Srinivasan, Improving returns on stock investment through neural network selection, *Expert Systems with Applications*, Vol. 17, Issue 4, <http://linkinghub.elsevier.com/retrieve/pii/S095741749900041X>, 1999, pp. 295–301.
- [10] Halbert White, Economic prediction using neural networks: the case of IBM daily stock returns, *IEEE International Conference on Neural Networks*, <http://ieeexplore.ieee.org/document/23959/>, IEEE, 1988, pp. 451–458.
- [11] Yauheniya Shynkevicha, T.M. McGinnity, Sonya A. Coleman, and Ammar Belatreche, Forecasting movements of health-care stock prices based on different categories of news articles using multiple kernel learning, *Decision Support Systems*, Vol. 85, <http://linkinghub.elsevier.com/retrieve/pii/S0167923616300252>, 2016, pp. 74–83.
- [12] Han Lock Siew and Md Jan Nordin, Regression Techniques for the Prediction of Stock Price Trend, 2012 International Conference on Statistics in Science, Business and Engineering (ICSSBE), IEEE, 2012.
- [13] Chi Ma, Junnan Liu, Hongyan Sun, and Haibin Jin, A hybrid financial time series model based on neural networks, IEEE, 2017.
- [14] Lean Yu, Shouyang Wang, and Kin Keung Lai, Mining Stock Market Tendency Using GABased Support Vector Machines, *Internet and Network Economics*, http://link.springer.com/10.1007/11600930_33, 2005, pp. 336–345.
- [15] Fu-Yuan Huang, Integration of an Improved Particle Swarm Algorithm and Fuzzy Neural Network for Shanghai Stock Market Prediction, 2008 Workshop on Power Electronics and Intelligent Transportation System.[Online].August 2008, IEEE, <http://ieeexplore.ieee.org/document/4634852/>, 2008, pp. 242–247.
- [16] Ching-hsue Cheng, Tai-liang Chen, and Hiajong Teoh, Multiple-Period Modified Fuzzy Time-Series for Forecasting TAIEX, Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007), <http://ieeexplore.ieee.org/document/4406190/>, IEEE, 2007, pp. 2–6.
- [17] Pei-Chann Chang, Chin-Yuan Fan, and ShihHsin Chen, Financial Time Series Data Forecasting by Wavelet and TSK Fuzzy Rule Based System, Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007), <http://ieeexplore.ieee.org/document/4406255/>, IEEE, 2007, pp. 331–335.

- [18] Lixin Yu and Yan-Qing Zhang, Evolutionary Fuzzy Neural Networks for Hybrid Financial Prediction, IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews), Vol. 35, Issue 2, <http://ieeexplore.ieee.org/document/1424198/>, IEEE, 2005, pp. 244–249.
- [19] Chokri Slim, Neuro-fuzzy network based on extended Kalman filtering for financial time series, World Academy of Science, Engineering and Technology, Vol. 15, <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.193.4464&rep=rep1&type=pdf>, 2006, pp. 134–139.
- [20] Gérard Biau and Luc Devroye, Lectures on the Nearest Neighbor Method, Springer, 2015.
- [21] Saed Sayad, K Nearest Neighbors - Classification, [Online]. 2017, [http://www.saedsayad.com/k_nearest_neighbors .htm](http://www.saedsayad.com/k_nearest_neighbors.htm), 2017, [Accessed: 10 January 2017].
- [22] Dan Morris, Bayes' Theorem: A Visual Introduction for Beginners, Blue Windmill Media, 2017.
- [23] James V Stone, Bayes' Rule With R: A Tutorial Introduction to Bayesian Analysis, Sebtel Press, 2016.
- [24] Marco Scutari and Jean-Baptiste Denis, Bayesian Networks: With Examples in R, CRC Press, 2014.
- [25] Peter Bruce and Andrew Bruce, Practical Statistics for Data Scientists: 50 Essential Concepts, O'Reilly Media, 2017.
- [26] Ciaran Walsh, Key Management Ratios, 4th Edition (Financial Times Series), Prentice Hall, 2009.
- [27] Seyed Enayatollah Alavi, Hasanali Sinaei, and Elham Afsharirad, Predict the Trend of Stock Prices Using Machine Learning Techniques, IAIEST, 2015.
- [28] Lock Siew Han and Md Jan Nordin, Integrated Multiple Linear Regression-One Rule Classification Model for the Prediction of Stock Price Trend, Journal of Computer Sciences, Vol 13 (9), 2017, pp. 422-429.

- [29] Abdolhossein Zameni and Othman Yong, Share Price Performance of Malaysian IPOs Around Lock-Up Expirations, *Advanced Science Letters*, Vol 23(9), 2017, pp. 8094-8102.
- [30] Abdolhossein Zameni and Othman Yong, LockUp Expiry And Trading Volume Behaviour Of Malaysian Initial Public Offering's, *International Journal of Economics and Financial Issues*, Vol 6(3), 2016, pp. 12-21.
- [31] Hani A.K. Ihlayyel, Nurfaadhina Mohd Sharef, Mohd Zakree Ahmed Nazri, and Azuraliza Abu Bakar, An Enhanced Feature Representation Based On Linear Regression Model For Stock Market Prediction, *Intelligent Data Analysis*, Vol 22(1), 2018, pp. 45-76.
- [32] Lida Nikmanesh and Abu Hassan Shaari Mohd Nor, Macroeconomic Determinants of Stock Market Volatility: An Empirical Study of Malaysia and Indonesia, *Asian Academy of Management Journal*, Vol 21(1), 2016, pp. 161- 180.
- [33] Luigi Troiano, Pravesh Kriplani, and Irene Díaz, Regression Driven F-Transform and Application to Smoothing of Financial Time Series, *IEEE*, 2017.