IPO PRECISION USING ML

Project report submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Electronics and Communication Engineering

by

Vipul Khandekar - 20uec147
Akash Tiwari - 20ucc012
Ankit Gulia - 20ucc015
Aadesh Malani - 20ucc001

Under Guidance of
Dr. Ashish Kumar Dwivedi

Department of Electronics and Communication Engineering
The LNM Institute of Information Technology, Jaipur

November 2023

The LNM Institute of Information Technology
Jaipur, India

CERTIFICATE

This is to certify that the project entitled "IPO price prediction" , submitted by Vipul Khandekar (20uec147), Akash Tiwari (20ucc12) ,Ankit Gulia (20ucc015) , Aadesh Malani(20ucc001) in partial fulfillment of the requirement of degree in Bachelor of Technology (B. Tech), is a bonafide record of work carried out by them at the Department of Electronics and Communication Engineering, The LNM Institute of Information Technology, Jaipur, (Rajasthan) India, during the academic session 2023-2024 under my supervision and guidance and the same has not been submitted elsewhere for award of any other degree. In my/our opinion, this report is of standard required for the award of the degree of Bachelor of Technology (B. Tech).

Date

Adviser: Dr. Ashish Kumar Dwivedi

Acknowledgments

We would like express our special thanks to Dr. Ashish Kumar Dwivedi for his support and guidance throughout the semester. Your suggestion and advice was very essential for us for the project's progress.
We would also like to thank the academic setup for providing us this golden opportunity to learn in different field and use that knowledge to build our project that can useful in real life and can have an impact over people's life.

Abstract

Listing of share on the stock exchanges is called as IPO (Initial Public Offering). Whenever a company want to expand it's business they have raise funds and for that they have two ways:

1. By taking loan from other people in the form of Bonds in which they promise to pay the bond holder pre determined interest for a certain period of time.

2. The other method or we can say more popular method is to dilute the equity of existing share holder and they list the share on the stock exchanges that means opening door to the people for investment opportunity in the company.

Before letting the retail investor to participate in the investment process, the company goes to a investor banker who analyze all the asset and liability of the company and looks at the financial statement to reach to a final valuation of the share then that share is sold to retail investor.

There have been cases where the price band of the IPO has been inflated so as to give an edge to the pre existing investor, currently there is no model in the market which can provide a accurate open price of an IPO.

The market analyst try to find the listing gains that a retail investor can have by looking at the factors such as:
1.Previous financial statement
2.Grey market price
3.The qualification of management who has been handling the company
4.The penetration of the company in the market
5.Size and growth in the market in which the company is operating

But in the previous times we have seen that these all research have been proved wrong and because of that many retail investors have faced a lot of loss, we can look at the popular example

of PAYTM, the company had great penetration in the market, it was India's largest IPO, the grey market price was also trading at higher price than that offered in the IPO, almost every investor was saying to invest in the company but when the IPO final made a debut on the exchange it opened at a loss of almost 10 percent on the same it closed at a loss of 25 percent and it currently trading at half of it's value, giving a loss of almost 50 percent that is hard to recover.

In 2020 the stock market boomed so did the IPO market therefore every company tried to list it's IPO and so it became difficult and confusing for the retail investor where to invest and so we have come up with an of building a model which analyzes various data related to an IPO such as NII investment value,QIB investment value, issue size , issue price band and tries to predict the opening price of the IPO which will make it easy for retail investor to invest in the right IPO.

Contents

Chapter 1

Introduction

1.1 The Area of Work

IPO price prediction is specialised area of work where we need to analyze different aspects of the company such as the financial performance over the year, the projected growth in the industry and how well it is equipped to cope up with the upcoming challenges.
By analyzing these aspects we can come up with the model that can predict the listing gains considering the above mentioned factors as well other important factors which will help the retail investor to make a proper decision whether to invest in the particular IPO or not, this model will help the retail investor to gain insights about the IPO and so in this way the we can protect the most vulnerable part of the stock market that is the retail investor from making wrong investment decision and falling prey to operators in the market.

1.2 Problem Addressed

At the present time there is no system existing which can provide the information about how much we have the possibility to gain from the IPO of a company. There are many renowned market analyst who provide their advise on the IPO but their advise is based on the previous history of the company, board of directors only therefore at times they have been proven wrong by the market, also sometimes people try to apply for an IPO based on the grey market price but that also has given disappointing result in the past.
Because of this the retail investors loose out a lot of money as for applying in the IPO we have to buy a lot of shares almost of the value Rs 14000 in India.So we have tried to make a model using data such as how much money is being put into IPO but big investors and banks and using the algorithm based on the data we have tried to predict the listing gains which can provide assistance to the retail investors. By this way we can predict the listing gains of the IPO and based on that information the retail investor can decide whether they want to invest in the IPO or not.

1.3 Existing System

At this present point of time there is no such concrete existing system which can be accurate enough to predict the price of IPO. Retail investor mostly try to take advise from the financial analyst or CA who looks into the qualitative aspects of the company such as the board of directors their qualification on how well they can handle the difficult situation also they have a look at the existing investor and the potential investor that is told by the company so these information can be manipulated.
Many a times they also try to compare other companies in the same domain as the company who is making debut and then try to make inference regarding the prices but that can go way wrong as we know that every company even in the same domain is operated very differently.

The model that we are trying to build is unique of its own kind and we are looking at the quantitative aspects as they very well explain the quality of the company.

Chapter 2

Literature Review

2.1 Introduction

In this chapter we will be mainly concerned about all the worked we put in to achieve the desired goal So all we want is a perfect ML model to predict the IPO price with maximising accuracy of the model We started by collecting data from various sites available on internet using web scrapping (Web scraping is an automatic method to obtain large amounts of data from websites) technique we used Apify as our web scrapper and we used this web scrapper on sites like moneycontrol and chittorgarh Now the main purpose of the project was to learn, make and apply Machine Learning Models to data set after properly sorting and completing the data We read many research papers involving Machine Learning Model and how to apply them on various kinds of data set to maximise or accuracy

2.2 About the Data Set

One of the most important parts of any machine learning project is the Data Collection and Preparation. The act of obtaining and analysing data on certain variables in an established system is known as data collection or data gathering. This procedure allows one to analyse results and provide answers to pertinent queries

Data Collection: After finding various websites related to IPO price information and other data points related to it, it has to be extracted in a structured manner for it to be useful. We used web scraping (web scraping is a technique used to gather information from websites. It is also known as online harvesting or web data extraction. The World Wide Web may be accessed directly by web scraping software utilising a web browser or the Hypertext Transfer Protocol) to gather the information and then cleaned out undesirable information that we did not need at this point

Exploratory Data Analysis: Data scientists utilise exploratory data analysis (EDA), which frequently makes use of data visualisation techniques, to examine and analyse data sets and summarise their key properties. Using scatter plots and histograms to identify the useful data points and relations that affect the Open Price of IPO and remove all the information that is not needed to filter out unnecessary information.// // Feature Engineering and Selection: Exploratory Data Analysis inevitably leads to feature engineering and feature choice. The process of ”feature engineering” entails taking unprocessed data from the chosen data sets and turning it into ”features” that more accurately reflect the core issue that needs to be resolved. From here we found out the factors that directly affect the open price of an IPO and build this in a structured manner.

Training and Evaluation: After applying Exploratory Data Analysis and Feature Engineering and Selection, a data set was created upon which the algorithms will be tested and models will be built upon.The data set consisted of 494 companies. The recent company IPO was listed on 4/13/2022 at the time of writing and the earliest IPO which we gathered was dated on 20th of September 2006. After some iterative runs we arrived at some issues that needed to be dealt before moving forward. First of them was to tackle the inflation effect that was present in the data and was hindering the algorithm into giving hiked results. As the data ranges over more than 16 years the value of Rupee has inflated with time. If we used the data we have we would have build our model on prices whose value is not correct if we measure it in the present year, so we needed to inflate the price of every data point that had money involved which was present in most of the attributes. So we added some new attributes to tackle the situation

Extra attributes added are
1. Inflation Rate: The rate at which prices increase over time, resulting in a fall in the purchasing value of money. 2. Present Value of Issue Size(Cr): The Issue Size which is represented in Crores is adjusted as per inflation. 3. Present Value of Issue Price: The Issue Price is adjusted as per inflation. 4. Present Value of Open Price: The Open Price is adjusted as per inflation. 5. Present Value of Closing Price: The Closing Price is adjusted as per inflation.

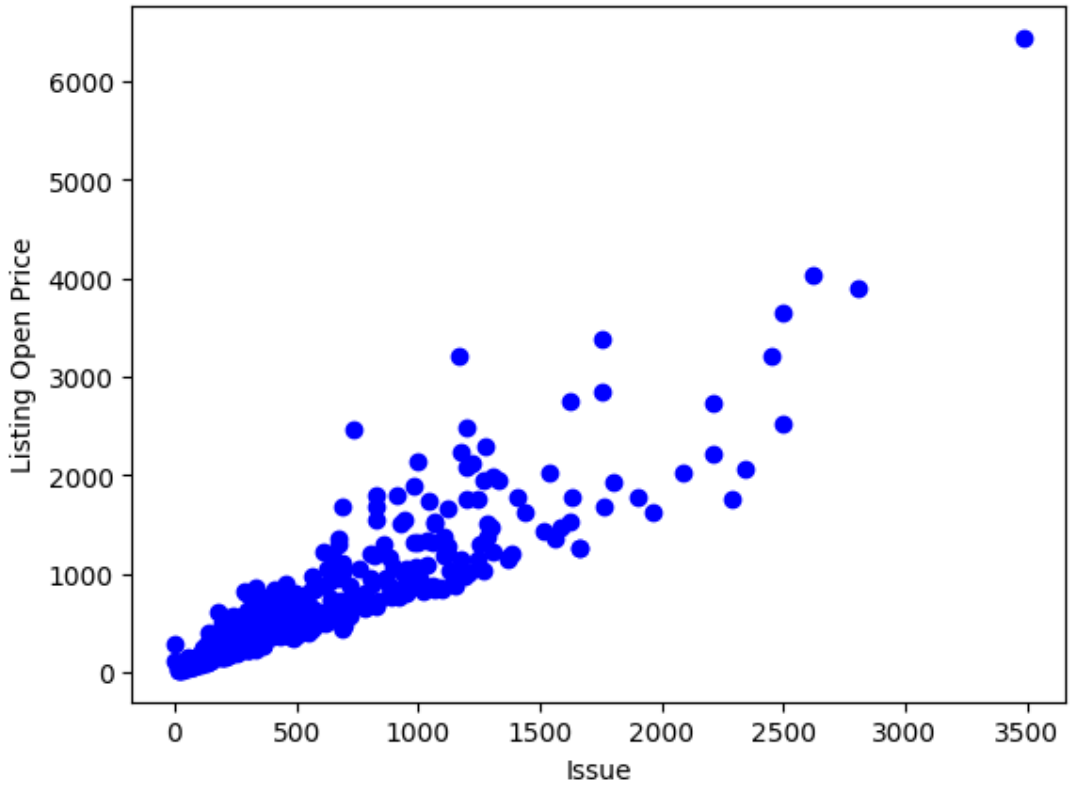

FIGURE 2.1: Exploratory data analysis for listing opening price VS issue price

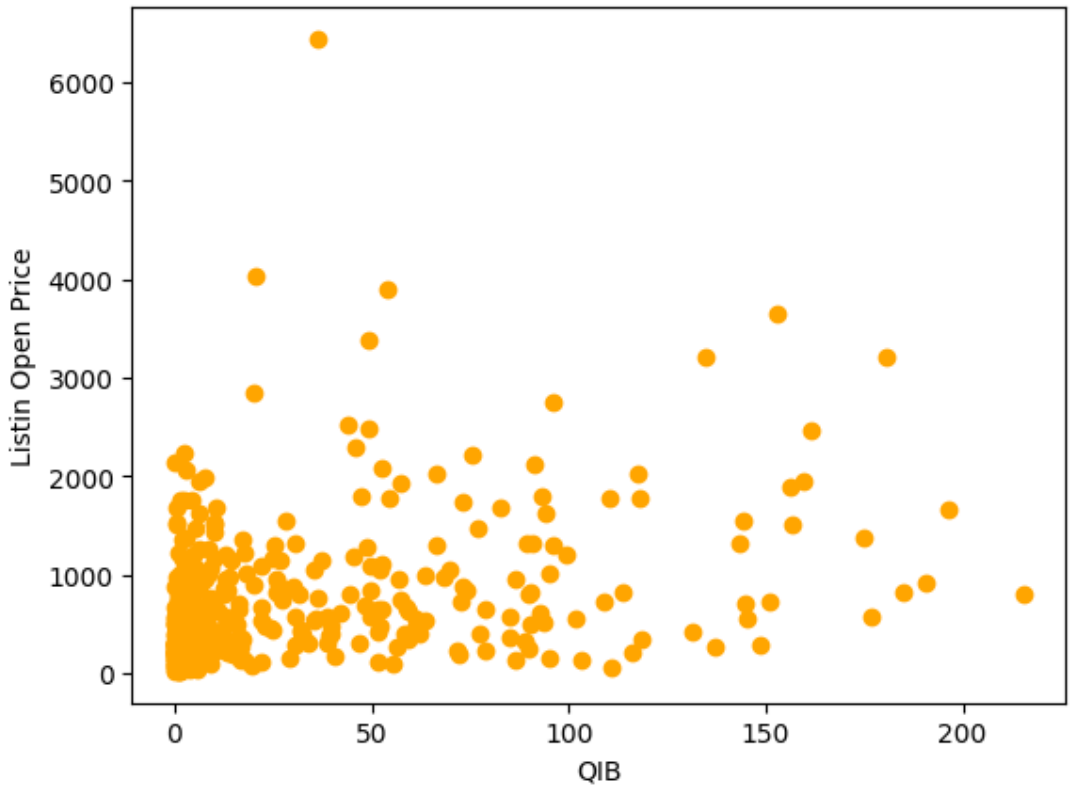

FIGURE 2.2: Exploratory data analysis for listing opening price VS Qualified institutional buyer

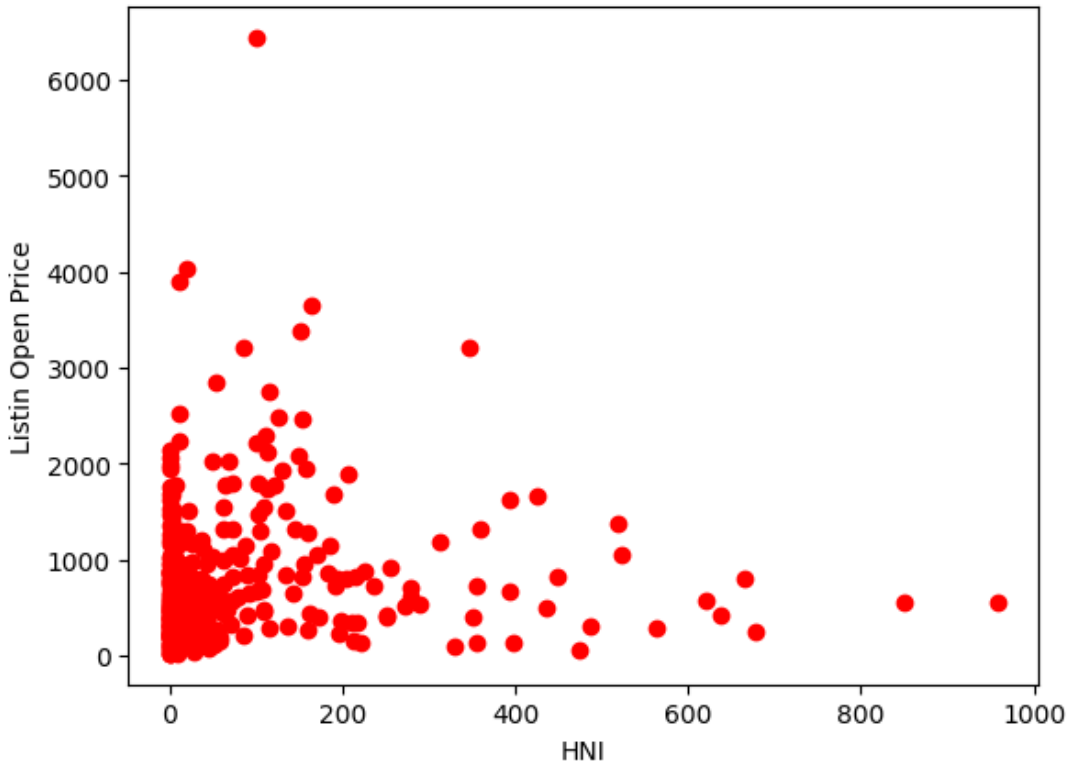

FIGURE 2.3: Exploratory data analysis for listing opening price VS High net worth indiviual participation

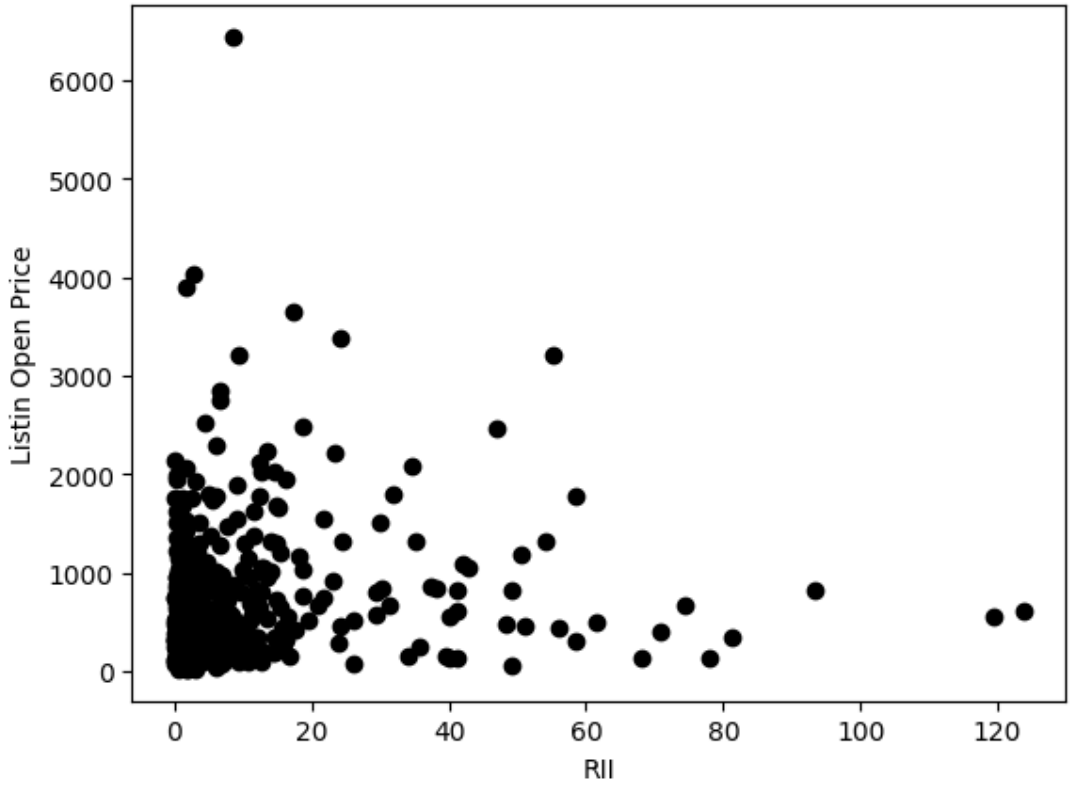

FIGURE 2.4: Exploratory data analysis for listing opening price VS Retail participation

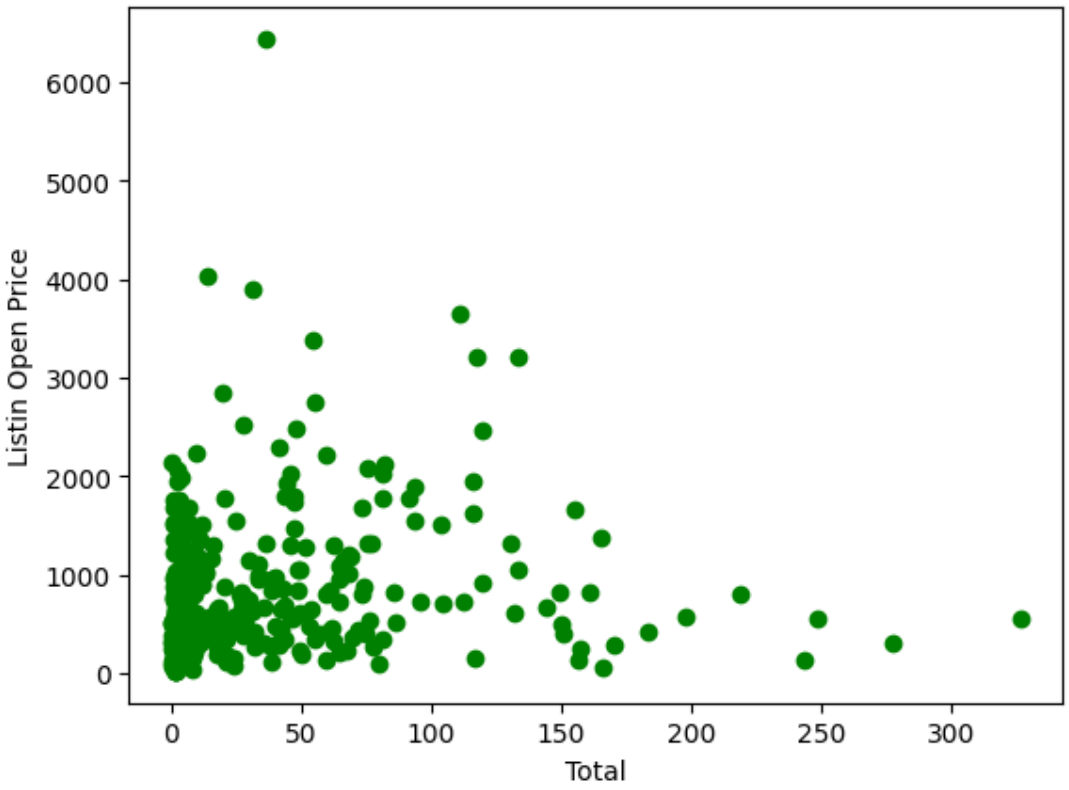

FIGURE 2.5: Exploratory data analysis for listing opening price VS total participation

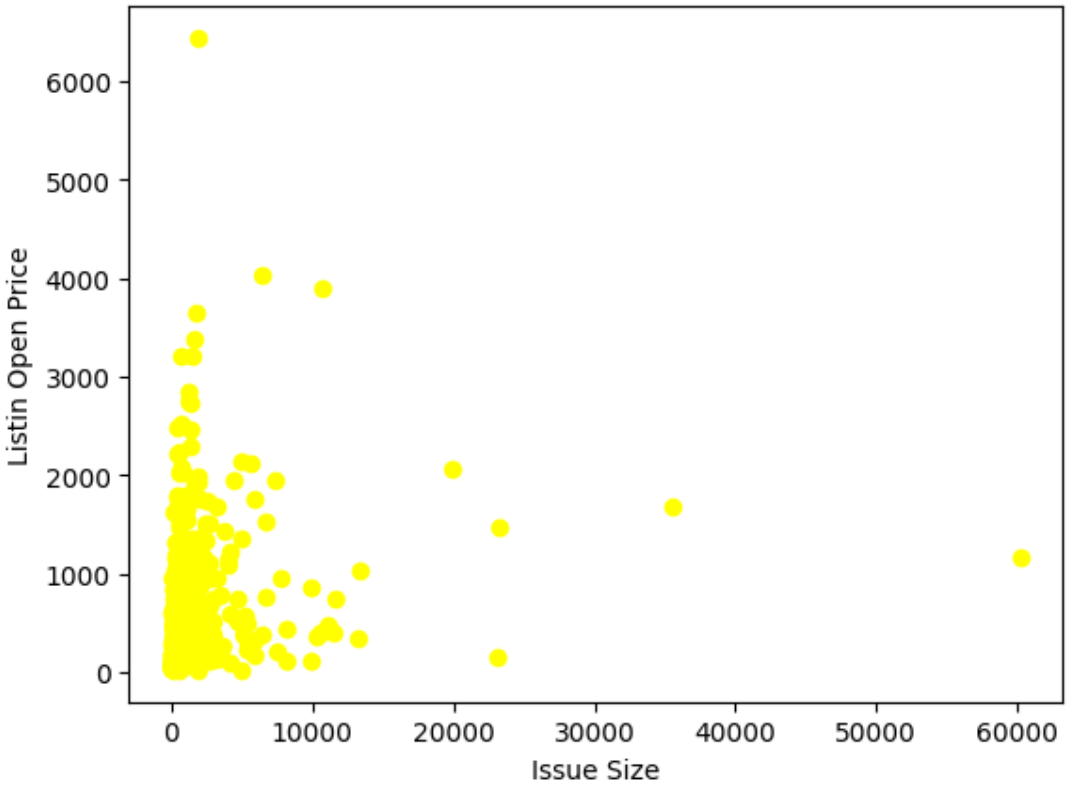

FIGURE 2.6: Exploratory data analysis for listing opening price VS issue size

Chapter 3

Algorithms Used

3.1 Introduction

The algorithms used till now for model training and for predicting the price of a particular IPO are

- Random Imputation(for dealing with the null points)
- Decision Tree
- Random Forrest
- Cross Folding

3.2 Algorithms

- Random Imputation : Random Sample Imputation take a random observation from the feature. After that we use random observation to replace NaN in that feature. It should be used when data is missing completely at random (MCAR)

- Decision Tree : A decision tree is a non-parametric supervised learning algorithm, which is utilized for both classification and regression tasks. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes.
  A decision tree is a tree-like model where each node represents a feature or attribute, and each branch represents a possible value for that feature. The leaves of the tree represent the final decision or classification. The decision tree algorithm builds the tree by recursively splitting the data based on the most informative feature until a stopping criterion is met.
  When building a decision tree, the algorithm recursively splits the data into subsets

based on the most informative feature, until a stopping criterion is met. This means that the algorithm starts by selecting the feature that provides the most information gain, which is the feature that splits the data into subsets that have the highest purity or homogeneity in terms of the target variable.

The splitting process is repeated on each subset, creating a new branch for each possible value of the selected feature, until the stopping criterion is met. The stopping criterion can be based on different conditions, such as a maximum depth of the tree, a minimum number of samples required to split a node, or a minimum improvement in information gain.

The goal of the decision tree algorithm is to create a tree that accurately predicts the target variable based on the values of the input features. By recursively splitting the data based on the most informative feature, the algorithm can create a tree that captures the relationships between the input features and the target variable. However, the decision tree algorithm is prone to overfitting, which is why other algorithms such as random forests are often used to improve accuracy and reduce overfitting.

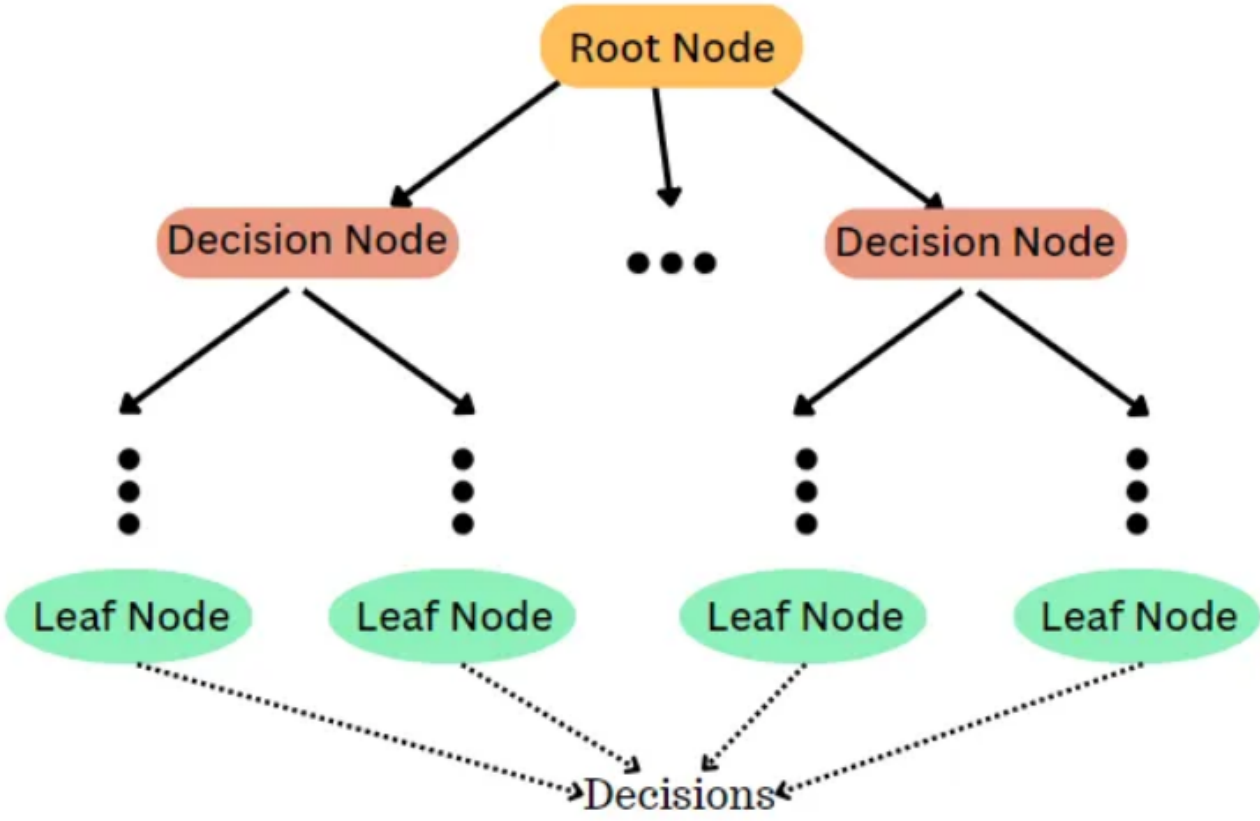

FIGURE 3.1: Basic structure of decision tree

Let's take a example so that we can learn about the basic working of the decision tree, Let's say that we have to decide whether or not the atmosphere outside good enough for us to go out then we would be looking at 2 features
1.Temperature
2.Humidity

Depending on these two feature we would be deciding whether or not we would go outside so if the temperature is more then 22 then we would go out but if the temperature is less than 22 then would be taking humidity into consideration and so we would then say that if humidity is less than 67 it is good for us to move out so the decision tree would look like as follows:

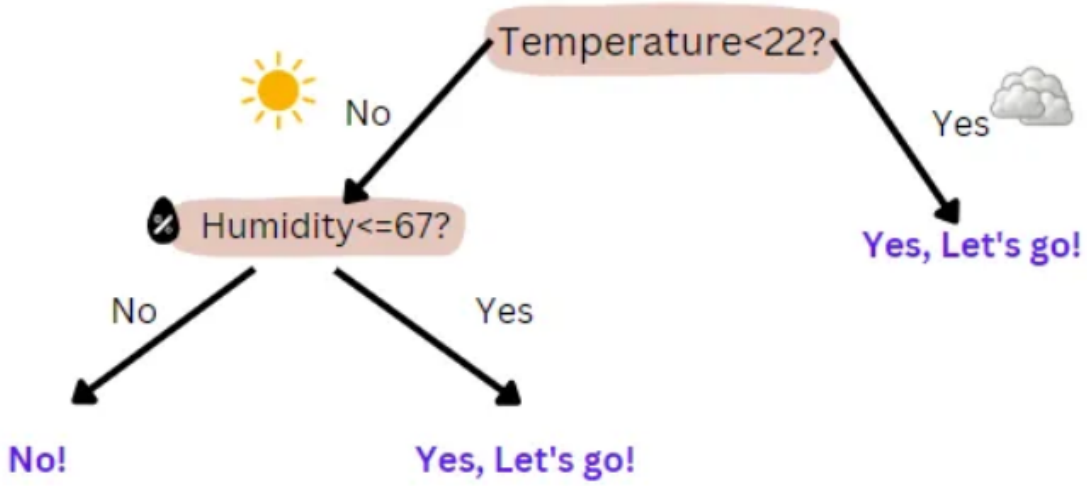

FIGURE 3.2: Basic structure of decision tree

- Random Forrest : Random forest, on the other hand, is an ensemble method that builds multiple decision trees and combines their predictions to improve accuracy and reduce overfitting. Each decision tree in the random forest is trained on a different subset of the data, and a random subset of features is considered for each split in the tree.

  The random forest algorithm uses the concept of bagging (bootstrap aggregating) to build the individual decision trees, where random subsets of the training data are used to train each tree. Additionally, random subsets of the features are considered for each split in the tree. By combining multiple decision trees, the random forest algorithm can reduce the variance of the model and increase the accuracy of predictions.

Overall, decision trees are simple and easy to understand but prone to overfitting. Random forests, on the other hand, are more complex and have better accuracy, especially for large datasets.

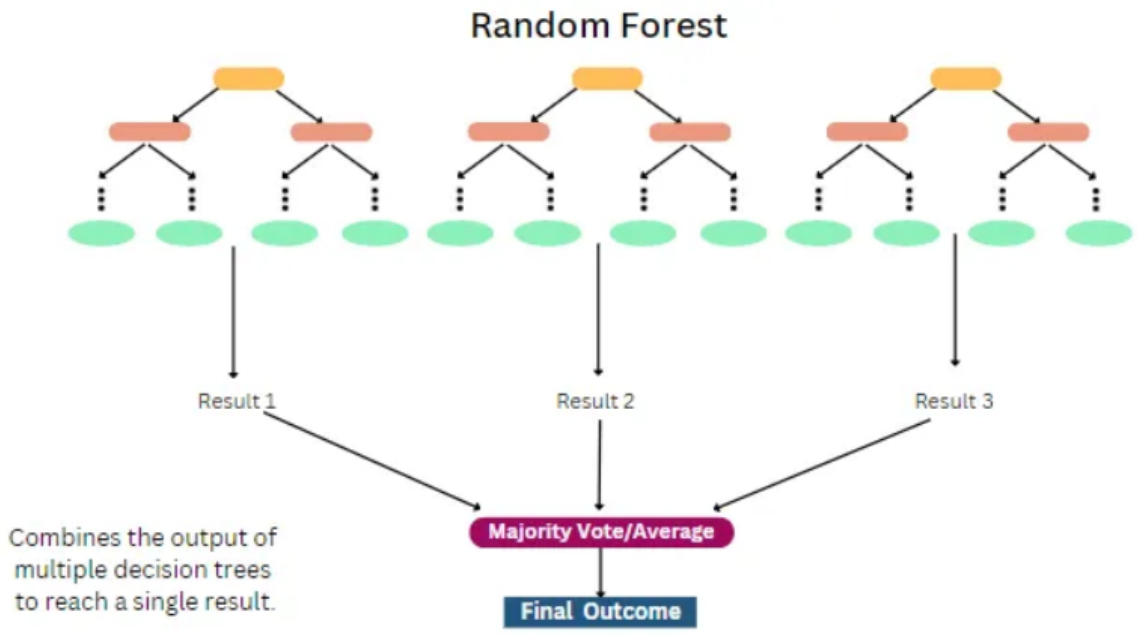

FIGURE 3.3: Basic random forest structure

3.2.1 Selecting best feature for each node

We can manually select the best feature amongst the one we have chosen by doing exploratory data analysis but the most used criteria are Gini impurity and information gain methods which automatically help us to decide the splitting criteria at each node.

3.2.1.1 Entropy

The entropy basic formula used is :

S represents the dataset on which entropy is computed
In p(c), the letter ”c” denotes ”the classes in set, S
p(c) signifies the probability of the number of points belonging to class c relative to the total number of data points in the dataset

The entropy is 0 when all data points in the set belong to the same class (perfectly pure set), and it is maximal when the distribution of classes is uniform (maximum impurity).

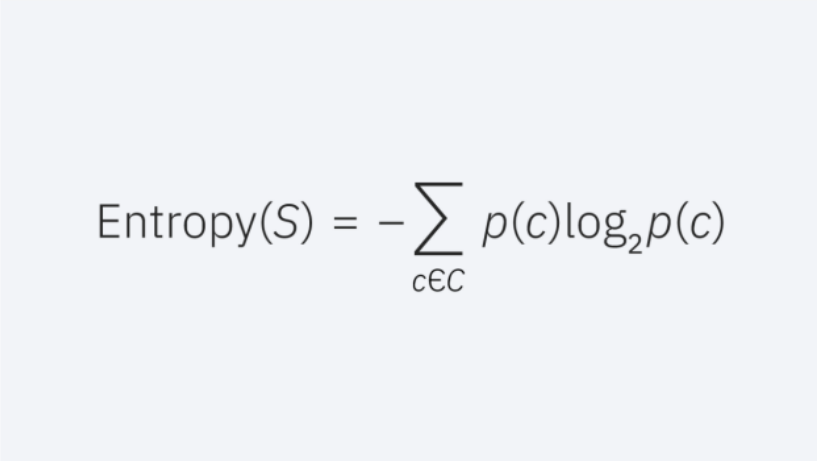

FIGURE 3.4

3.2.1.2 Information Gain

Entropy's value varies depending on the entity. Before a split, the value is different, and after the split, the value changes. Information Gain is the term used to describe the value shift. The optimal split is eventually found by the entity that groups the training data the best while continuing to successfully locate the correct answer.The following relationship can be used to indicate information gain, with the symbols denoting the following:

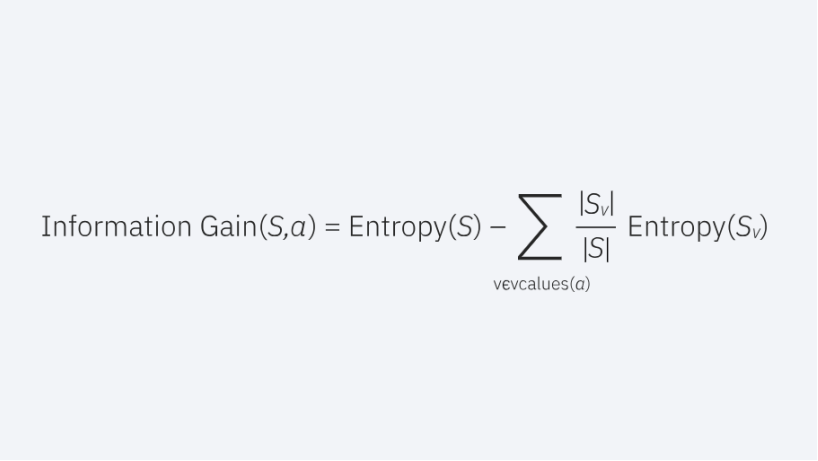

FIGURE 3.5

- a "identifies a particular attribute or class label."
"The entropy of dataset, S" is represented by "Entropy(S)".
- "Sv / S" stands for "probability of the number of Sv-related data points to the total number of S-related data points in the dataset."
The expression "Entropy(Sv)" denotes "the entropy of dataset, Sv."

3.2.1.3 Gini impurity

Gini impurity is a measure of the disorder or impurity in a set of data points within a decision tree context. It quantifies the likelihood of incorrectly classifying a randomly chosen element's label within the set. A Gini impurity of 0 indicates a perfectly pure set, where all elements belong to the same class, while a Gini impurity of 1 implies maximum impurity, suggesting an equal distribution of elements across different classes. In decision tree algorithms, the Gini impurity is utilized as a criterion for selecting the best split at each node, aiming to create subsets that are as pure as possible with respect to the target variable.

$$\text{Gini Impurity} = 1 - \sum_i (p_i)^2$$

FIGURE 3.6

- Ensemble Methods : The essence of ensemble lies in gathering multiple results into one to identify the most favorable outcome. Decision trees and other classifiers, which are used in machine learning, group their predictions into one and select the best among them. ”Boosting and bagging, commonly referred to as bootstrap aggregation,” are currently the most widely utilized ensemble methods. ”The bagging technique” was developed by Leo Breiman in 1996. This is done by ”picking a random sample of data from a training set with replacement,” which allows for the selection of individual data points several times. These models are independently trained after generating multiple data samples. Depending on the job (e.g., regression or classification), the average or majority of those predictions result in a more accurate estimate”

”An uncorrelated forest of decision trees by combining feature randomization with bagging, extends the bagging method” is the result of the method Random Forest uses. Feature randomization, also known as ”the random subspace technique” or feature bagging, is the process of assembling a collection of randomly generated features. In addition, it aids in preserving as many low relations as feasible inside the various

decision trees.The main distinction between random forests and decision trees is this. Random forests choose only a portion of those feature divides, while decision trees take into account all possible quality splits. Every decision tree in the ensemble is built using the bootstrap sample, a data sample taken from a training set using replacement known as random forest technique.One-third of the training sample, or the out-of-bag (oob) sample, is set aside for test data. Next, feature bagging is employed to present adding a second randomization to the dataset increases its variety and reduces decision-making tree association. When faced with various issues, the actions we must take to ensure the Expectations will differ. For instance, ”the majority vote, the classification task, or the projected class will be determined using the most common categorical variable. For the regression task, the individual decision trees will be averaged.

3.2.2 How it Works

To commence the training process, it is imperative to furnish the hyper parameters. A minimum of three hyper parameters must be explicitly specified for the progression to unfold. Examples of such hyper parameters encompass node size and tree count. Following this, we can seamlessly transition to executing classification or regression tasks employing the designated algorithm.

Subsequently, we adhere to a systematic procedure elucidating the algorithm's functionality succinctly. In essence, each decision tree within the ensemble constituting the random forest method is constructed from a data sample extracted with replacement from a training set, commonly referred to as the bootstrap sample. A fraction, specifically one-third, of the training sample is earmarked as the out-of-bag (oob) sample, serving as the test data. A second randomization infusion is introduced through feature bagging, thereby augmenting dataset diversity and diminishing the correlation among decision trees.

The predictive process varies depending on the nature of the problem, as expounded earlier. Following this, the ”out-of-bag sample” is harnessed for predictions within the framework of the ”cross-validation method.”

Impurity	Task	Formula	Description
Gini impurity	Classification	$\sum_{i=1}^{C} f_i(1-f_i)$	f_i is the frequency of label i at a node and C is the number of unique labels.
Entropy	Classification	$\sum_{i=1}^{C} -f_i \log(f_i)$	f_i is the frequency of label i at a node and C is the number of unique labels.
Variance / Mean Square Error (MSE)	Regression	$\frac{1}{N}\sum_{i=1}^{N}(y_i-\mu)^2$	y_i is label for an instance, N is the number of instances and μ is the mean given by $\frac{1}{N}\sum_{i=1}^{N} y_i$
Variance / Mean Absolute Error (MAE) (Scikit-learn only)	Regression	$\frac{1}{N}\sum_{i=1}^{N}\lvert y_i-\mu\rvert$	y_i is label for an instance, N is the number of instances and μ is the mean given by $\frac{1}{N}\sum_{i=1}^{N} y_i$

FIGURE 3.7

3.2.3 Error Matrix Used

In this section, we delve into the prevalent metrics associated with regression models and explore the error metrics employed within the confines of this project:

1. Mean Absolute Error (MAE):

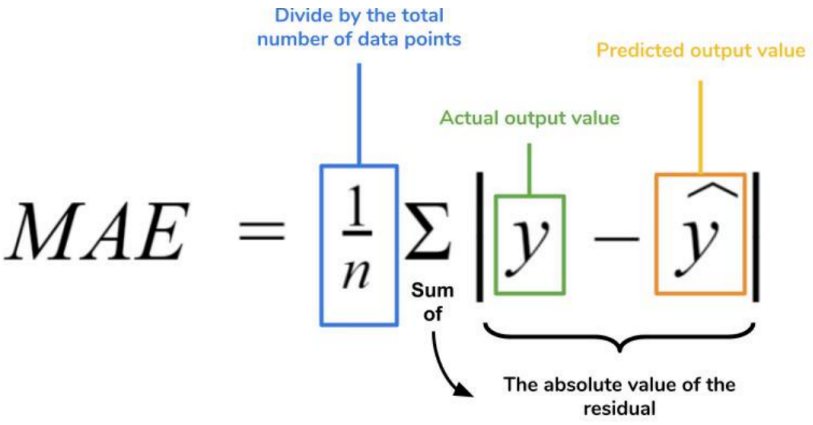

FIGURE 3.8

In the realm of statistics, the mean absolute error (MAE) serves as a gauge for discrepancies between paired observations depicting identical phenomena. Computed by dividing the cumulative absolute errors by the sample size, MAE aligns its scale with that of the measured data. It's worth noting that alternative formulations may incorporate relative frequencies as weight factors. Being a "scale-dependent accuracy metric," it becomes impractical to juxtapose predicted values derived from diverse scales.

2. Mean Squared Error (MSE):

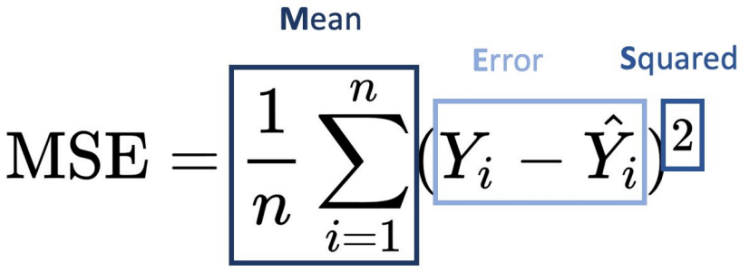

FIGURE 3.9

The mean squared error (MSE) or mean squared deviation (MSD) of an estimator gauges the average of the squared errors, depicting the average squared disparity between estimated values and the actual value. MSE, functioning as a risk function, corresponds to the anticipated value of the squared error loss. The elusive nature of MSE being strictly positive (and never negative) is attributed to randomness or the estimator's oversight of data that could enhance the precision of the estimate. In the realm of machine learning, MSE might allude to empirical risk but, more accurately, aligns with "empirical risk minimization" — the mean loss observed on a dataset.

3. Root Mean Squared Error(RMSE):

$$RMS = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_i - p_i)^2}$$

FIGURE 3.10

The assessment of deviations between values predicted by a model or estimator and the observed values, whether from a sample or a population, is commonly done through the root-mean-square deviation (RMSD) or root-mean-square error (RMSE). RMSD, being the quadratic mean of the square root of the second sample moment of disparities between expected and observed values, proves instrumental in evaluating forecasting accuracy. When calculations extend beyond the data sample used for estimation, the discrepancies are termed errors or prediction errors instead of residuals. RMSD efficiently consolidates the magnitudes of forecasting errors across diverse data points into a singular measure of predictive efficacy. Serving as a metric for accuracy, RMSD facilitates the assessment of prediction errors across multiple models. A theoretical perfection is represented by a numerical value of 0 (almost unattainable in practice), and RMSD consistently maintains a non-negative disposition. The preference typically leans towards a smaller RMSD over a larger one; however, caution must be exercised, as the measure's sensitivity to the scale of numbers used renders cross-data comparisons inaccurate. The RMSD, defined as the average of the square roots of squared errors, reflects each error's impact proportionate to the squared error's magnitude, making it susceptible to the influence of outliers.

- K-Nearest Neighbour: For tackling classification and regression forecasting challenges, one may employ K-nearest neighbors (KNN), a method within the realm of supervised machine learning. Functioning as a supervised learning classifier, the K-nearest neighbor method anticipates and classifies the categorization of each data entity. While it is harnessed to address both "classification or regression problems," it predominantly finds its application as a classification algorithm.

  Much like its role in classification quandaries, the concept is also enlisted for tackling regression problems. However, in this scenario, the prediction of classification involves leveraging the average of the k nearest neighbors. The key divergence lies in the nature of the data – discrete data guides classification, whereas regression is tailored for handling "continuous" data points. Nevertheless, for either pursuit, the establishment of measurements precedes classification. The commonly utilized metric for measurement is the Euclidean distance, a subject we delve into in greater detail below. It's noteworthy that the K-nearest neighbors (KNN) technique, characterized by its omission of a

FIGURE 3.11

dedicated training phase in favor of preserving a training dataset, is a member of the cadre of lazy learning models. This nomenclature stems from its reliance on memory to retain all training data, creating a concurrent scenario where all calculations unfold simultaneously with classification or prediction. An alternate descriptor for this method is a memory-based or instance-based system, underscoring its dependence on memory for housing the entirety of its training data.

KNN distance metrics:

The core aim of the ”k-nearest neighbor algorithm” is to identify the nearest neighbors of a query point for subsequent classification. To operationalize this, the utilization of the following metrics is imperative:

Determine your distance metrics:

In gauging proximity, it is imperative to compute ”the distance between the query point and the other data points.” These distance measurements not only facilitate the determination of closeness but also streamline the creation of ”decision boundaries,” effectively categorizing query points into distinct categories. The depiction of decision boundaries often leans on the utilization of Voronoi diagrams.

1.EUCLIDEAN DISTANCE:

This is actually the most common and a simple method to calculate the distance between two points. The basic formula used is similar to the straight line distance between the points.

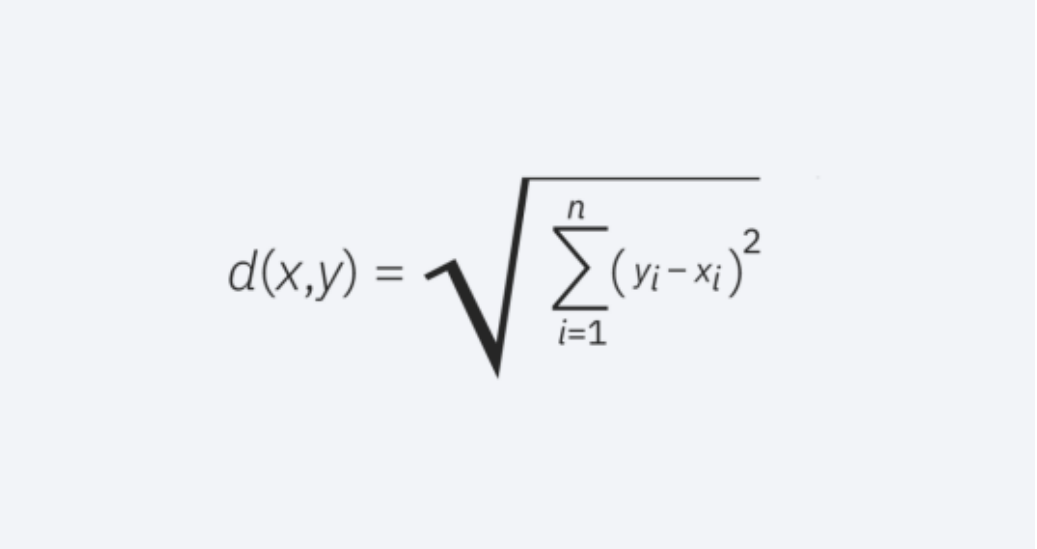

FIGURE 3.12

2.MANHATTAN DIATANCE

Measuring the distance between two points in absolute terms is called as Manhattan distance. We can also think of this formula as measuring the distance between two location on the map where people might have to travel and so it also called as city block distance.

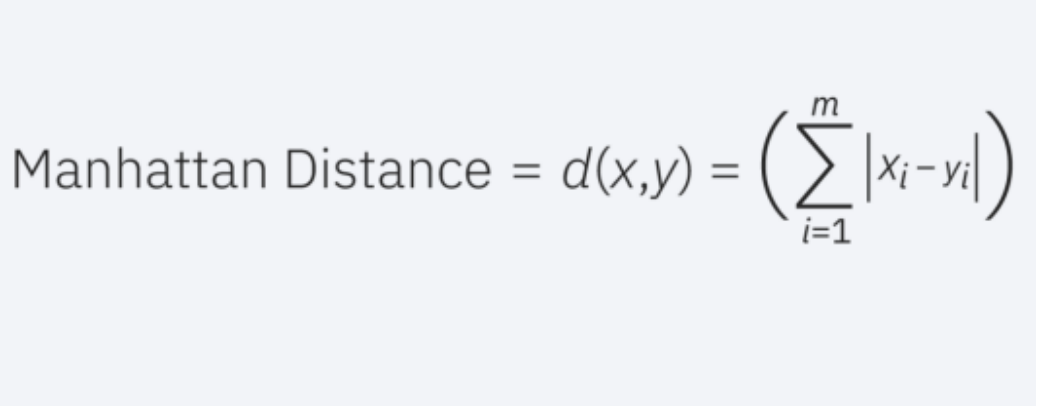

FIGURE 3.13

3.MINKOWSKI DISTANCE

This is a way to measure distance which is again similar to the Manhattan distance formula.

$$\text{Minkowski Distance} = \left(\sum_{i=1}^{n} \left|x_i - y_i\right|\right)^{1/p}$$

FIGURE 3.14

4.HAMMING DISTANCE

Basically this is a way in which we measure a distance strings and Boolean vectors when they are not overlapping.If we take a examplewe can say that if we are going to cmpare two words and only three letters are dofferent in the two then we can say that the hamming distance is 3 simply.

$$\text{Hamming Distance} = D_H = \left(\sum_{i=1}^{k} \left|x_i - y_i\right|\right)$$

$$x=y \qquad D=0$$
$$x\neq y \qquad D\neq 1$$

FIGURE 3.15

Advantages and disadvantages of the KNN algorithm

ADVANTAGES:

1.Easy to use: This algorithm is basic and can be understood easily because it is clear and accurate.

2. Limited parameters: Compared to other matheds used this only need a 'k' value and a way to measure the distance so we can say that not much parameters are required for analysis using this algorithm.

3.Adaptiveness: Whenever we add new information to this algosithm it adapts easily and adjusts to the new information fastly.

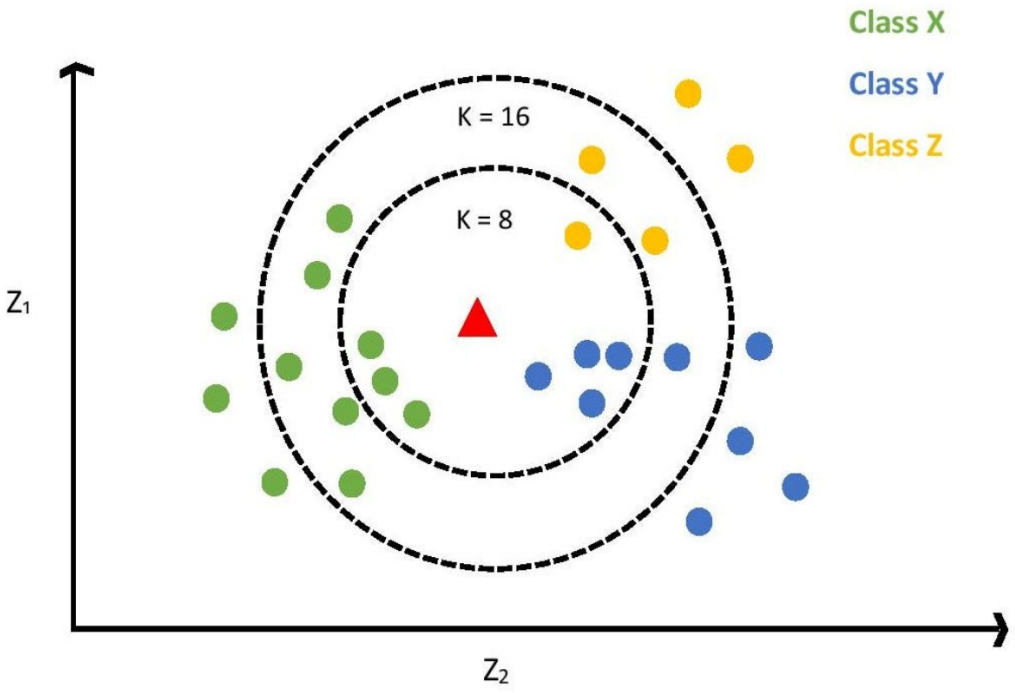

FIGURE 3.16

DISADVANTAGES

1.Memory utilization:The amount of memory used by KNN is more compared to other algorithms and which makes this slow, it would be useful to use other methods if we want to save time and money.

2.Issue of dimensionality: Whenever there are more number of factors on which our output depend this algorithm face a lot of problem, especially when the data given to train this algorithm is less.

3. Overfitting: Whenever the algorithm works on one type of data only then it is not able to generalize the result and provide us with bogus expected results and this same thing happens in this algorithm also where the data fit wery closely so the choice of selecting neighbours is difficult and sometimes the algorithm is not able to find the right balance between that.

- Cross Folding : Basically whenever we are working with a lot of data the most basic approach is that we divide the data in two part and the larger part is then used to train the data and the smaller data set is used to eventually test the model and find out how accurately the model is working.

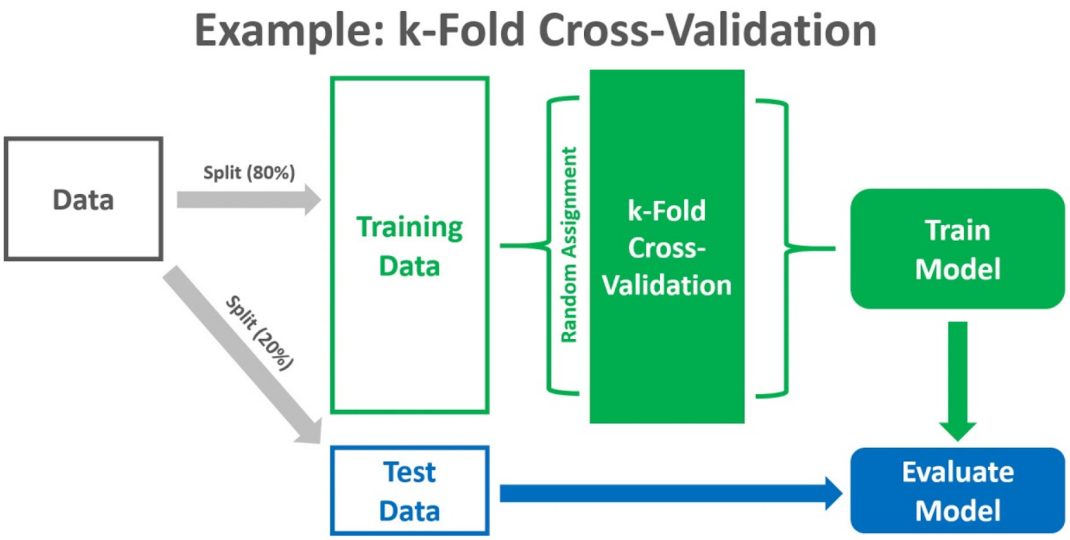

FIGURE 3.17: WORKING OF CROSSFOLD

Why are we actually dividing the data set is that if we are training the model on a particular data and then again testing on that data only then this dataset would be very fimiliarising to the model and so the results would look very accurate however when we give a new set of data to the model then only we would be able to know whether or not thee model is working fine on the new data or not. We in our model are using K cross fold validation in which we are actually diving the dataset in k part and then we are using k-1 subset of the data to train the model and finally use the kth data subset to actually test the model on and find out how accurately our model is working.

To find out best of the neighbours we sould be using V fold cross validation but unfortunately we can't use that with feature selection.
1.Randomly assigning to fold: We decide how many folds we want to cross validate and then the system then randomly put all the cases in the particular folds which are given heading 1 to V.

2.Setting of random seed:This particular feature is used to get some consistant resuts during the different sessions so this will ensure that even after the process is random but we would still get the same results each time we are running the system.
3.Using the field to assaign different cases: A numerical field is created to show which fold cases belong to fild values between 1 to V.

If we want to define in some simple steps which are followed in cross validation then they are:
1.Place some of of the data portion aside: This set aside data part would be used to finally vildate if what model we are using is fine or not.
2. Training the model: Major portion of the data is used to train the model well enough so that iit is able to generalize all the patterns visible in the data so as to predict accurately.
3.Evaluation of the model: We would finally after training the whole set with the training dat we would be testing on the validation data and ensure if the model is working fine with unseen data or not.

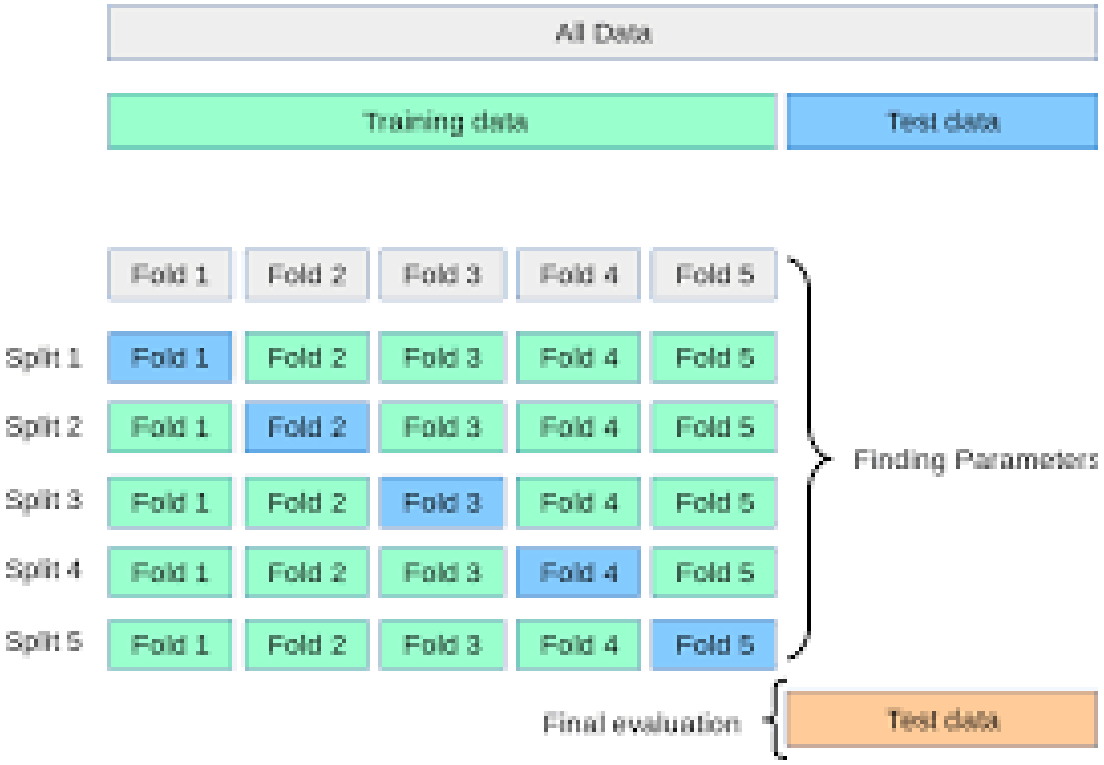

FIGURE 3.18: WORKING OF CROSSFOLD

- Gradient Boosting

  What is Boosting?

  Gradient Boosting elucidates the concept of Boosting, which stands as a formidable solution to challenges encountered in the training phase. It operates as an ensemble learning technique, amalgamating multiple feeble learners to forge a robust learner. In the iterative process of boosting, a randomly selected data subset undergoes fitting with a model, and subsequently, each model in the sequence endeavors to compensate for the deficiencies of its predecessor. The amalgamation of weak rules from individual classifiers during each iteration forms a unified and potent prediction rule.

  Ensemble Learning

  The theory of the ”wisdom of crowds” posits that decisions from a larger collective often surpass those of an individual expert. This concept finds practical application in Ensembles, where a gathering of basic learners collaborates to yield a more precise ultimate prediction. In essence, a single model, termed a base or weak learner, might struggle in isolation due to excessive variation or pronounced bias. Nonetheless, through the assembly of these less proficient learners, a superior learner emerges, benefitting from the amalgamation that mitigates bias or variance and thereby enhances overall model performance.

  Illustrating ensemble methods, decision trees serve as a prime example due to their susceptibility to overfitting (”high variance and low bias”) in the absence of proper pruning, or underfitting (”low variance and high bias”) when constrained in size, such as decision stumps, characterized by a mere one level. Ensemble methods act as a preventive measure against these tendencies, ensuring the model’s adaptability to novel datasets. It’s crucial to note that when an algorithm succumbs to ”overfitting or underfitting its training dataset,” the outcomes tend to be suboptimal when applied to previously unseen data. Given the proclivity of decision trees to manifest substantial bias or high variance, it becomes pertinent to acknowledge that various modeling techniques leverage ensemble learning to pinpoint the elusive ”bias-variance sweet spot.”

  Bagging VS Boosting

  Ensemble learning techniques primarily fall into two categories: boosting and bagging. This examination sheds light on the fundamental distinction in the training methodologies among various learning approaches. Bagging, characterized by parallel instruction of weak learners, stands in contrast to boosting, where strong learners are sequentially trained. Consequently, multiple models are constructed, with each successive iteration accentuating the weights of inaccurately classified data. This redistribution of weights empowers the algorithm to discern the parameters that demand heightened focus, thereby enhancing its performance. Noteworthy among boosting algorithms is AdaBoost, denoting ”adaptive boosting algorithm,” which carved its niche upon its inception. Additionally, there’s a cohort of boosting techniques including XGBoost, GradientBoost, and others such as BrownBoost.

Another differentiating factor lies in how bagging and boosting are employed. Unlike boosting, frequently employed in scenarios characterized by low variance and high bias, bagging methods find application with weak learners exhibiting high volatility and low bias. The occurrence of more overfitting with boosting approaches, as opposed to bagging, is contingent on the dataset. Yet, the mitigation of this issue is achievable through adept parameter adjustments.

Consequently, bagging and boosting find diverse applications in the real world. Boosting, with its effectiveness, is prominently applied in search engines and image processing, particularly in recognition tasks. On the other hand, bagging has found practical utility in fields like statistical genomics and the intricate processes of loan approval.

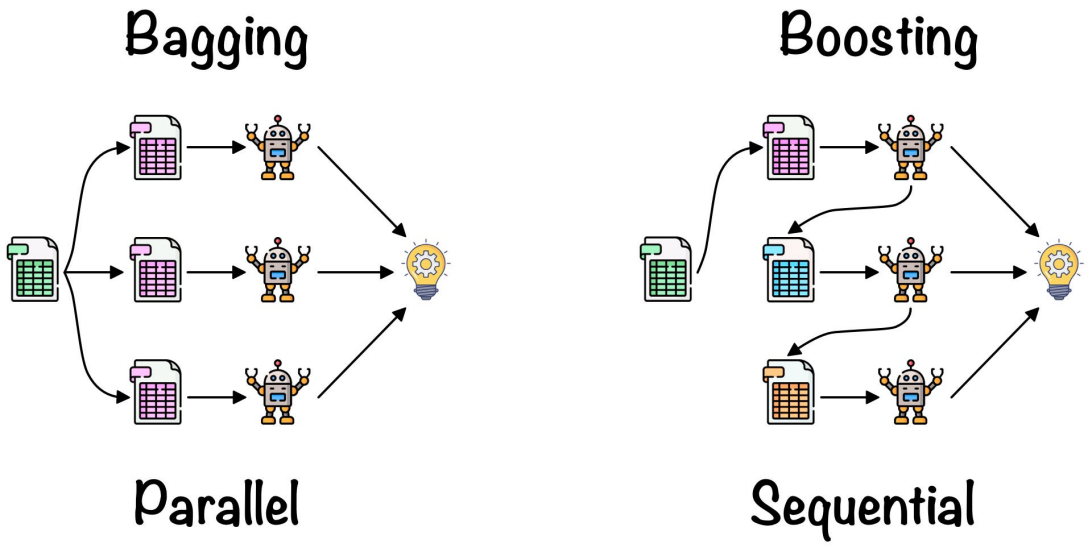

FIGURE 3.19: Bagging and Boosting

Types of boosting

Boosting strategies center on iteratively amalgamating ”feeble learners” to craft ”a formidable learner,” culminating in enhanced predictive capabilities. It’s crucial to note that a feeble learner slightly outperforms chance when classifying data. In endeavors such as image retrieval, this approach has the potential to outperform ”neural networks and support vector machines” in yielding resilient results.

Following are the different ways we could apply boosting

(a) Adaptive boosting or AdaBoost:

Developed by Yoav Freund and Robert Schapire, the AdaBoost algorithm employs an iterative strategy to diminish training errors. It systematically identifies misclassified data points and adjusts their weights to curtail errors. The model undergoes sequential

refinement until it yields the most potent predictor.

(b) Gradient Boosting:

Conceived by Jerome H. Friedman and Leo Breiman, Gradient Boosting is a methodical approach that incrementally introduces predictors to an ensemble. Each new predictor aims to rectify the errors of its predecessor. In contrast to AdaBoost, which adjusts the weights of individual data points, gradient boosting focuses on training with the residual errors from the preceding predictor. The nomenclature ”Gradient Boosting” stems from the amalgamation of the gradient descent algorithm and the boosting method in its execution.

(c) Extreme gradient boosting or XGBoost:

XGBoost, short for ”Extreme Gradient Boosting,” is a system meticulously crafted for both speed and scale within the realm of gradient boosting.

Benefits and challenges of boosting

As much as any other method the boosting method also has its fair share of advantages and disadvantages that we have been listed below:

Some of its advantages are:

• Ease of Implementation:
Boosting offers a seamless integration with numerous hyper-parameter tweaking options, facilitating enhanced fitting. Given that boosting methods, such as AdaBoost and XGBoost, come equipped with built-in mechanisms for handling missing data, the need for extensive data preprocessing is obviated. The simplicity of implementing well-known boosting techniques in Python is further underscored by the user-friendly features of the scikit-learn library for ensemble methods (also recognized as sklearn.ensemble).

• Reduction of bias:
Boosting algorithms, through the sequential amalgamation of multiple weak learners, progressively refine predictions. This iterative process proves instrumental in mitigating excessive bias, a common challenge encountered in logistic regression models and shallow decision trees.

• Computational Efficiency:
Boosting algorithms can aid in reducing dimensionality and improving computational efficiency because they only choose characteristics during training that improve their predictive value.

The main problmes that we need to tackle in boosting are as follows:

• Prone to Overfitting:
The issue of overfitting sparks a debate regarding whether boosting acts as a mitigating factor or exacerbates the problem. Classified as a challenge, if overfitting does occur, the resultant outcomes may not be reliable for extrapolation to newer and unseen data.

• High Computational Demand:
Scaling "sequential training in boosting" presents a formidable challenge. Boosting models can incur significant computational expenses as each estimator builds upon its predecessors. However, XGBoost strives to address scalability issues evident in other types of boosting approaches. In contrast to bagging, training times for boosting algorithms may be extended due to the impact of a larger set of parameters on the model's behavior.

- **Gradient Boosting:** Gradient Boosting proves versatile in addressing both classification and regression tasks. It furnishes a prediction model in the guise of an assembly of weak prediction models, often resembling decision trees. Particularly noteworthy is the superiority exhibited by the resulting method, termed gradient-boosted trees, which frequently outperforms random forest when decision trees serve as the weak learners. The tree structure in the gradient boosting algorithm is constructed in a stage-wise manner akin to earlier boosting approaches. However, it distinguishes itself by extending those techniques to enable optimization for any differentiable loss function.

  This approach incorporates a crucial parameter known as shrinkage. Shrinkage occurs when the prediction from a tree in the ensemble is scaled by the learning rate (eta), which falls within the range of 0 to 1. The act of scaling is aptly termed shrinkage. Achieving a desired level of model performance entails maintaining a balance between the ratio of estimators to eta; a reduction in the learning rate necessitates an augmentation in the number of estimators. Following the training of all trees, predictions become feasible. Each tree contributes to the final prediction through the formula:

  y(pred) = y1 + (eta * r1) + (eta * r2) + + (eta*rN)

  In scikit-learn, the class name for gradient boosting regression is Gradient Boosting Regressor, while Gradient Boosting Classifier is employed for classification tasks, both employing a parallel approach.

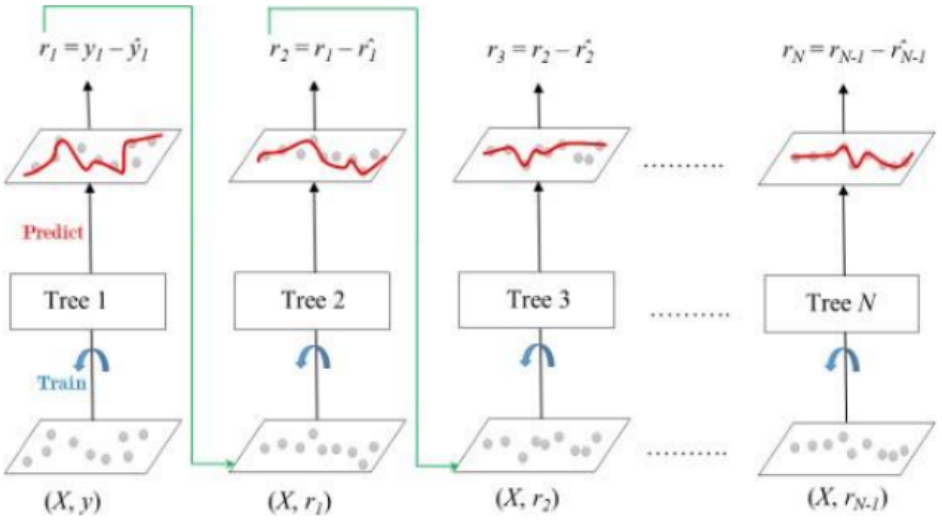

FIGURE 3.20

Applications of Gradient Boosting

Applications of Gradient Boosting extend across a myriad of industries, proving particularly well-suited for various artificial intelligence initiatives, encompassing:

- **Healthcare:** In the realm of healthcare, boosting plays a pivotal role in predicting medical outcomes, such as assessing the likelihood of cancer survival or the risks associated with cardiovascular conditions. Utilizing boosting techniques aids in minimizing prediction errors. Research indicates notable advancements, showcasing that ensemble approaches significantly enhance accuracy in identifying individuals who stand to benefit from preventative cardiovascular disease therapy, while concurrently sparing unnecessary treatment for others. Boosting applied to various genomics platforms has demonstrated potential in improving the prediction of cancer survival time, as evidenced by additional studies in the field.

- **IT:** The "Viola-Jones boosting algorithm" finds application in image retrieval within the IT domain. Additionally, search engines leverage gradient boosted regression trees for optimizing page rankings. As per findings from Cornell, boosted classifiers offer the advantage of early termination of calculations when it becomes evident which path the computation should follow.

  This implies that image scanners, guided by boosted classifiers, consider only photographs genuinely containing the sought object. Similarly, search engines employing gradient boosted regression trees can cease the evaluation of lower-ranked pages once the direction of a prediction becomes evident.

- **Finance:** The fusion of deep learning models with boosting proves instrumental in automating critical activities within the finance sector. This amalgamation is particularly

impactful in areas such as fraud detection and pricing analysis. The enhanced accuracy in evaluating vast datasets contributes to minimizing financial losses. Notably, procedures in credit card fraud detection and financial product pricing analysis see improvements through the application of boosting techniques.

Chapter 4

Proposed Work

4.1 Proposed Work

Throughout time, a plethora of machine learning techniques and models have been applied to scrutinize the stock market, aiming to prognosticate stock prices by leveraging diverse algorithms and accessible data. Despite these advancements, the realm of Initial Public Offerings (commonly referred to as ”IPOs”) remains untouched by these predictive endeavors. An IPO fundamentally signifies the initial issuance of shares from a privately held company to the public, a transformative process shifting ownership from private to public spheres. Consequently, the procedural shift of an IPO is colloquially termed as ”going public” in the financial realm.

To date, an algorithm or machine learning model capable of accurately predicting the opening price of an IPO remains elusive. Published papers have delved into assessing the underpricing and overpricing tendencies associated with specific IPOs. Drawing insights from these references, we've meticulously analyzed the multitude of factors, attributes, and prevailing conditions that influence the fluctuation of IPO prices, discerning patterns of ascent or descent. Inspired by these insights, we've pioneered our own suite of models, leveraging the concepts and attributes gleaned from over a decade of scrutinizing the history of Indian IPOs, all in pursuit of forecasting the listing open price of an IPO.

Chapter 5

Simulation and results

5.1 Simulation and Results

5.1.1 The results of different algorithms used:

The subsequent segments encapsulate comprehensive results linked to individual algorithms along with their associated particulars. Commencing with an amalgamated summary of all employed algorithms and their respective methodologies, the section progresses to delineate visual representations showcasing the contrast among these algorithms. These visual aids aim to elucidate the comparative analyses, shedding light on the models' utilization, performance, and ultimately identifying the most optimal solutions derived from our endeavors.

1. Compiled Results
The table provided below encompasses a comprehensive catalog of the algorithms employed, accompanied by the corresponding accuracy metrics utilized for each algorithm. It further delineates intricate details pertinent to every model and algorithm. The accuracy measurements for each process have been meticulously documented post execution through the dataset. These recorded results stand independent, derived from a solitary run for each model, devoid of influence from other models or biases.These accuracy metrics represent individual executions of the models and serve as standalone evaluations without interdependencies or partiality towards any particular model.

Accuracy Metrics:

accuracy score The 'sklearn.metrics accuracy score()' function within the 'scikit-learn' library is widely used by Python practitioners to calculate the accuracy scores of predictive models. This function accepts two sets of labels: the actual (real) labels of the samples and the predicted labels generated by the model. By comparing these sets, it computes the accuracy score, which is returned as a floating-point value.

cross val score Certainly, the method you're describing is known as cross-validation, a technique used in machine learning to assess a model's performance across multiple iterations or folds of the dataset. Unlike a single train/test split, cross-validation provides a more robust evaluation by partitioning the dataset into several subsets (folds) for training and testing. One popular method for cross-validation is k-fold cross-validation. In k-fold cross-validation, the dataset is divided into k equal-sized folds. The model is trained and evaluated k times, each time using a different fold as the test set and the remaining folds as the training set.

R2 Score The R2 regression score, also known as the Coefficient of Determination, operates on a scale where the optimal score is one, but it can also dip into negative territory. Negative outcomes arise when models yield notably poor results. An R2 score of 0.0 is assigned to a "constant model," one that steadfastly foretells the expected value of y while overlooking the intricacies of the input characteristics.

gbr.score The GradientBoostingRegressor fabricates "an incremental model through a progressive stage-wise method, facilitating the optimization of any differentiable loss functions. At each level, a regression tree is intricately molded based on the negative gradient derived from the specified loss function." This model is equipped with a scoring mechanism that furnishes the accuracy percentage pertinent to the model in question. In this context, 'gbr' serves as the nomenclature for the model, meticulously crafted as 'gbr = GradientBoostingRegressor(**gbr params).'

- **Results Table:**

Algorithms	Accuracy	Accuracy Metric Used
Random Forest (n estimators=100)	80.56	accuracy_score
Random Forest (n estimators=50)	77.37	accuracy_score
KNN (cross val)	61.63	cross_val_score
KNN (normalized)	74.03	accuracy_score
Decision Tree (depth = 10)	60.07	accuracy_score
Prediction with Intervals	68.45	r2_score
Gradient Boosting	84.5	gbr.score

FIGURE 5.1

- **Results Visualized:**

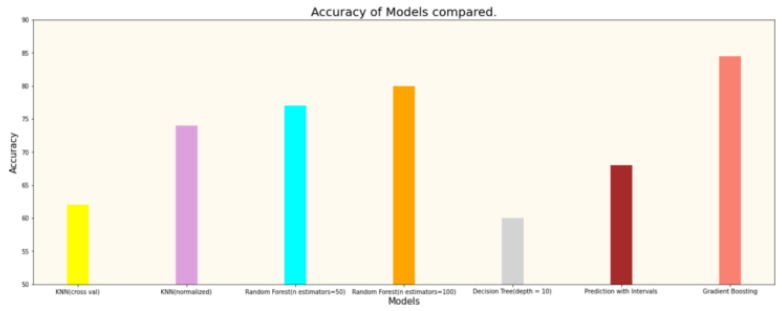

FIGURE 5.2

2. Visualized Results of Algorithms:

This segment showcases visual representations elucidating the correlation between the target variable—the Open Price of each company in the dataset—and the predicted Open Price for one-third of the IPO companies utilized as test data. Employing the train-test split methodology, a fraction of the data, precisely one-third, was earmarked for testing. The graphs meticulously compare the Open Price for this subset against both the actual and predicted Open Prices.

Through the process of organizing data into an intelligible format and unveiling patterns and outliers, data visualization serves as a narrative tool. A robust visualization not only emphasizes crucial information but also minimizes the noise inherent in the data. However, the enhancement of a graph's aesthetics or the addition of informational components to an infographic is not a straightforward task. Effective data presentation demands a meticulous equilibrium between form and function.

Data visualization offers users a means to perceive intricate data visually, such as through graphs or charts, unveiling trends that may elude detection through statistics alone. A paramount quality of data visualization lies in its proficiency to discern patterns within datasets. For many, it holds true that "when data is presented visually rather than in tabular form, spotting trends becomes much more straightforward."

Random Forest

Y axis - “Predicted Open price”. X axis - “Number of Companies”

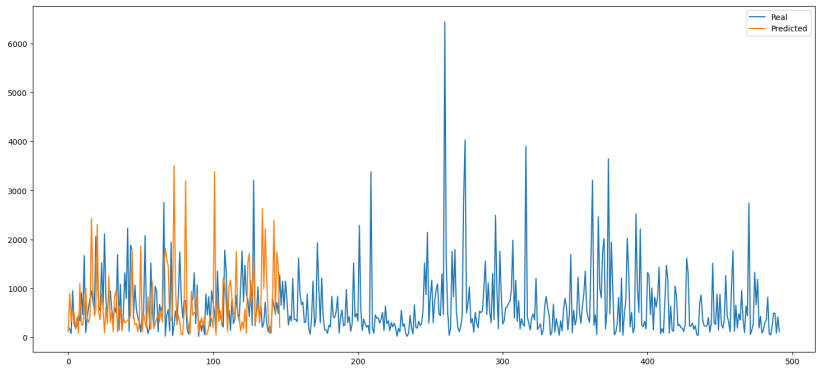

FIGURE 5.3

KNN

Y axis - “Predicted Open price”. X axis - “Number of Companies”.

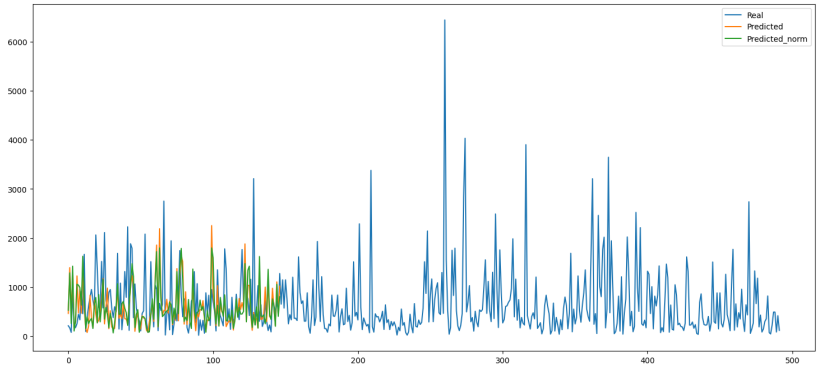

FIGURE 5.4

Gradient Boosting

Y axis - “Predicted Open price”. X axis - “Number of Companies”

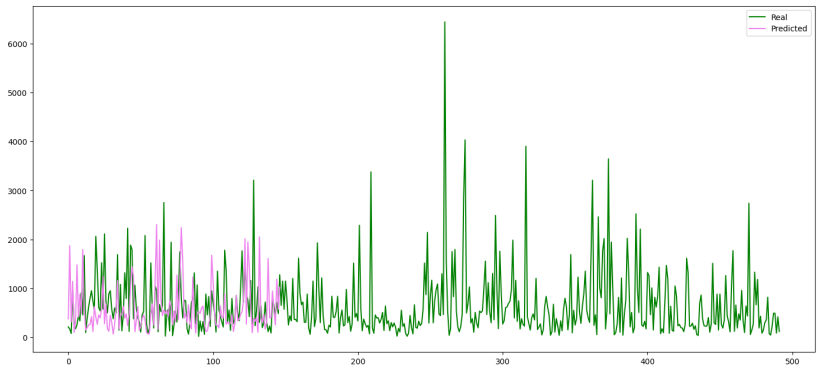

FIGURE 5.5

Prediction with Intervals

X axis is the ”Issue Price” and Y axis is the listing ”Open Price” , with high and low limit

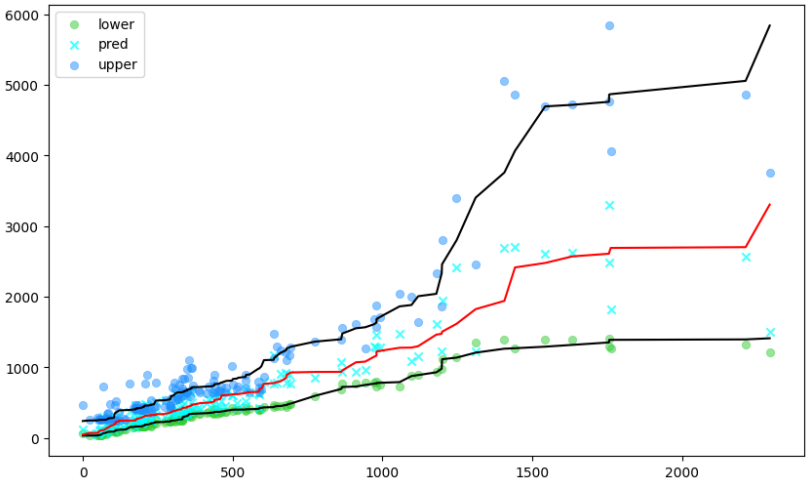

FIGURE 5.6

Chapter 6

Conclusion

In summary, we managed to identify certain factors—albeit not an exhaustive list—that directly influence the Opening Price of an IPO. The ”Issue price” emerges as a pivotal element, exerting a direct impact on the initial market value. Additionally, our exploration unveiled the substantial influence wielded by the underwriters of a given IPO, shaping its performance in the subsequent trajectory.

The utilization of diverse machine learning models in our study enabled the prediction of the Listing Open price for the IPOs, each model showcasing distinct advantages and disadvantages. Yet, this domain proves expansive, with room for refinement and exploration. There exist untapped avenues, such as pinpointing precise inflection points in price dynamics, which, when harnessed effectively, could enhance our predictive capabilities.

Nonetheless, a comprehensive examination of economic conditions and various influencing factors is imperative for a nuanced understanding. Delving deeper into these facets can refine our comprehension of the market and the specific needs of an IPO. This, in turn, aids in the meticulous selection of optimal models, ensuring heightened accuracy for the ultimate benefit of stakeholders—a goal that remains paramount in this multifaceted landscape.

Glossary:

Issue Price - “The price at which shares are offered for sale when they first become accessible to the public” is known as the issue price.

Open Price - The price at which freshly issued securities begin trading on an exchange on the first trading day.

Oversubscribed - An agreement that is oversubscribed refers to one in which more investors apply for shares than there are available. Usually a hint that an initial public offering is a popular one and will launch at a significant premium.

Underwriters - A firm looking to “issue shares in an initial public offering (IPO) and investors” are connected by an “underwriter” in a new stock offering. “The underwriter aids in the firm’s

IPO preparation by taking into account things like the amount of capital to be raised, the kind of securities to be issued, and the terms of the underwriter's contract with the company".

Underpricing - "When an IPO stock closes at a higher price than the initial offer price on the first trading day", is what is known as underpricing.

Bibliography

Here are the sources we used in our project which contains the sites where we collected our datasets and from where we derived inspiration as well:

1. https://www.moneycontrol.com/ipo/ipo-historic-table?classic=true

2. https://www.iposcoop.com/last-12-months/

3. https://www.investopedia.com/articles/financial-theory/11/how-an-ipo-is-valued.asp

4. https://www.sciencedirect.com/science/article/pii/S2214845019302686

5. https://www.angelone.in/knowledge-center/ipo/types-of-ipo-investor

6. https://www.iimidr.ac.in/wp-content/uploads/Underpricing-of-Initial-Public-India.pdf

7. http://hj.diva-portal.org/smash/get/diva2:531502/FULLTEXT01.pdf

8. https://www.chittorgarh.com/report/report-list/116/49/

9. https://towardsdatascience.com/getting-rich-quick-with-machine-learning-and-stockmarket

10. https://towardsdatascience.com/decision-trees-explained-entropy-information-gaingini-c

11. https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/

12. https://machinelearningmastery.com/regression-metrics-for-machine-learning/

13. https://scikitlearn.org/0.17/modules/generated/sklearn.metrics.r2 score.html

14. https://towardsdatascience.com/all-you-need-to-know-about-gradient-boostingalgorithm

15. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.

16. https://www.javatpoint.com/accuracy score-in-sklearn

17. https://www.ibm.com/in-en/topics/decision-trees

18. https://emeritus.org/blog/data-analytics-what-data-collection/

19. https://www.imperva.com/learn/application-security/web-scraping-attack/

20. https://www.ibm.com/in-en/topics/exploratory-data-analysis