Predicting survival status of lung cancer patients using machine learning

Predicting survival status of lung cancer patients using machine learning

By

Aishwarya Mohan

A thesis

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES OF MCMASTER UNIVERISITY

IN PARTIAL FULFILMENT OF THE REQUIRMENTS FOR THE DEGREE OF MASTER OF APPLIED SCIENCES

Master of Applied Science (2021) McMaster university

Electrical and computer engineering Hamilton, Ontario, Canada

TITLE: Predicting survival status of lung cancer patients using machine learning

AUTHOR: Aishwarya Mohan

SUPERVISOR: Dr. Aleksandar Jeremic

NUMBER OF PAGES: x, 111

Abstract

5-year survival rate of patients with metastasized non-small cell lung cancer (NSCLC) who received chemotherapy was less than 5% (Kathryn C. Arbour, 2019). Our ability to provide survival status of a patient i.e. Alive or death at any time in future is important from at least two standpoints: a) from clinical standpoint it enables clinicians to provide optimal delivery of healthcare and b) from personal standpoint by providing patient's family with opportunities to plan their life ahead and potentially cope with emotional aspect of loss of life.

Acknowledgement

First, I would like to thank my supervisor, Dr. Aleksandar Jeremic who has diligently guided me within the last two years. The door to his office was always open whenever I faced any challenge in my research.

I wish to dedicate this thesis to my wonderful parents, who have sacrificed and invested 27 years of their lives for me to grow and learn. Thank you for always believing in me and motivating me to try harder.

Table of Contents

List of figures

Chapter 1
Introduction

According to American Cancer Society, lung cancer is the leading cause of cancer death among men and women, for almost 25% of all cancer deaths. As the mortality rate of lung cancer is high, it belongs to a group that has the worst survival prognosis (Camilla Mattiuzzi, 2019).

Generally, after diagnosis the patient's family expects to know the patient's chances of survival from a clinician. An ability to predict life expectancy can be beneficial from both emotional standpoint and clinical standpoint, as it reduces stress on patient’s family to cope with situation and can also allow clinicians to evaluate patients' risk, likelihood of survival and postoperative treatment procedures. Due to the very nature of the disease, lung cancer datasets are generally imbalanced where majority of patient population has low chances of survival. As a result, predictive modeling on imbalanced datasets where the majority of patients have low chances of survival (Liang, 2019) makes it more challenging to accurately predict survival status of patients with higher chances of survival. Thus, for clinicians to accurately evaluate patients’ risk and further

design appropriate post treatment procedures it is equally important to accurately predict both true negatives and true positives.

Increasing diagnostic lab tests indicates a potential of vast biomedical data assuming there are plenty of electronic health records of patients. As a result, rapid increase in volume and complexity of biomedical data can be utilized to draw patterns and inferences. One of the promising techniques that can be helpful in finding patterns from a large patient cohort data is predictive modeling which utilizes biomedical data to investigate relationships between the factors and the dependencies that further help us predict survival status of patient. Ultimately, this can help patients with personalized medication and risk assessment. Developing algorithms and mathematical models that can generate reliable predictions on an imbalanced dataset is a daunting task because of the underlying dependencies and bias which can be complex. As a result, number of factors influencing the predictions are huge. To implement this technique in medical practice we need rigorous training procedures for complexities. Even in this case, the underlying assumption of these techniques is that certain statistical/probabilistic models can describe these dependencies which may not be true in certain cases (i.e., there may exist certain number of outliers in every dataset). In addition, we need to design vigorous testing, validation, and verification procedures because of overwhelming intricacies such as variability from patient- to -patient that needs to be evaluated.

1.1 Motivation

Unequal distribution of data between majority class 1 (Dead) i.e. patients that are less likely to survive and minority class 0 (Alive) i.e. patients that are likely to survive can induce bias towards majority class, leaving minority class samples to be often misclassified. Misclassification of minority class can lead to hectic post-operative treatment procedures, high dosage of recommended drugs and accelerated health follow-ups and diagnostic tests which can cause stress both physically and psychologically.

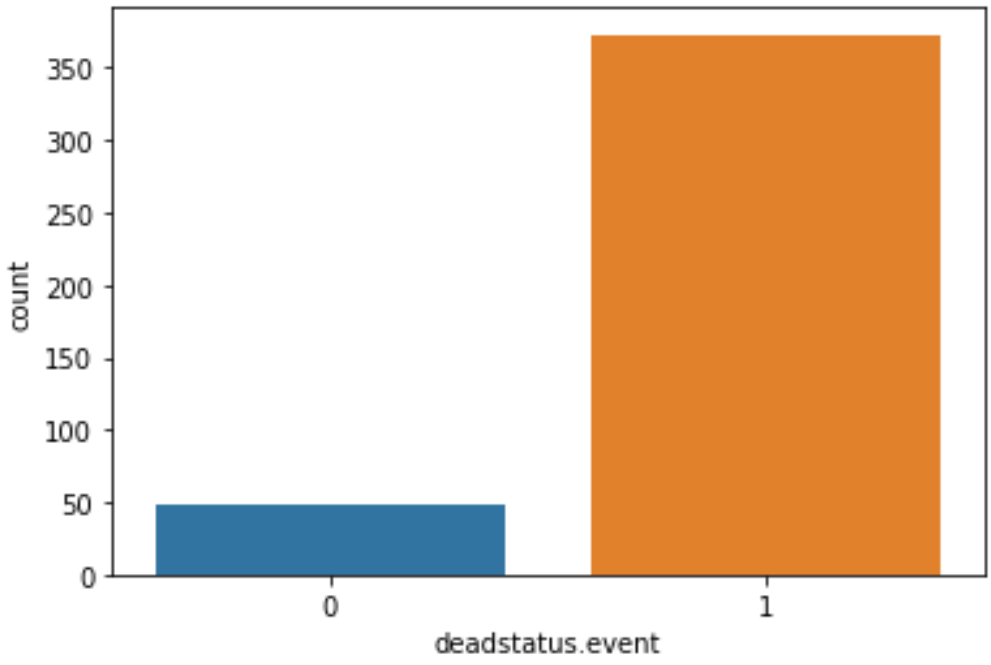


Figure 1: Class count for non-small cell lung cancer data

An ability to correctly predict survival status of patient at a given time by clinician can alleviate this stress. Hence, to use machine learning models in clinical practice they should be designed in such a way that they are robust towards bias induced by majority class. These models play an important role in

developing a computerized decision-making support systems for clinicians. Prediction of survival status at a given point in time reflects risk status of patient. These models can also be used as a risk assessment tool to help us determine which patients should be offered imaging. However, all these tools suffer from aforementioned common challenge of bias towards majority class. Furthermore, they are also dynamic in nature and needs to be updated continuously as the environment changes. Hence, model should be constructed and designed in such a way that it can adjust if there are changes in the subset of the population. However, the importance of different factors varies at different stages of treatment. The predictive values of different features for different patients may be changed. From clinician's perspective they would want various things such as predicting results, optimizing the treatment, healthcare planning and patients planning. Models are developed based on data that includes patient events, relationships between factors, parameters, treatments and outcome events. Laboratory indicators such as results of biochemical tests can also be used in model development.

1.2 Challenges

Performance of machine learning models varies from field to field as well as from application to application. For example, financial institutions that use analytics as a service (AaaS) focus on trade-offs between false positives and false negatives and as a result, model prediction is sent to a queue which is then analyzed by risk analysts before coming to a decision whereas an automobile

company working on a vision related project generally uses confidence score as a threshold in each detection for making a prediction. Often machine learning results are compared to that of an expert opinion leading to a hybrid decision making system in some industries. The comparison is still under investigation as machine learning systems take less time, are cheaper to deploy and perform better in some areas. Consider this scenario, when a human mind is going through 10,000 records with 100 feature columns, it is highly likely for mind to lose track of patterns and going through such a cumbersome process is not feasible. On the other hand, machines can perform these computations faster and draw patterns by scanning through high dimensional matrices quickly.

Generally, medical datasets regardless of its size cannot cover all the possible cases as the problem is stochastic in its essence. Hence, clinicians often estimate the overall survival (OS) or progression free survival (PFS) with a confidence interval. One of the reasons behind this is that there are no pre-defined guidelines that a certain parameter decides survival prediction. Similarly, when it comes to different categories of physician, every clinician gives a different survival time estimation for patients in a recent study. The difference in accuracy of survival estimates by clinicians among the three categories of physicians i.e., residents, younger physicians and registrars needs explanation and basis. Physicians overestimated the survival status of patients, albeit resident doctors were the most accurate (Christelle Clément-Duchêne, 2010).

1.2.1 Training challenges

One of the many current challenges in translating machine learning algorithms to clinical practice are related to the fact that healthcare data is not readily available for machine learning. Datasets are often siloed in a multitude of medical imaging archival systems, pathology systems, EHRs, and insurance databases which are difficult to bring together. Adapting unified data formats, such as Fast Healthcare Interoperability Resources (Joshua C Mandel, 2016), offer better aggregation of data, albeit improved interoperability does not necessarily fix the problem of inconsistent semantic coding in EHR data (William R Hersh, 2013).

1.2.2 Evaluation challenges

1.2.2.1 Adaptability of model results by clinicians

Metrics currently provided by software's used in Machine learning are not in concordance to data useful to clinicians. Accuracy results from a machine learning model does not mean model will be efficient when used in medical practice (Pearse A. Keane, 2018) and is not easily understood by many clinicians. There is no single metric that captures all the properties of a model. However, metrics are typically used to summarise model's performance.

1.2.2.2 Benchmarked dataset

The comparison of algorithms across studies is challenging due to each study's performance being reported using variable predictive modelling techniques on different populations with different sample distributions and properties.

Algorithms need to be subjected to comparison on a benchmarked dataset that is representative of the inference population, using same evaluation metrics. Without this, clinicians will have difficulty determining which algorithm is likely to perform best for their patients.

1.2.3 Model issues

1.2.3.1 Bias and Variance trade-off

Outliers in machine learning can reflect ignored society biases, with a risk of unknown accuracies in minority subgroups, posing a risk of amplifying biases present in the historical data (Kate Crawford, 2016). Algorithmic bias can be categorized into three fields: model bias, model variance, outcome noise (Irene Chen, 2018). A greater awareness of these risks and biases is needed to empower clinicians to participate critically in system design. Ultimately, this will help guide researchers to ensure correct order of steps is followed that quantify bias before deploying models. Vigorous validation on metrics by population subgroups should be performed including age, ethnicity, gender, socio-demographic status and location.

1.2.3.2 Rigorous quality control

In order to have safe and efficient deployment of machine learning algorithms formulation of necessary regulatory frameworks are important (Christopher J. Kelly, 2019). This is challenging given the current pace of innovation, risks involved and dynamic nature of machine learning models. These regulations will give confidence to clinicians and healthcare systems. U.S. Food and Drug

Administration guidance has begun developing regulations ensuring safe deployment of artificial intelligence devices to patients (Food and Drug Administration. Proposed Regulatory Framework for Modifications to Artificial Intelligence/Machine Learning (AI/ML)-Based Software as a Medical Device (SaMD), 2019).

1.2.3.3 Adoption in healthcare

In order to ensure machine learning solutions can reach patients, it is important to focus on clinical application, patient outcomes, develop advance methods for algorithmic interpretability and achieving better understanding of human–computer interaction (Christopher J. Kelly, 2019).

1.3 Thesis structure

In this thesis, we investigate different approaches for predicting survival status of patients suffering from non-small cell lung cancer. In Chapter 2, we review background of machine learning and related work in cancer prediction followed by steps to follow before applying machine learning classifiers to training dataset. In chapter 3, we present different classifiers on which our analysis will be performed and later in the chapter we list evaluation metrics for measuring performance. In chapter 4, related dataset and results from different tests performed on training data will be discussed. In last chapter, we conclude our findings for this study and present suggestions for future work.

Chapter 2

Background and related work

2.1 Machine learning background

Machine learning has been used in healthcare for nearly 30 years. In other words, machine learning is being used as a tool in cancer diagnosis, detection, prediction, and prognosis (Joseph A. Cruz, 2007) from a long time.

Aim of cancer prediction and prognosis are different from aim of cancer detection and diagnosis. In cancer prediction or prognosis one is concerned with three goals: 1) risk assessment 2) cancer recurrence and 3) cancer survivability. In the first case, we are trying to predict the likelihood of developing a type of cancer before occurrence of the disease (Joseph A. Cruz, 2007). In the second case, we are trying to predict recurrence of cancer once its resolved. In the third case we are trying to predict an outcome such as life expectancy and survivability chances. In last two situations success of the prognostic prediction is dependent on the quality of the diagnosis.

Machine learning is a subset branch of artificial intelligence that learns from past examples later uses that learning to identify patterns or predict trends in the data. Machine learning methods can utilize Boolean logic, absolute conditionality, conditional probabilities (the probability of X given Y) and unconventional optimization strategies for model building and identifying patterns which are not employed in statistics (Joseph A. Cruz, 2007). These latter methods resemble the approaches that is used naturally by humans and animals i.e., learning from experience. Machine learning uses concepts from statistics and probability, but it is more powerful because as it allows inferential learning which is not possible using conventional statistical methodologies (Richard O. Duda, 2001). Although powerful, these approaches assume that the variables are independent, and that data is linear i.e., it can be modeled using linear combinations of these variables. However, when the relationships are non-linear, and the variables are interdependent (or conditionally dependent) conventional statistics is not sufficient. Under these circumstances machine learning tends to offer promising results. Many biological and environment systems are non-linear, and their parameters are also conditionally dependent. Many physical systems are linear, and their parameters are essentially independent. As with any method, one should understand the limitations and assumptions associated with dataset and the algorithms being applied. If a machine learning experiment is properly designed taking all assumptions into account, implemented, with result validation and testing, there is a good chance of success for the problem at hand. However, in many cases that is not the case.

For example, as dimensionality in a dataset increases, variables span a lower-dimensional subspace in which some of the dimensions become redundant in terms of other dimensions, thus yielding perfect multiple correlation (Lever, 2017).The only solution is to reduce the number of features or increase the number of training records. As a general rule, the records-per-feature ratio should always exceed 5:1 (Somorjai RL, 2003). Size of the training set is as important as the variety of the training set. Training observations should be selected in a way such that it is similar to what classifier expects to encounter during inference. Training repeatedly on less examples with too low variability leads to over-training or training on noise (Rodvold DM, 2001). An over-trained classifier, will generally perform poorly when it tries to identify patterns in unseen data. There are cases when conventional statistics becomes more powerful or more accurate than machine learning due to wrong initial notions about the interdependence and non-linearity of the data. Similarly, not all machine learning methods are equal. Some are better for certain kinds of problems while others are better for other kinds of problems. For instance, some machine learning algorithms are scaled to the size of the biological domains, others do not. Similarly, some methods may have assumptions or data requirements that render them inapplicable to the problem we are trying to solve. Performance of machine learning algorithms depends on both the dataset and the algorithms, finding an optimal solution can be a challenging. Therefore, it is critically important to try more than one machine learning method on any given training set.

Machine learning algorithms can be classified into three categories:

1) supervised learning

2) unsupervised learning

3) reinforcement learning.

Supervised learning are algorithms that involve direct supervision of a target label. In this case, a person labels sample data and sets strict boundaries upon which the algorithm operates. The model learns through perception and identifies structures in the information. When the model is given a dataset, it consequently discovers patterns in the dataset.

Depending on the type of outcome one can choose suitable machine learning methods for model building. Supervised machine learning methods can be used in two kinds of applications regression and classification. Regression analysis is used to estimate a target variable based on a set of features by estimating the relationships between a dependent variable i.e., outcome variable and one or more independent variables called predictors, or features. Classification is a process of categorizing data into respective classes. These classes are often referred to as target, label, or categories. The whole process of predictive modeling involves series of steps ranging from data pre-processing, feature correlation, feature selection, model building and testing.

2.2 Data pre-processing

2.2.1 Missing data in clinical records

Clinically captured data often presents the challenge of missing data leading to bias or negative impact on analytical outcomes (Harshad Hegde, 2019) .For example, failure of staff to consistently document a value in an electronic health record, or technical failures precluding data capture by a device designed to track specific data, or capture of data mainly in unstructured formats not mined electronically without manual abstraction or pre-processing, making these data not readily available for analysis. Data that is not consistently recorded leads to missing records and consequently limiting the analysis.

Machine learning algorithms may not perform well on missing data as they depend on finding relationships between the variables. For a dataset that is very large and the number of missing values in the data are very small (less than 5%), these values can be ignored and analysis can be performed on the rest of the data (Janus Christian Jakobsen, 2017).

Sometimes in lung cancer datasets, there is a feature column for smoking. Generally, not all the records have an entry for that field and information for some patients is missing. This can be because person diagnosed with lung cancer did not smoke or there can be other reasons for an empty entry. However in our dataset all the missing values are less than 5% and are omitted for our analysis.

2.3 Feature selection

Electronic health records (EHRs) contain variety of information about patients and their stays in health facilities (Elyne Scheurwegs, 2017). Clinical codes reflect diagnoses and procedures related to a patient stay and are primarily assigned for reporting and reimbursement purposes. Their widespread adoption in hospitals makes them a viable information source in research and monitoring applications.

However, not all the information present in the dataset may contribute to estimating the outcome and may result in information overlap. Using feature selection, we can prevent less information overlap and use it to provide a dense representation of contributing features.

Feature selection is the process of selecting the most relevant features for building a model such that the performance of the model is not affected. For example, in a model that inferences on predicting survival status of a patient at a given time, two features such as survival time and date of demise are highly correlated, resulting in information overlap. Removing multicollinearity generally improves the accuracy. This method is different from dimensionality reduction as it excludes and includes features present in the data without changing them often acting as a filter muting the irrelevant features. Ultimately, it helps in making the model simpler by reducing the complexity and sometimes also increases the accuracy of the model. Not every time a dataset needs feature selection techniques, but it is a useful practice to compare relationships between the features ultimately, removing multicollinearity if there is any present.

Chapter 3
Methodology

3.1 Introduction

In this chapter, we will present the classifiers that will be used in our study and later in chapter we describe the evaluation metrics for inferencing on testing dataset.

3.2 Classifiers

Training a model that predicts the survival status at a given time, means forecasting the odds of outcome instead of forecasting the point estimate of the occurrence. In our case there are two disease outcomes i.e. Alive and death, if the result of odds are greater than 50% then the predicted class is assigned value 1 otherwise it is 0.

3.2.1 Logistic Regression

Logistic regression models are statistical models describing relationship between a qualitative dependent variable and an independent variable. They are often used to study the effects of predictor variables and gives probability as an

outcome for each observation which further depending upon the threshold can further categorize survival status into two outcomes i.e., 0 (Alive) or 1 (Death). In such cases the model is called a binary logistic model. Apart from getting a binary outcome, binary logistic model can be used to answer different questions such as probability of getting lung cancer with the change in additional pounds a person gains, or every pack of cigarettes smoked per day. It can also provide answers based on the relationships between features such as how smoking, age and previous histopathology of patient can have an influence on the probability of having a lung cancer.

Logistic Regression is a kind of classification algorithm which finds relationship between the features and probability of event status i.e., death or alive using logit function.

The sigmoid function can be written as:

$$p = \frac{1}{1 + e^{-y}}$$

where p is the value of probability

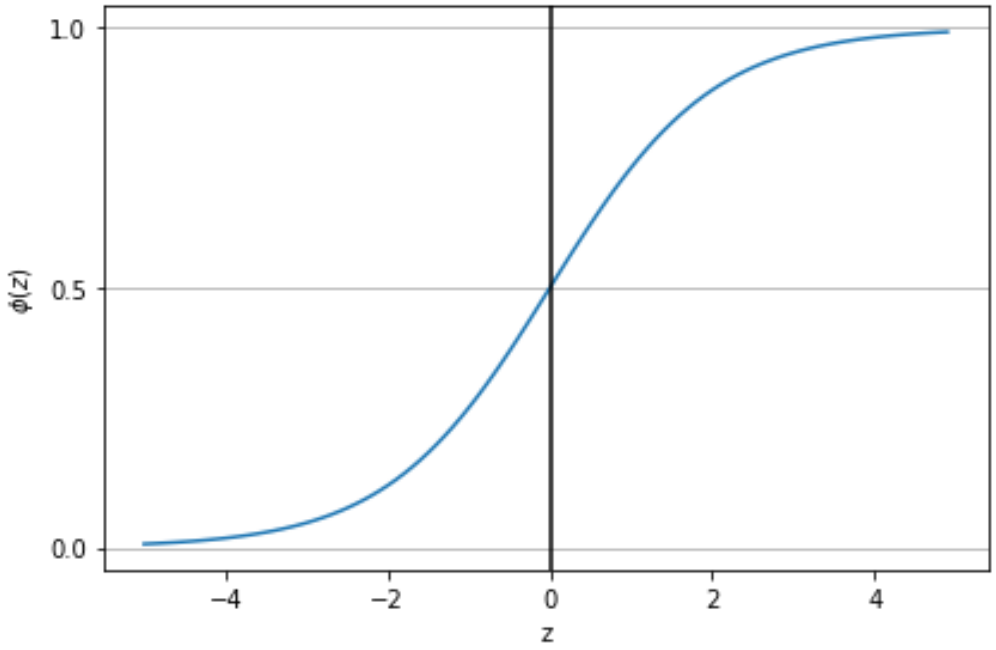


Figure 2: Sigmoid function

Sigmoid function is commonly used in machine learning as it maps an arbitrary real value to (0,1) range which means it can be interpreted as probability. For large values of z (nonlinearly dependent on a particular patient data) the probability (p) goes to one which can be interpreted as a patient being dead at a given time. Similarly, if value of argument z is small the probability of survival becomes 0 i.e., patient is alive at a given time.

3.2.2 Decision Trees

A Decision tree is in the form of flowchart like tree structure where node on the top is called the root node, the middle nodes are the internal nodes, and the bottom nodes are called leaf nodes. The lines connecting these nodes are called branches.

In a decision tree, each internal node splits the instance space into two or more subsets according to a discrete function of the input attributes values. This process is repeated on each derived subspace in a recursive manner called recursive partitioning. The recursion take place until, splitting no longer adds value to the predictions as seen in Figure 3. The main goal of the decision tree system is to decrease class entropy or increase the information gain at every child node creation. Given below is the equation of entropy E for a dataset with k classes and Information gain *IG(S,t)* of an attribute t .

$$E(S) = e = -\sum_{i=1}^{k} p_i log_2(p_i)$$

where p_i is the probability of randomly picking an element of a class

$$IG(S,t) = E(S) - \sum_{i \in Values(t)} \frac{|S_i|}{|S|} CE(s_i)$$

The Gini index is a measure of inequality in the sample. Its value ranges from 0 to 0.5. A value of 0 indicates samples are homogeneous whereas 0.5 reflects maximal inequality among elements.

$$Gini\ Index = 1 - \sum_{i=1}^{n} p_i^2$$

Using decision trees for predictive modelling of biomedical dataset increases interpretation and can be used to explain the underlying decision process.

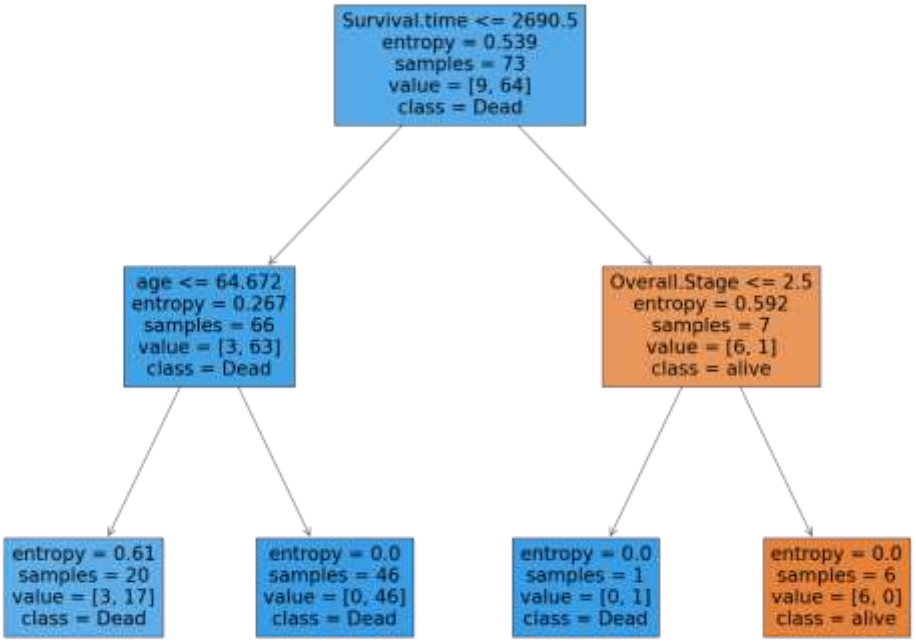


Figure 3: Decision tree classifying the disease outcome

3.2.3 Gradient Boosting

Boosting is defined as a strategy that involves combination of multiple simple models resulting in an overall stronger model. The simple models are called as weak learners. For example, the flow chart in Figure 4 below explains the gradient boosting method for N trees. Tree 1 is trained using a feature matrix X and target variable y. The predictions $y_1(hat)$ are used in evaluation of training set loss function r_1. Tree 2 is trained using the feature column matrix X and the loss function r_1 of Tree 1 as target variable. The predicted results $r_1(hat)$ are further used to evaluate loss function r_2. The entire process is repeated till all the N trees forming the ensemble are trained.

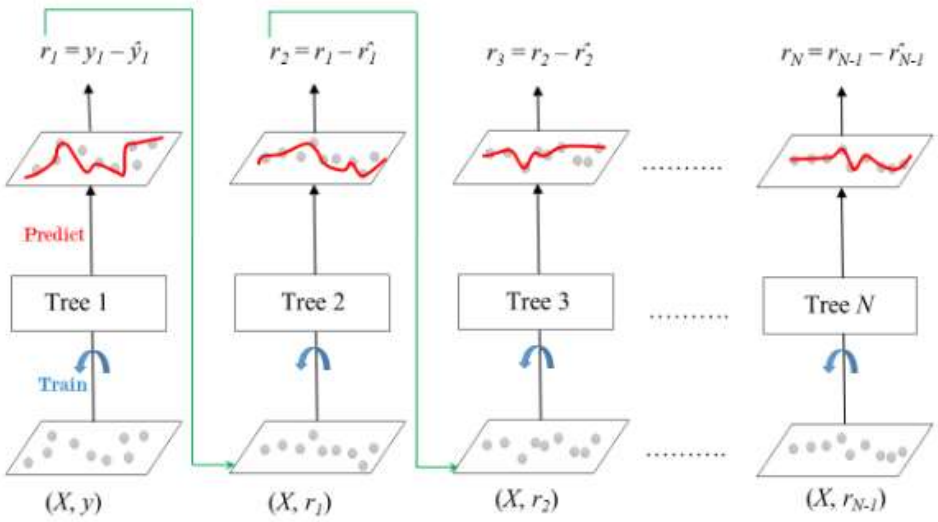


Figure 4: Flow chart for Gradient boosting

An initial model F_0 is defined for predicting the label y. The residual $(y - F_0)$ is related to this model. Another new model h_1 is fit to the residuals from the last step. F_0 and h_1 are added to give F_1

$$F_1(X) = F_0(X) + h_1(X)$$

To improve the performance of F_1, as seen in previous step we can model after the residuals of F_1 and create a new model F_2:

$$F_2(X) = F_1(X) + h_2(X)$$

This is done for *'m'* iterations, until the residuals are minimized:

$$F_m(X) = F_{m-1}(X) + h_m(X)$$

In other words, instead of fitting a model on the data at each iteration, it fits a new model to the residual errors made by the previous model.

Gradient boosting is a type of boosting where the objective is to minimize the loss function by adding weak learners using Gradient descent. It helps minimize any differentiable function.

At each node, a factor γ is multiplied with $h_m(X)$. This accounts for the difference in impact of each branch of the split. Gradient boosting helps in predicting the optimal gradient for the additive model, whereas for other gradient descent techniques aims at reducing error in the output at each run.

The following steps are involved in gradient boosting:

- $F_0(X)$ for initializing the boosting algorithm and can be defined as:

$$F_0(X) = argmin\,\gamma \sum_{i=1}^{n} L(y_i, \gamma)$$

- Computation of loss function is performed iteratively:

$$r_{im} = -\propto \left[\frac{\partial(L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(X)=F_{m-1}(X)}$$

where α is learning rate and $L(y_i, F(x_i)$ is loss function

- Each $h_m(X)$ is fit on the gradient obtained at each iteration
- The multiplicative factor γ_m for each terminal node is derived and the boosted version of model $F_m(x)$ is given as:

$$F_m(X) = F_{m-1}(X) + \gamma_m\, h_m(X)$$

3.2.4 XGBoost

XGBoost stands for extreme gradient boosting as it uses second-order Taylor expansion of the loss function to iterate and calculate weights ω at leaf nodes of the new tree K as seen in Figure 5 (Wanyue Zhao, 2020). Additionally, a regularization term is added to the loss function to control the complexity of the model and prevent it from overfitting. Therefore, XGBoost performs better in training efficiency, massive parallelism, and quadratic convergence (Wanyue Zhao, 2020).

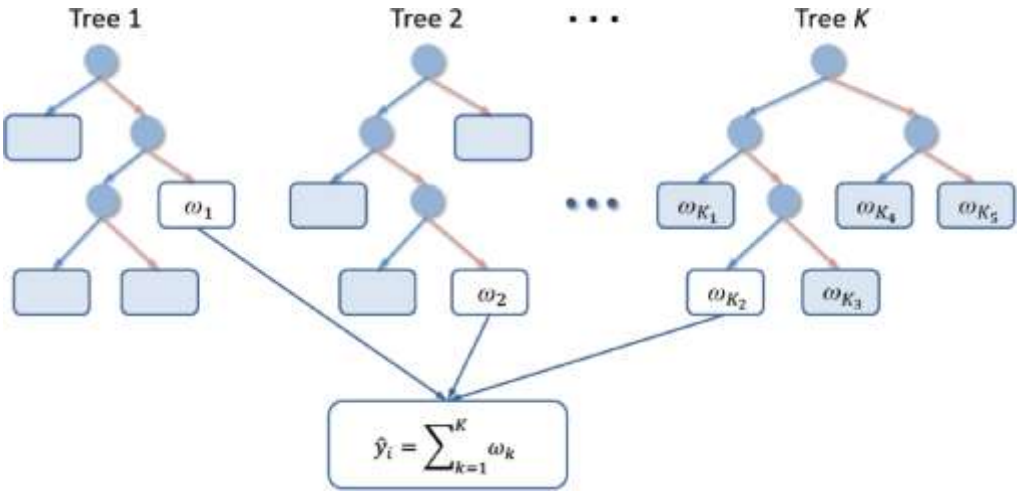


Figure 5: Iteration diagram for XGboost

It can perform well on imbalanced datasets as it calculates the second order gradients i.e., second partial derivatives of loss function ultimately giving more information about the direction of gradients and minimizes loss function.

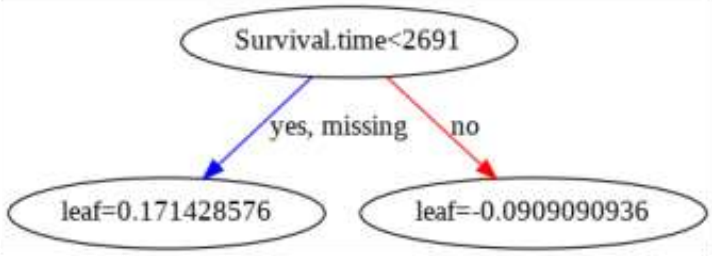


Figure 6: First decision tree

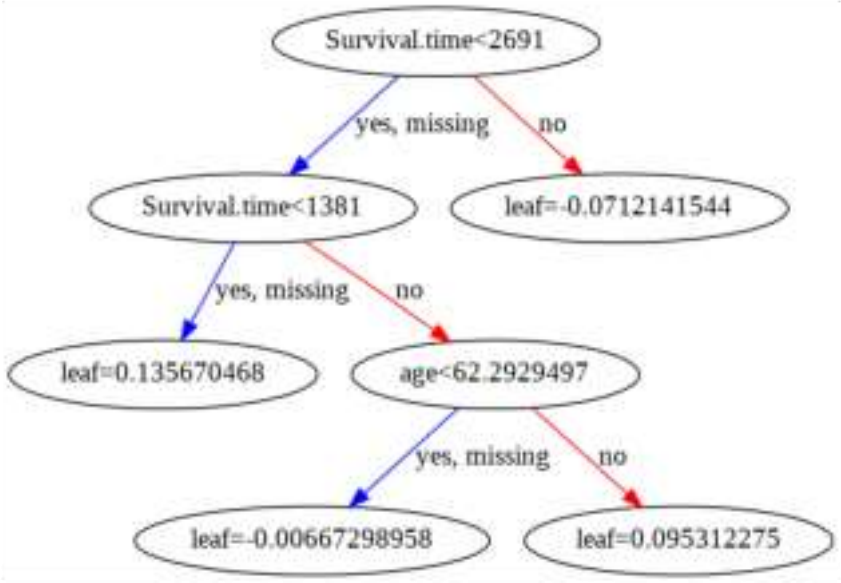


Figure 7: Fourth tree in the model

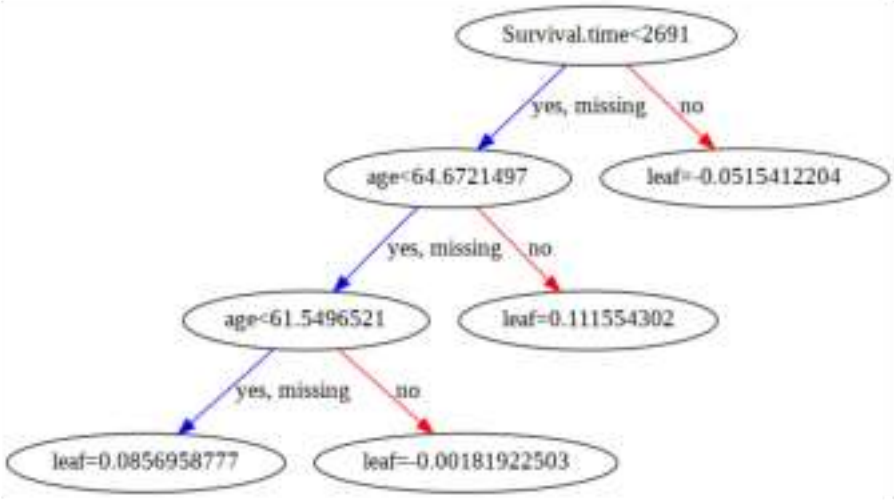


Figure 8: Tenth decision tree in the model

Figure 9: Twentieth decision tree in the model

We can see each decision tree is using different feature as the algorithm is trained. The first decision tree as seen in Figure 6 is making prediction solely on one feature i.e. survival time. However, as training progresses the model learns by from its predecessor to make accurate prediction thereafter using other features.

3.2.5 Balanced Bagging

Ensembles are the most efficient methods for improving the predictions of the minority class. Most of them use strategies from bagging and boosting. (Jerzy Blaszczynski, 2017). Bootstrap aggregation or bagging uses bootstrap sampling with replacement technique to estimate a population statistic from a subset sample by drawing multiple bootstrap samples. In a classification problem, predictions are made by taking the majority vote for the classes from all the predictions made by decision trees.

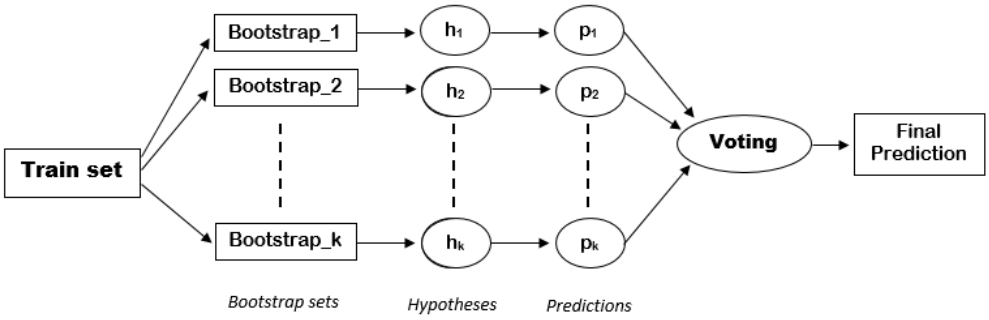


Figure 10: Iteration diagram for bagging

Each model h_1,h_2….h_k are made from different bootstrap sets. This ensemble method reduces the variance in the model without affecting bias.

$$F_{bag} = F_1 + F_2 + \cdots + F_M(X)$$

The term on the left side are predictions and on the right side are the individual learners.

3.2.6 Random Forests

Random forest is one of the most used ensembles learning method. It uses bagging and feature randomness when building each tree creating an uncorrelated forest of trees which makes decision by aggregating the votes from different trees. Due to random feature selection, the trees are more independent of each other as compared to regular bagging, which often results in better predictive performance.

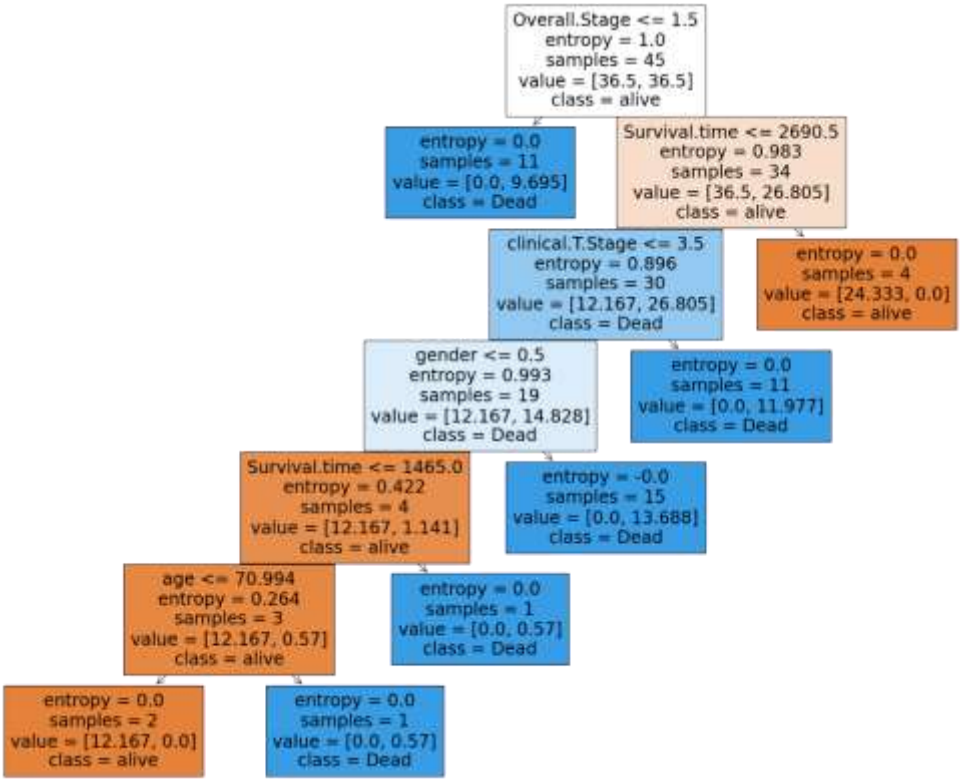


Figure 11: First decision tree in the random forest model

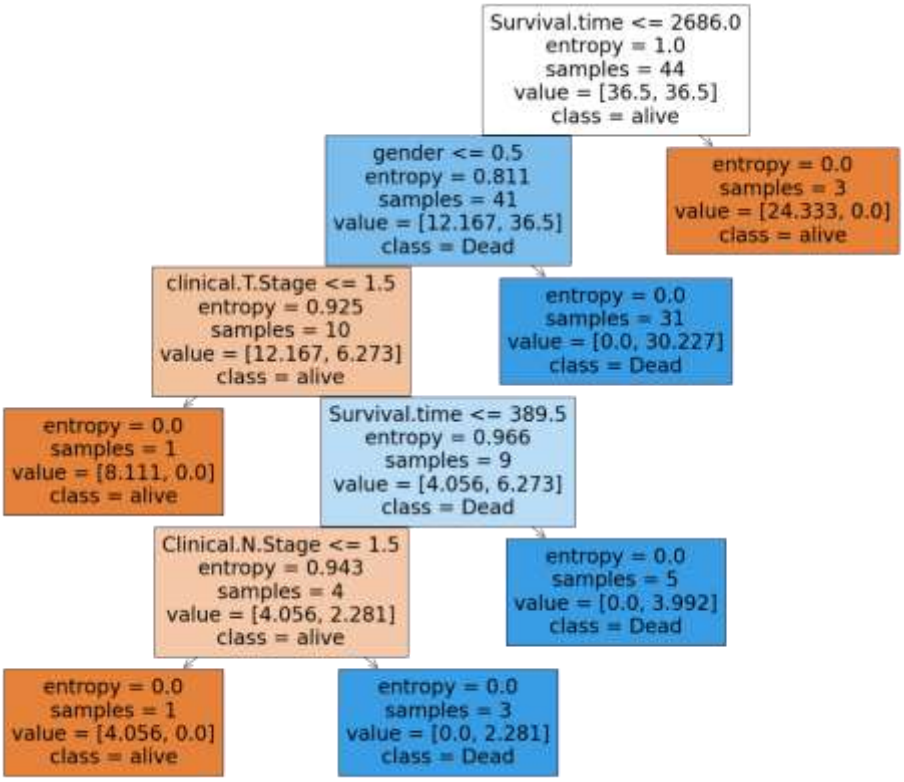


Figure 12: Fourth decision tree

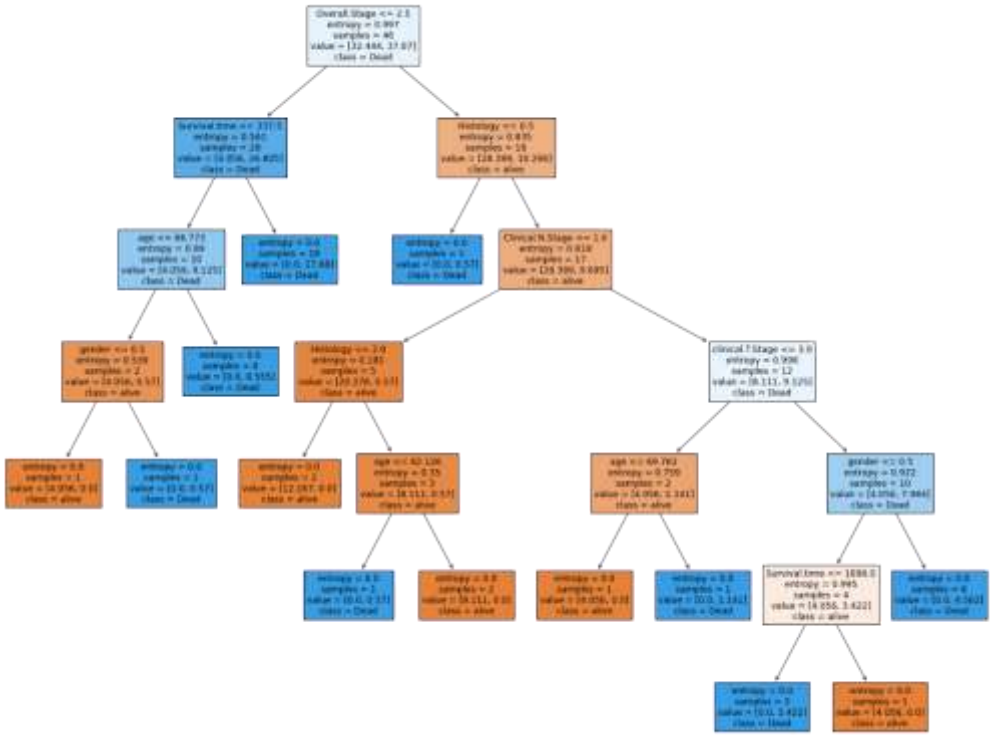


Figure 13: Tenth decision tree

3.3 Evaluation Metrics

After training classifiers on training data, our next step is to devise evaluation strategies to measure model performance on unseen dataset.

3.3.1 Confusion Matrix

It is a performance metric representing a tabular construct. It is a two-dimensional matrix, indexed in one dimension by the true class of survival status and the other by the class that the classifier assigns.

		Predicted	
		Death	Alive
Actual	Death	True Positive	False Negative
	Alive	False Positive	True Negative

Based on values in confusion matrix, we can calculate accuracy, recall, precision, specificity and misclassification rate for our model.

Accuracy: It determines the overall accuracy of the model.

$$Accuracy = \frac{True\ positive + True\ Negatives}{True\ Positive + True\ Negatives + False\ Positives + False\ negatives}$$

Recall: It indicates how many positive values, out of all the positive values, have been correctly predicted. It is also known as Sensitivity or Recall.

$$Recall = \frac{True\ positives}{True\ positives + False\ negatives}$$

Specificity: It indicates how many negative values, out of all the negative values, have been correctly predicted.

$$Specificity = \frac{True\ negatives}{True\ negatives + False\ positives}$$

Precision: It indicates how many values, out of all the predicted positive values are actually positive.

$$Precision = \frac{True\ positives}{True\ positives + False\ positives}$$

F-1 Score: F-1 score is the harmonic mean of precision and recall. It lies between 0 and 1. Higher score results in a better model.

$$F1\ Score = 2\left(\frac{(Precision * Recall)}{(Precision + Recall)}\right)$$

Often, for imbalance datasets such as Lung cancer, because of the nature of the disease majority of records belong to one class i.e. 1 (Death). In such datasets, accuracy can give false interpretations on classifier's performance. It recommended to look at metrics that provide a balanced view like specificity, recall and F-1 score.

3.3.2 Cumulative Gain charts

Cumulative gain charts and lift charts are metrics that evaluates model performance on portions of the total population as compared to a confusion matrix which determines the performance of the model on the entire population. In other words, we have a score for every percentage of population. For example, Figure 14 represents cumulative gain chart for a Logistic regression model that indicates ~78% of patients whose disease outcome at a given time is alive are captured in top 20% of data based on model i.e., we can identify and target ~78% of patients who will be alive if this model was used in clinical practice.

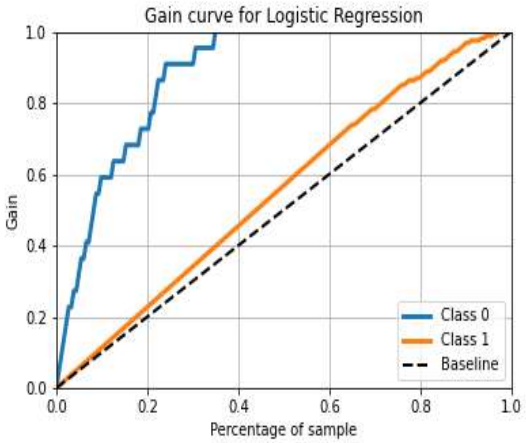


Figure 14: Cumulative gain chart for logistic regression model

3.3.3 Lift curves

Lift curve measures improvement a machine learning model brings in comparison to random predictions or in a case without a model. The improvement is called as lift. For example, figure 15 is the lift chart for logistic regression model indicating that we have a lift of ~3.8 for top two deciles, i.e., selecting top 20% of the records based on the model, we can get 3.8 times the total number of patients whose survival status is alive found by randomly selecting 20% without the model.

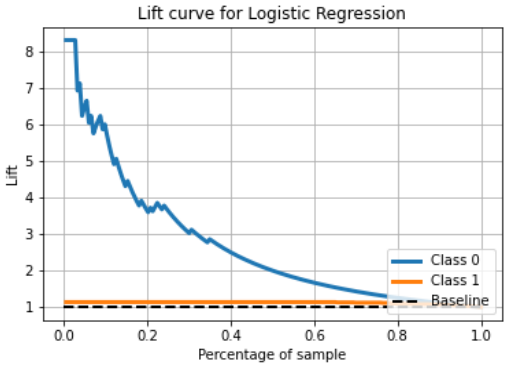


Figure 15: Lift curve for logistic regression model

3.3.4 K-S statistic charts

K-S or Kolmogorov-Smirnov chart measures the degree of separation between the positive and negative class. If K-S is 100, the patient population is partitioned into two separate groups in which one group contains all positives and the other has all the negatives. K-S will be 0, if the model cannot differentiate between both the groups. Figure 16 is K-S chart of logistic regression model indicating that by targeting top 57.4% patients we can capture ~95% of patients whose survival status is alive at a K-S value of 0.760.

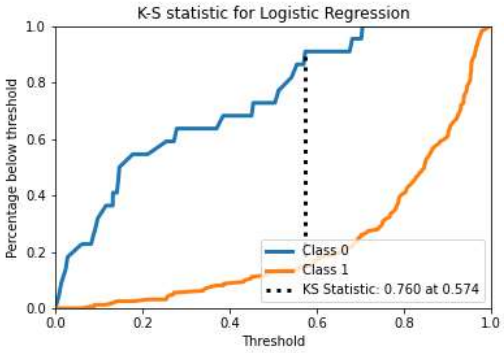


Figure 16: K-S statistic chart

3.3.5 Receiver Operator Characteristic (ROC)

ROC curves determine model's accuracy using Area Under Curve (AUC). The area under the curve (AUC), also referred to as index of accuracy (A) or concordant index, represents the performance of the ROC curve. Higher area results in better model. It is plotted between True Positive Rate on Y-axis and False Positive Rate on X-Axis. The dashed line represents the ROC curve at 0.5

threshold. At this point, sensitivity = specificity. As the curve gets higher, for example looking at ROC curve of Decision tree in the Figure 17, indicates model will have a higher accuracy.

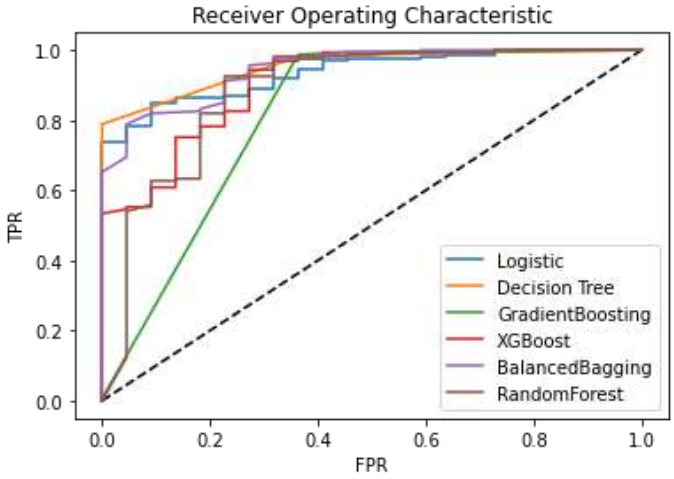


Figure 17: ROC curve

3.3.6 Precision – Recall Curve

A P-R curve is a graph with Precision values plotted on the y-axis and Recall values on the X-axis. Generally, a classifier with higher AUC on the ROC curve has higher AUC on the P-R curve. Figure 18 and Figure 17 are the P-R and ROC curve of the same testing set. We can see that AUC on ROC and P-R is higher for both Logistic regression and decision tree.

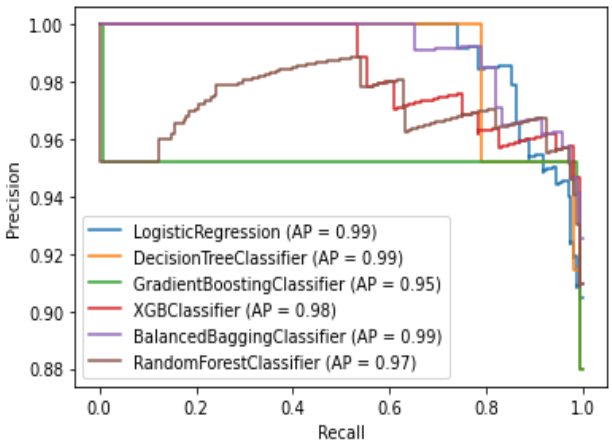


Figure 18: P-R curve

Chapter 4
Results and Discussions

In the previous section, the fundamental theory required to construct the model predicting disease outcome and the evaluation metrics to be used on testing set were presented. In this section, the results of models proposed in Chapter 3, will be shown. Furthermore, the results and their significance will also be discussed.

4.1 Dataset

The dataset used for evaluation of the proposed model is from MAASTRO Clinic, (Maastricht, The Netherlands). This dataset is open source and can be found at TCIA (The cancer imaging archive) under NSCLC (Aerts, 2019).

Four hundred and twenty-two consecutive patients were included (132 women and 290 men), with inoperable, histologic or cytologic conferred NSCLC, UICC stages I-IIIb, treated with radical radiotherapy alone (n = 196) or with chemo-radiation (n = 226). Mean age was 67.5 years (range: 33–91 years). The study

has been approved by the institutional review board. All research was carried out in accordance with Dutch law. The Institutional Review Board of the Maastricht University Medical Centre (MUMC+) waved review due to the retrospective nature of this study.

Out of 422 records, we have only 365 patients with all the information. The survival time (in days) in the dataset is from the start of the treatment and there is a possibility that the status of patient recorded may not be accurate i.e. the clinicians may not have received the information right when the event outcome occurred.

We used a combination of machine learning models, out of which some are prone to class imbalance and some are resistant to class imbalance in the dataset. The models we have used for our evaluation are decision trees, gradient boosting, balanced bagging, logistic regression and XGBoost. The hyperparameters were tuned with grid search cross validation. We also performed analysis on the effect of oversampling and undersampling on the dataset to overcome class imbalance and measure model performance. Furthermore, we experimented with different training and testing splits for optimal model performance and N-1 testing to measure metrics for both the classes. At last, we also perform unsupervised learning on the dataset. All of these approaches are aimed at recognizing optimal predictive modelling techniques for an imbalanced dataset, ensuring accurate predictions for both the disease outcomes.

4.2 Performance of model on different training testing splits

4.2.1 Train test split ratio of 50:50

As per the evaluation metrics defined in Chapter 3, Table 1 shows the results of metrics for all the models followed by Gain charts, Lift curves and K-S statistic charts.

Methods	Accuracy	Precision	Recall	F1score	Specificity	Mis-classification
Logistic regression	86%	73%	88%	79%	73%	14%
Decision Tree	94%	64%	98%	77%	60%	6%
Gradient Boosting	95%	64%	99%	77%	64%	5%
XGBoost	93%	50%	99%	67%	50%	7%
Balanced Bagging	90%	77%	91%	84%	77%	10%
Random Forests	91%	27%	99%	43%	27%	9%

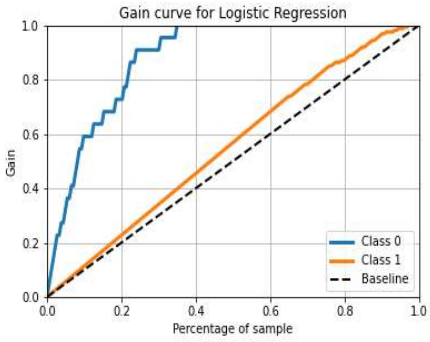
Table 1: Metric results for train test split ratio of 50:50

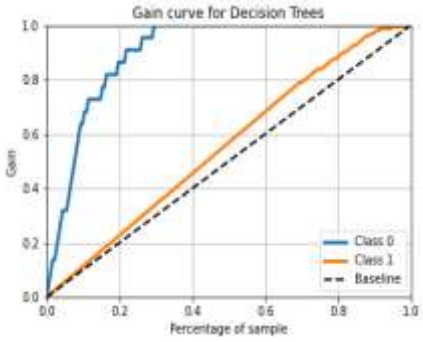
4.2.1.1 Dependence on training dataset

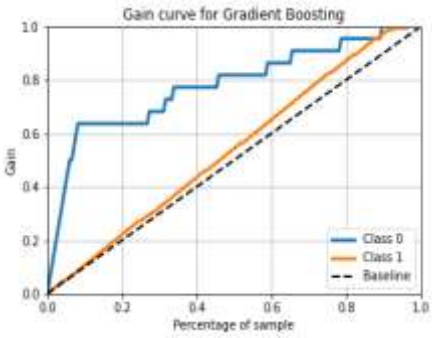
It is common for machine learning algorithms to give different results on each run. In our dataset, due to the very nature of the disease most of the patient population belongs to one class i.e., Death because of which the overall accuracy is mostly the contribution of the majority class and due to dependence of model on training dataset the specificity of the model has more variance due to limited samples of minority class in the dataset. The average specificity of each model and the variance is in Table 2.

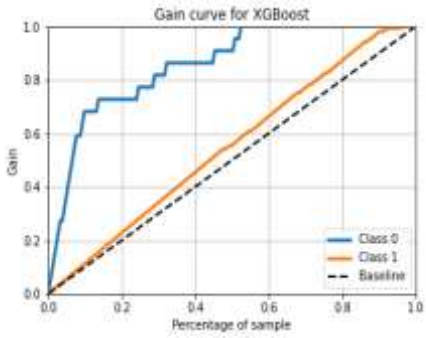
Methods	**Average Specificity**	**Variance**
Logistic regression	77%	1.2%
Decision Tree	49.2%	1.8%
Gradient Boosting	65.5%	0.7%
XGBoost	56.9%	0.5%
Balanced Bagging	76.9%	1.5%
Random Forests	53.8%	1.6%

Table 2: Average specificity and variance for all models









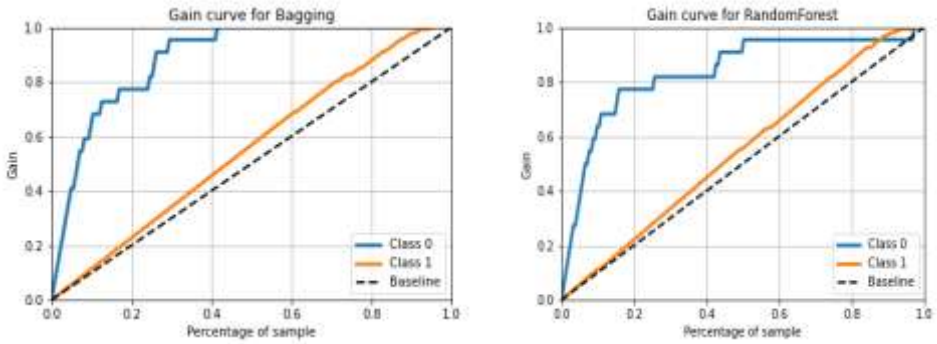


Figure 19: Cumulative gain charts for all the models

In all the gain charts for 50:50 split, Decision trees gives the highest gain for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models. However, there is a high variance for Decision trees and cannot be deemed as a robust model for both the classes. Logistic Regression can be a good choice as it has high gain, average specificity is greater than 77% and variance is 1.2% across ten runs of the model.

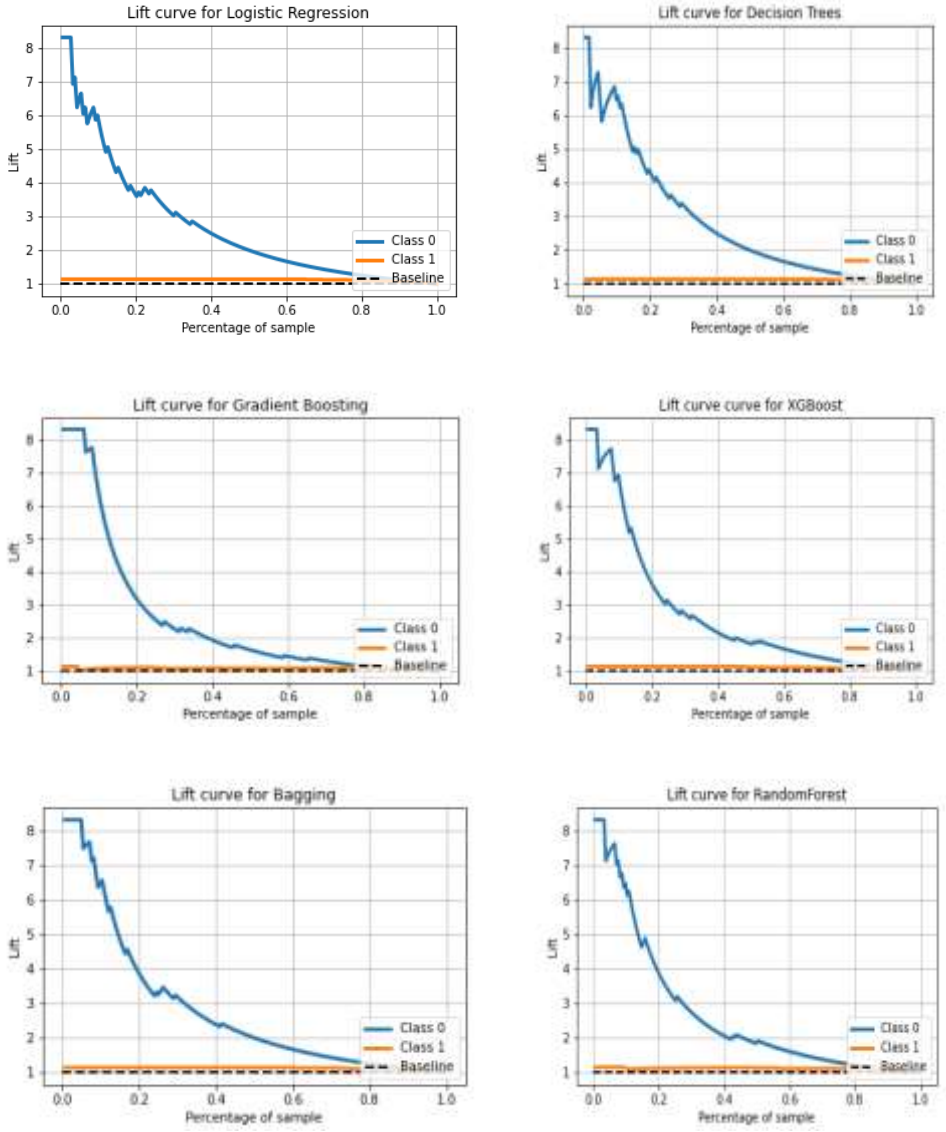


Figure 20: Lift charts for all the models

In all the lift charts for 50:50 split, Decision trees give the highest lift ~4.2 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models. However, as discussed previously they suffer from low stability.

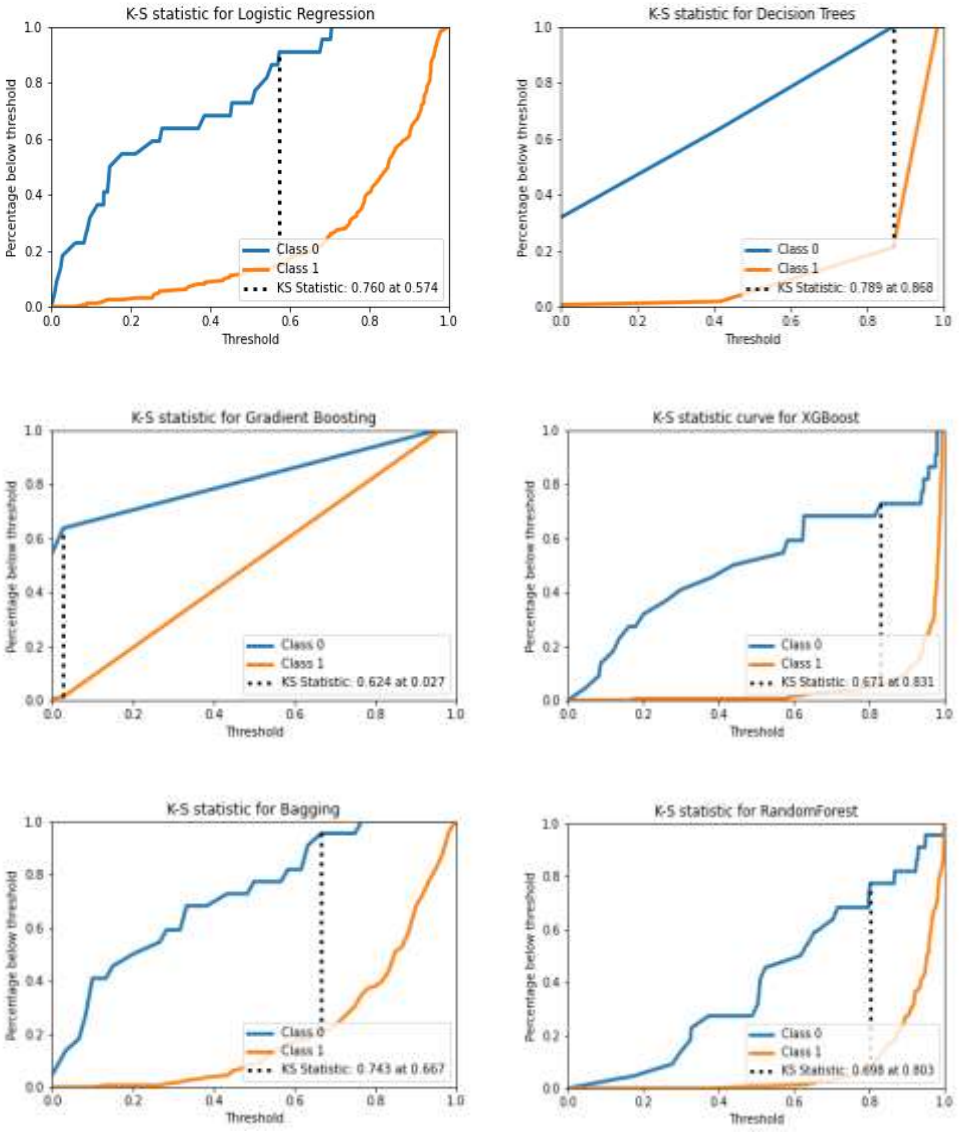


Figure 21: K-S statistic charts for all the models

In all the K-S charts for 50:50 split, Decision trees give the maximum separation between both the classes at K-S statistic value of 0.789. However, similar value of K-S statistic 0.76 can be achieved at lesser percentage of sample i.e., at top 57.4% of patient population in Logistic Regression as compared to top 86.8% in Decision trees.

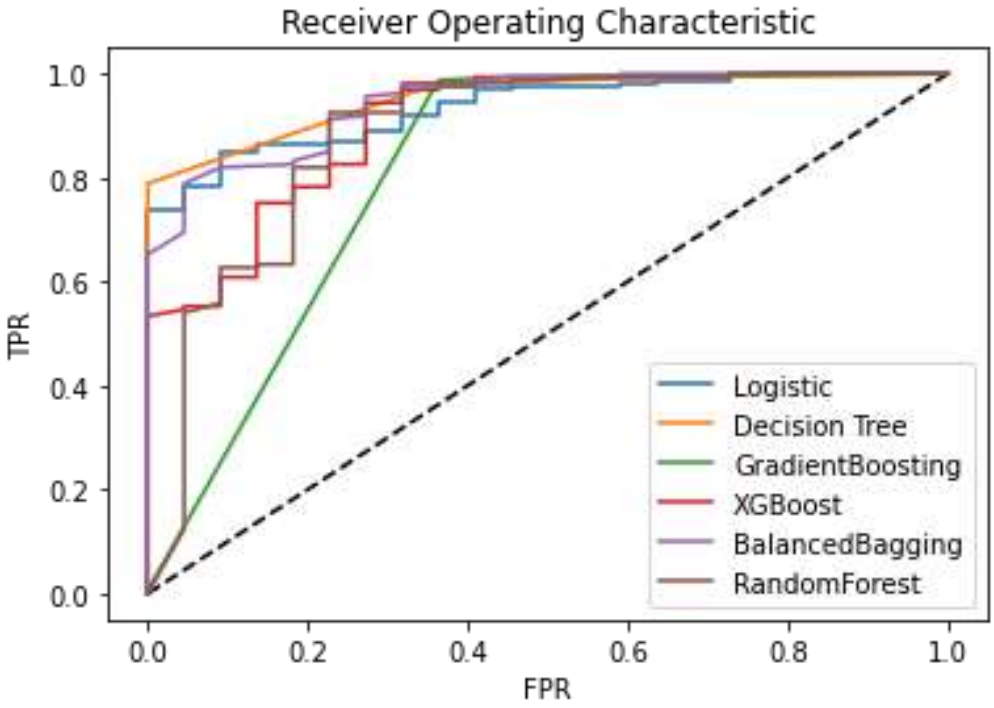


Figure 22: ROC curve

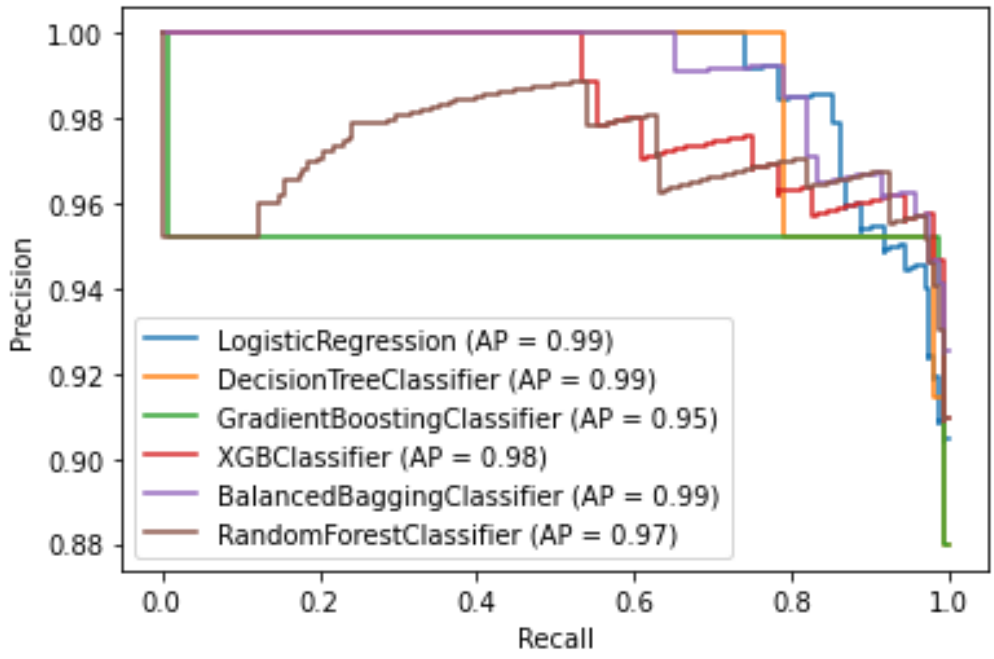


Figure 23: PR curve

4.2.2 Train test split ratio 60:40

As per the evaluation metrics defined in Chapter 3, Table 3 shows the results of metrics for all the models followed by Gain charts , Lift curves and K-S statistic charts.

Methods	Accuracy	Precision	Recall	F1score	Specificity	Mis-classification
Logistic regression	84%	78%	84%	81%	78%	16%
Decision Tree	95%	67%	98%	79%	67%	5%
Gradient Boosting	95%	61%	99%	76%	61%	5%
XGBoost	93%	50%	99%	66%	50%	7%
Balanced Bagging	88%	89%	88%	88%	89%	12%
Random Forests	92%	39%	100%	56%	39%	8%

Table 3: Metric results for train test split ratio of 60:40

4.2.2.1. Dependence on training dataset

The average specificity of each model and the variance for a test size of 40% can be seen in Table 4. This analysis is done only for 50:50 and 60:40 training testing data split as in further splits the number of samples increases in the training dataset and it becomes difficult to assess on a small test set.

Methods	Average Specificity	Variance
Logistic regression	79.9%	0.8%
Decision Tree	58.4%	1.2%
Gradient Boosting	54.5%	1.3%

XGBoost	55%	0.5%
Balanced Bagging	82%	0.8%
Random Forests	58.4%	1.3%

Table 4: Average specificity and variance for all models

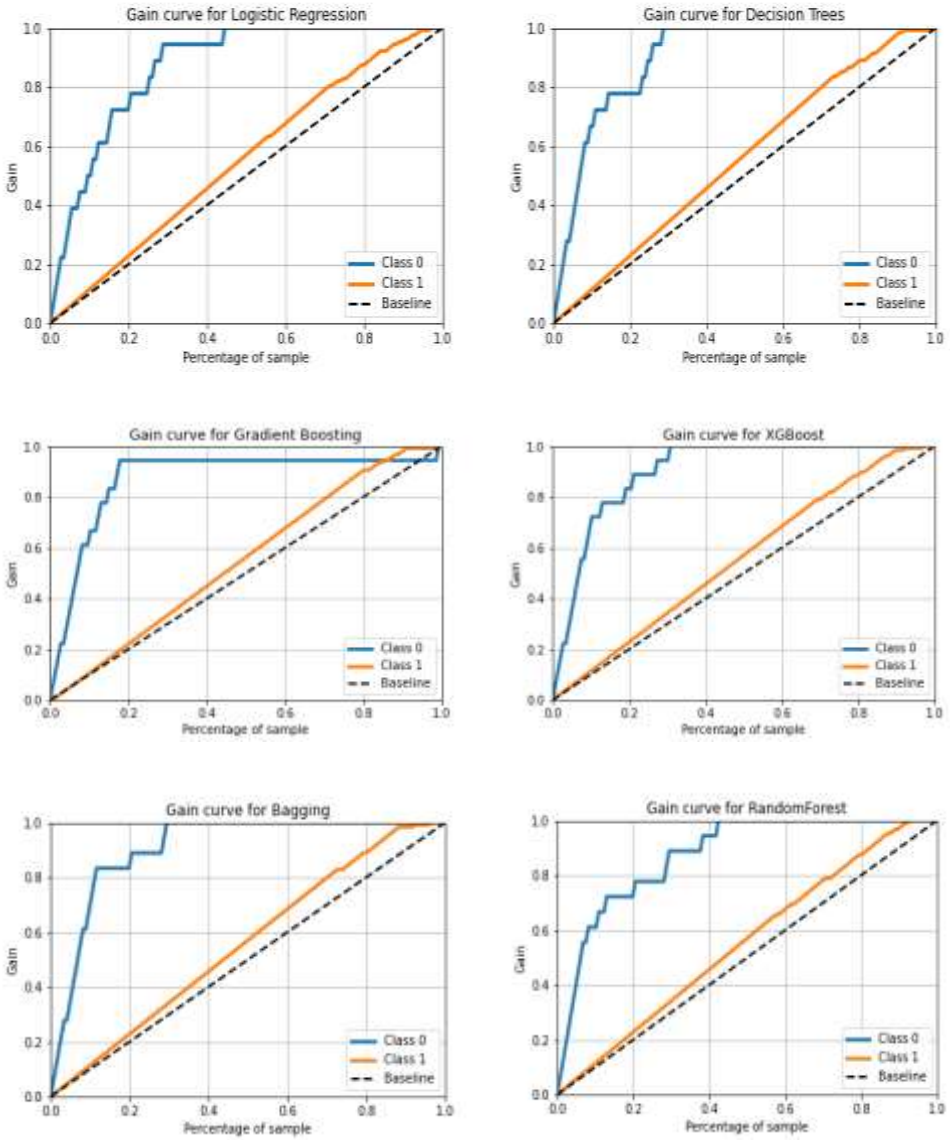


Figure 24: Gain curves for 60:40 training-testing data split for all the models

In all the gain charts for 60:40 split, Gradient Boosting gives the highest gain for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models. However, it may not be the best balanced model given the low F-1 score, recall and specificity as compared to other models. On the other hand, Balanced bagging is an appropriate choice with low variance and performs well for both the classes.

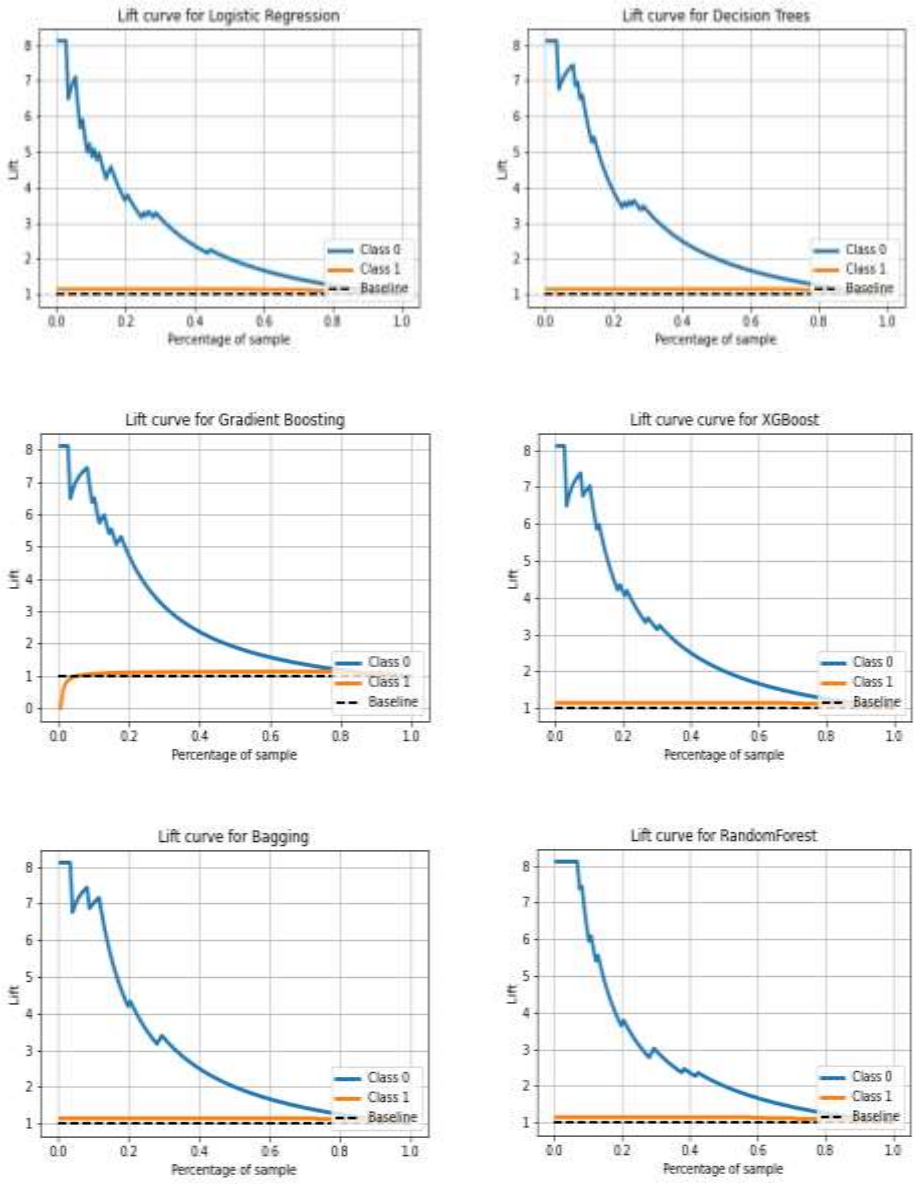


Figure 25: Lift curves for 60:40 training-testing data split for all the models

In all the lift charts for 60:40 split, Gradient Boosting gives the highest lift ~4.5 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models.

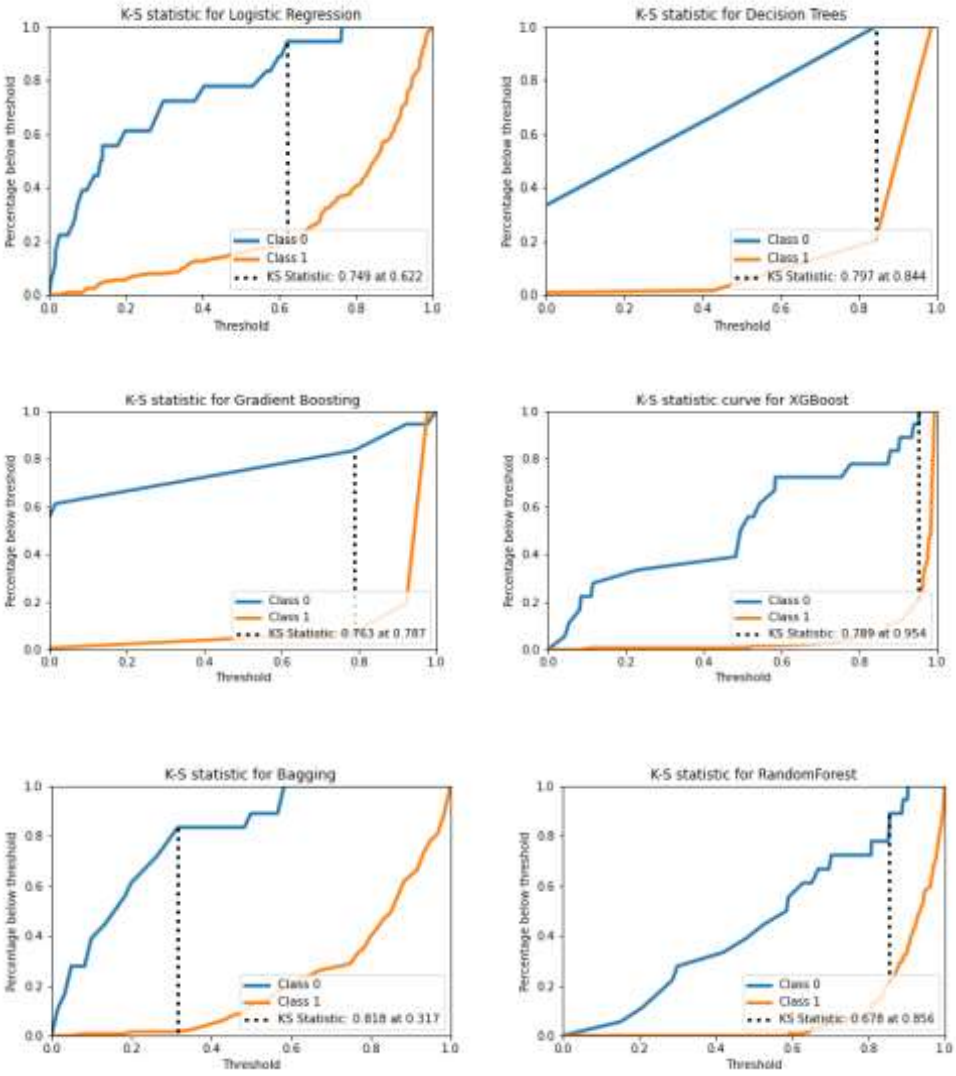


Figure 26: K-S statistic curves for 60:40 training-testing data split for all the models

In all the K-S charts for 60:40 split, Bagging give the maximum separation between both the classes at K-S statistic value of 0.818 which explains a high F1-score of 0.81 ensuring accurate predictions for both the classes.

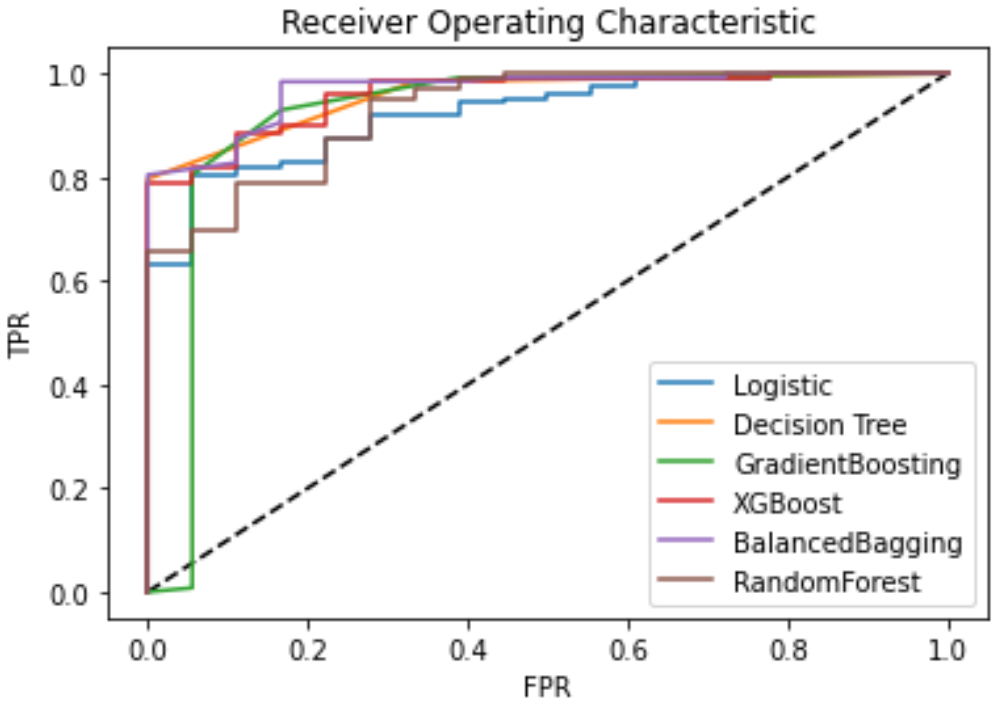


Figure 27: ROC curve

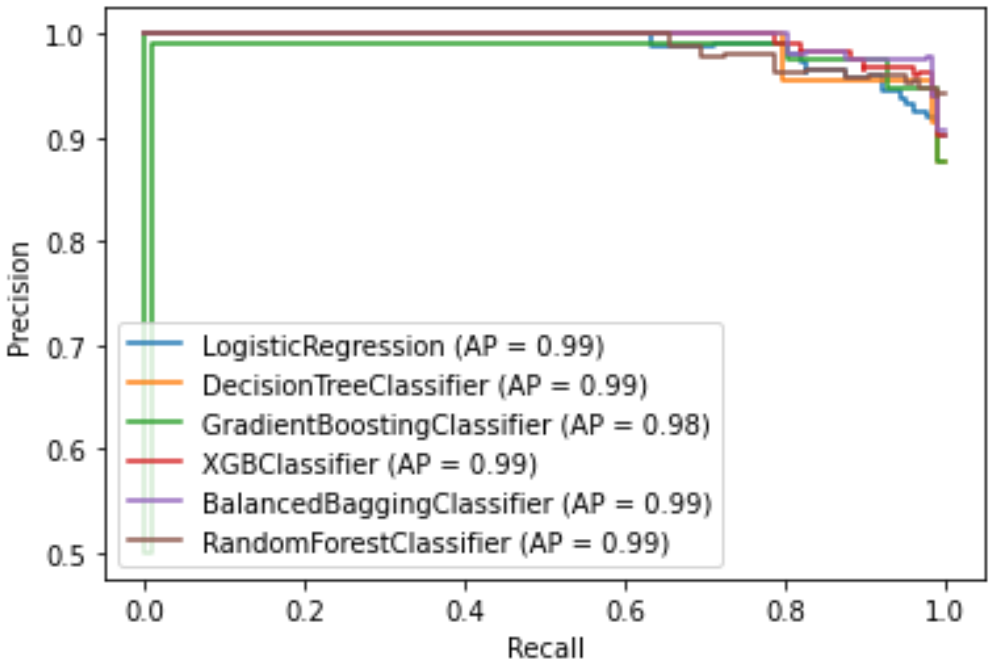


Figure 28: PR curve

4.2.3 Train test split ratio 80:20

As per the evaluation metrics defined in Chapter 3, Table 5 shows the results of metrics for all the models followed by Gain charts, Lift curves and K-S statistic charts.

Methods	Accuracy	Precision	Recall	F1score	Specificity	Mis-classification
Logistic regression	84%	89%	83%	86%	89%	16%
Decision Tree	97%	78%	100%	88%	78%	3%
Gradient Boosting	95%	56%	100%	71%	56%	5%
XGBoost	99%	89%	100%	94%	89%	1%
Balanced Bagging	86%	89%	86%	87%	89%	12%
Random Forests	96%	67%	100%	80%	67%	4%

Table 5: Metric results for train test split ratio of 80:20

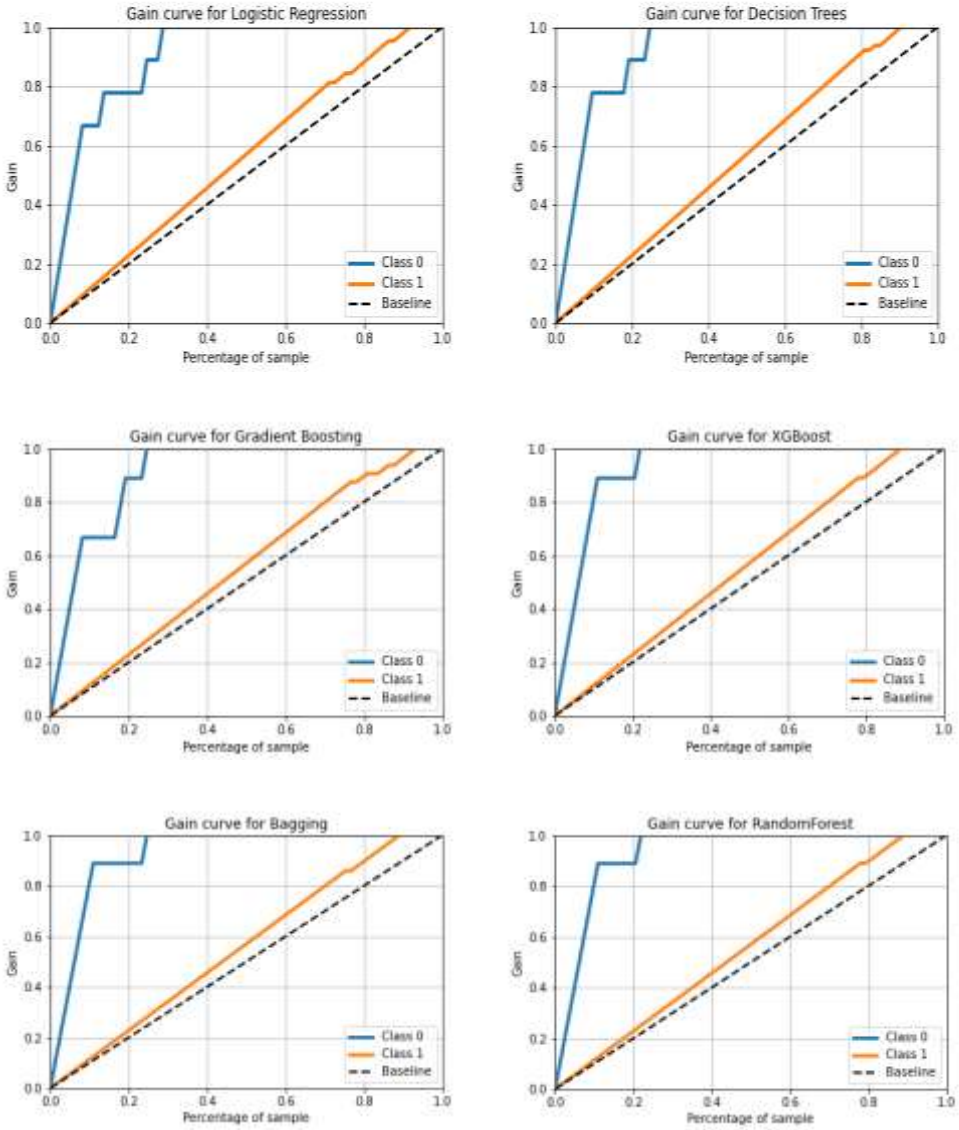


Figure 29: Gain curves for 80:20 training-testing data split for all the models

In all the gain charts for 80:20 split, all the models except logistic regression gave a gain of greater than 80% for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set. XGBoost performs well on all evaluation metrics with just 1% of

misclassification, high F-1 score and specificity making it a robust model towards both the classes.

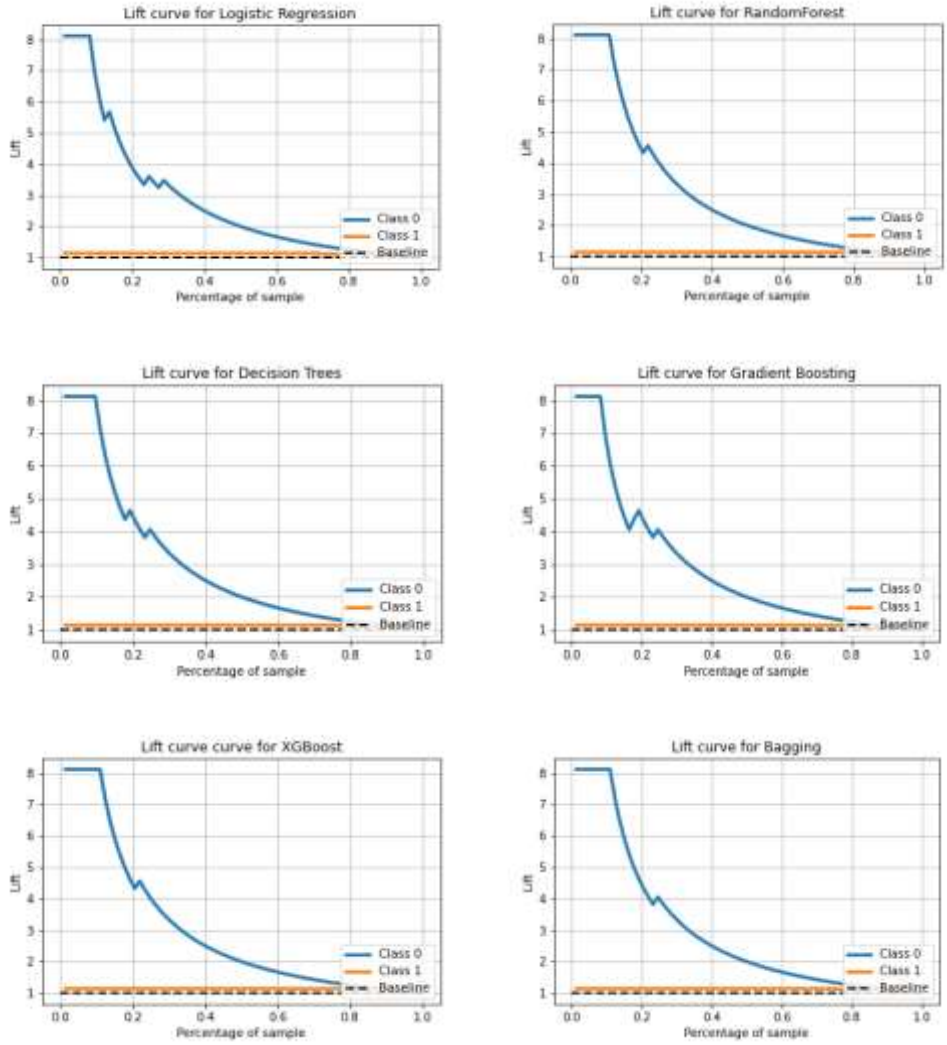


Figure 30: Lift curves for 80:20 training-testing data split for all the models

In all the lift charts for 80:20 split, Gradient Boosting gives the highest lift ~4.5 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models.

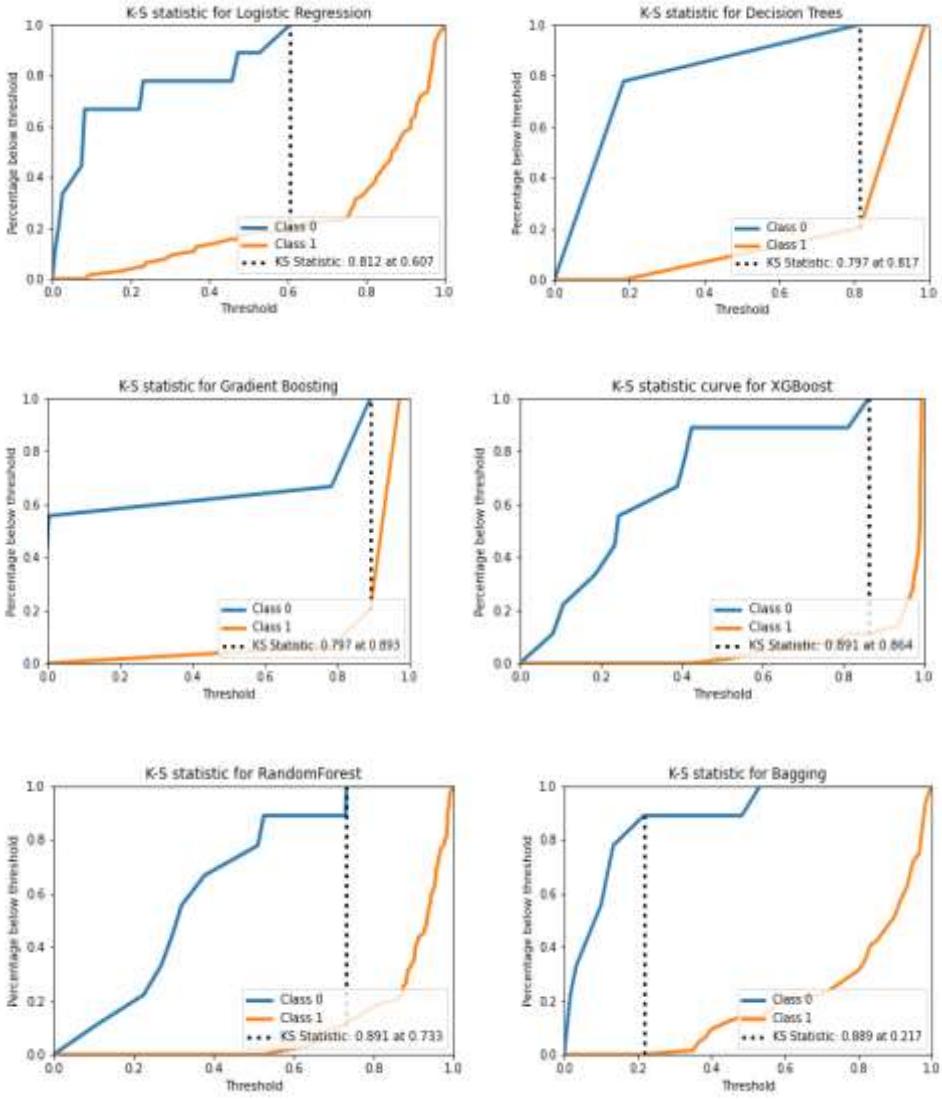


Figure 31: K-S statistic for 80:20 training-testing data split for all the models

In all K-S charts for 80:20 split, XGBoost and Random forest give the maximum separation between both the classes at K-S statistic value of 0.891 followed by Bagging. However, as discussed previously XGBoost is a more robust classifier.

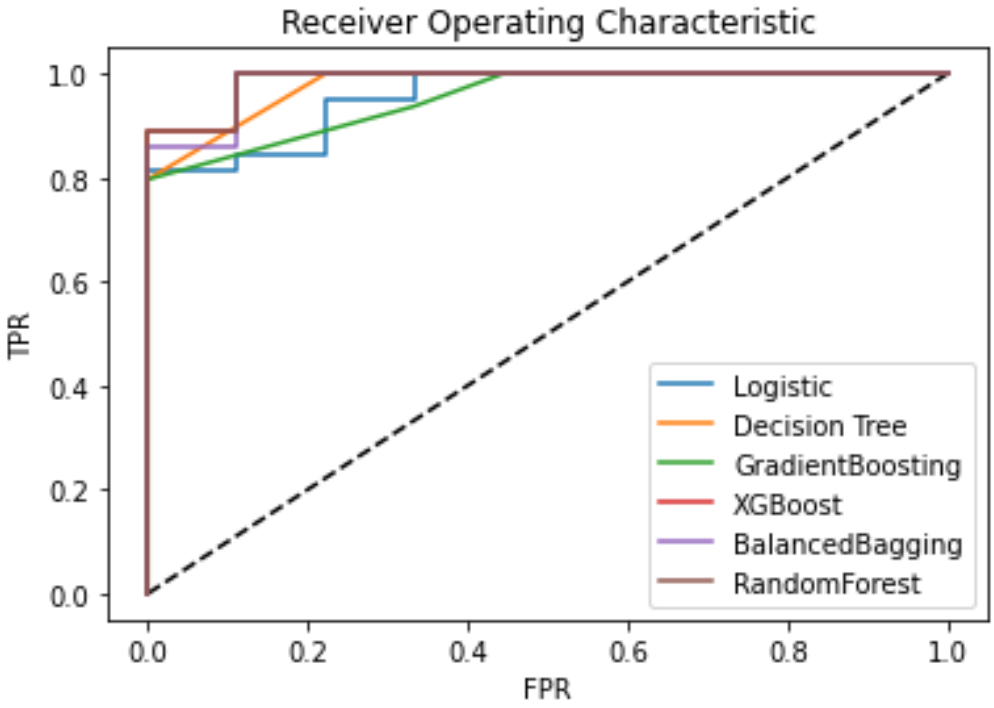


Figure 32: ROC-AUC

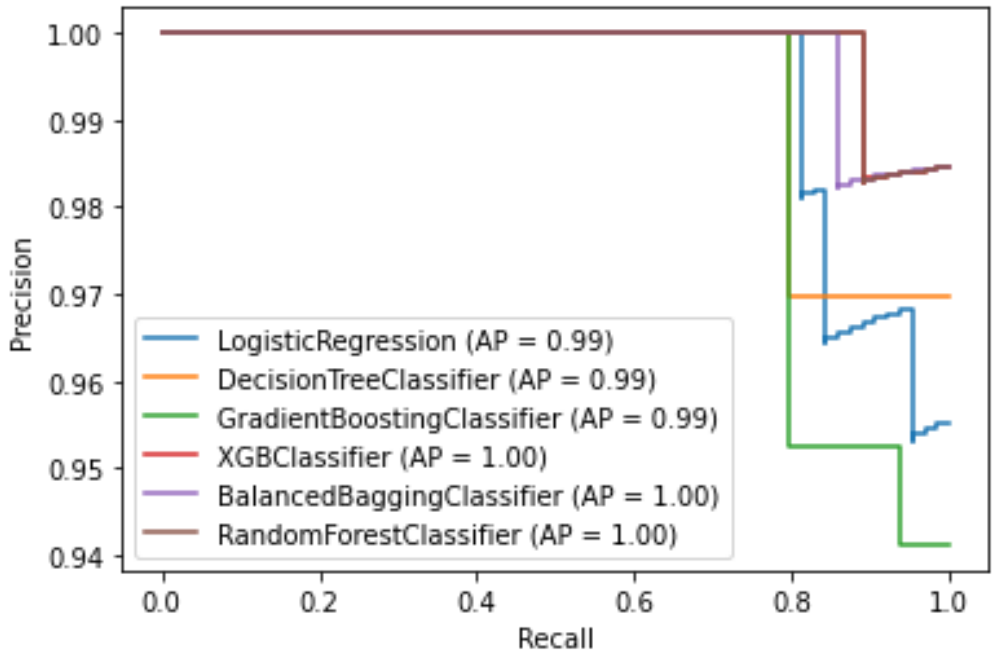


Figure 33: PR curve

4.2.4 Train test split ratio 90:10

As per the evaluation metrics defined in Chapter 3, Table 6 shows the results of metrics for all the models followed by Gain charts , Lift curves and K-S statistic charts.

Methods	Accuracy	Precision	Recall	F1score	Specificity	Mis-classification
Logistic regression	86%	75%	88%	81%	75%	14%
Decision Tree	92%	25%	100%	40%	25%	8%
Gradient Boosting	92%	25%	100%	40%	25%	8%
XGBoost	97%	75%	100%	86%	75%	3%
Balanced Bagging	89%	75%	91%	82%	75%	11%
Random Forests	92%	25%	100%	40%	25%	8%

Table 6: Metric results for train test split ratio of 90:10

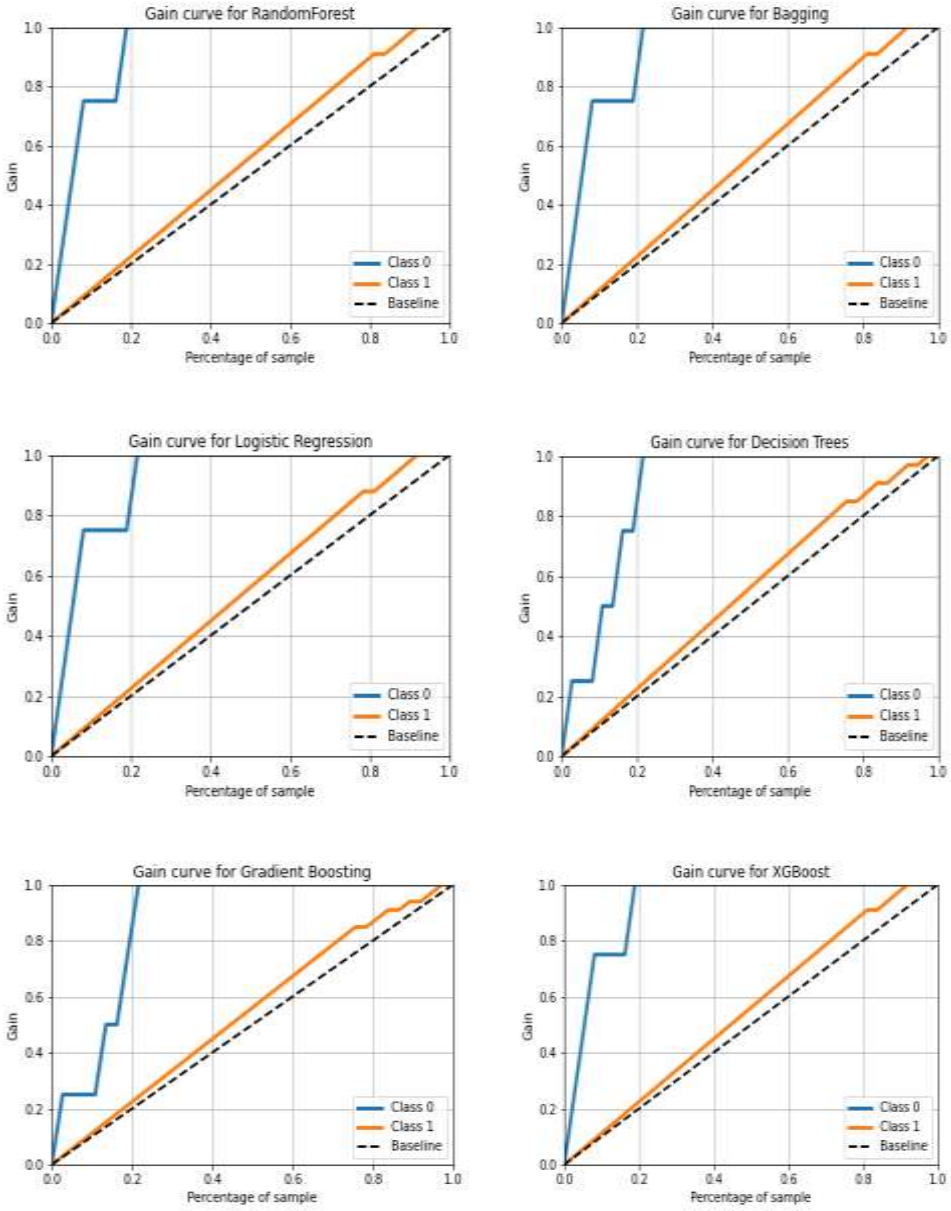


Figure 34: Gain curves for 90:10 training-testing data split for all the models

In all the gain charts for 90:10 split, all the models except logistic regression and Balanced bagging give a gain of greater than 95% for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set. Decision trees and Gradient Boosting are

not an appropriate choice to be used in clinical practice due to low precision, F-1 score and specificity, ultimately making it a less robust choice.

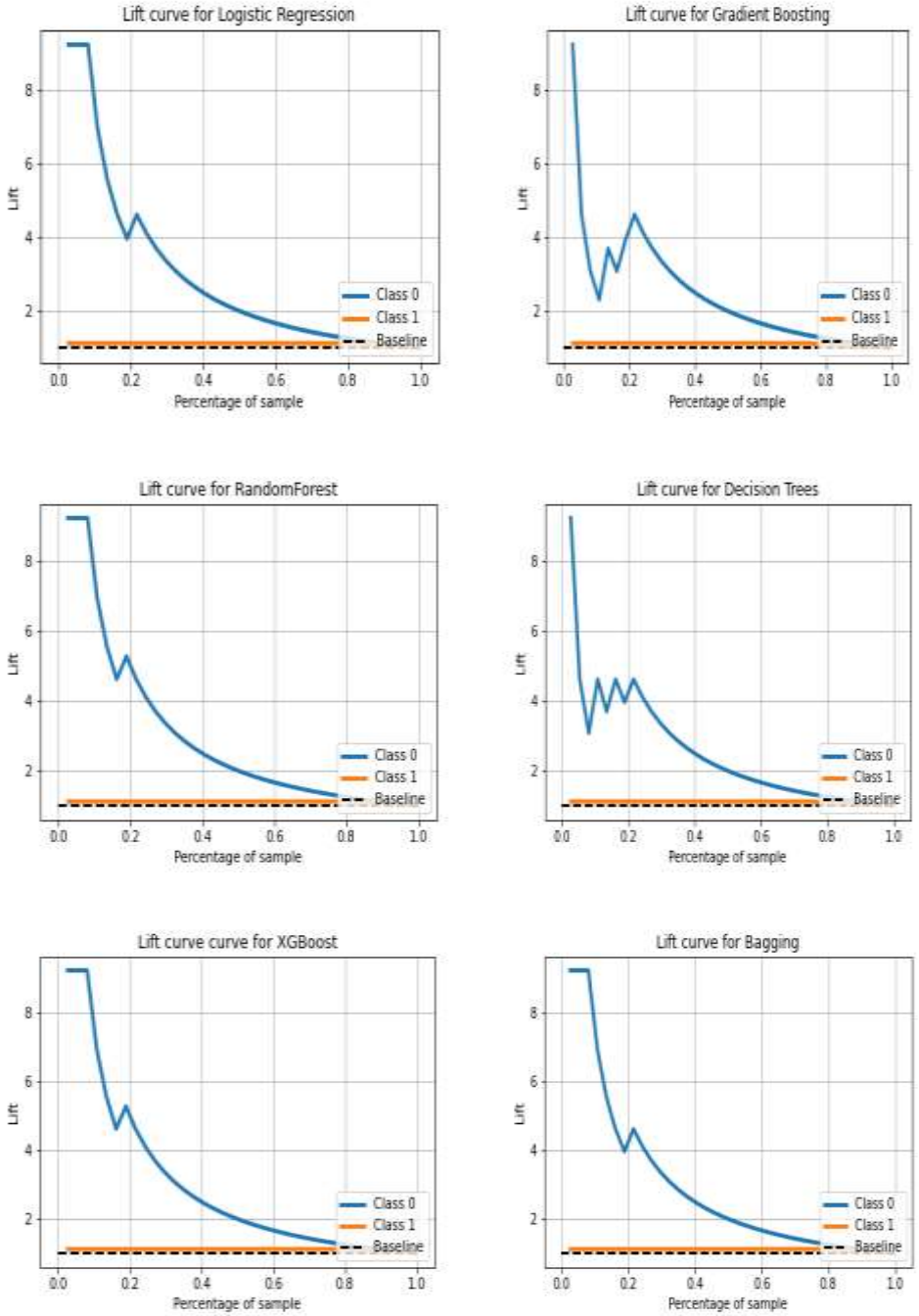


Figure 35: Lift curves for 90:10 training-testing data split for all the models

In all the lift charts for 90:10 split, Random Forest and XGBoost gave the highest lift greater than 5 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models. However, XGBoost performs well on all evaluation metrics specially high F-1 score, precision and specificity making it more robust as compared to Random forest with lowest misclassification percentage.

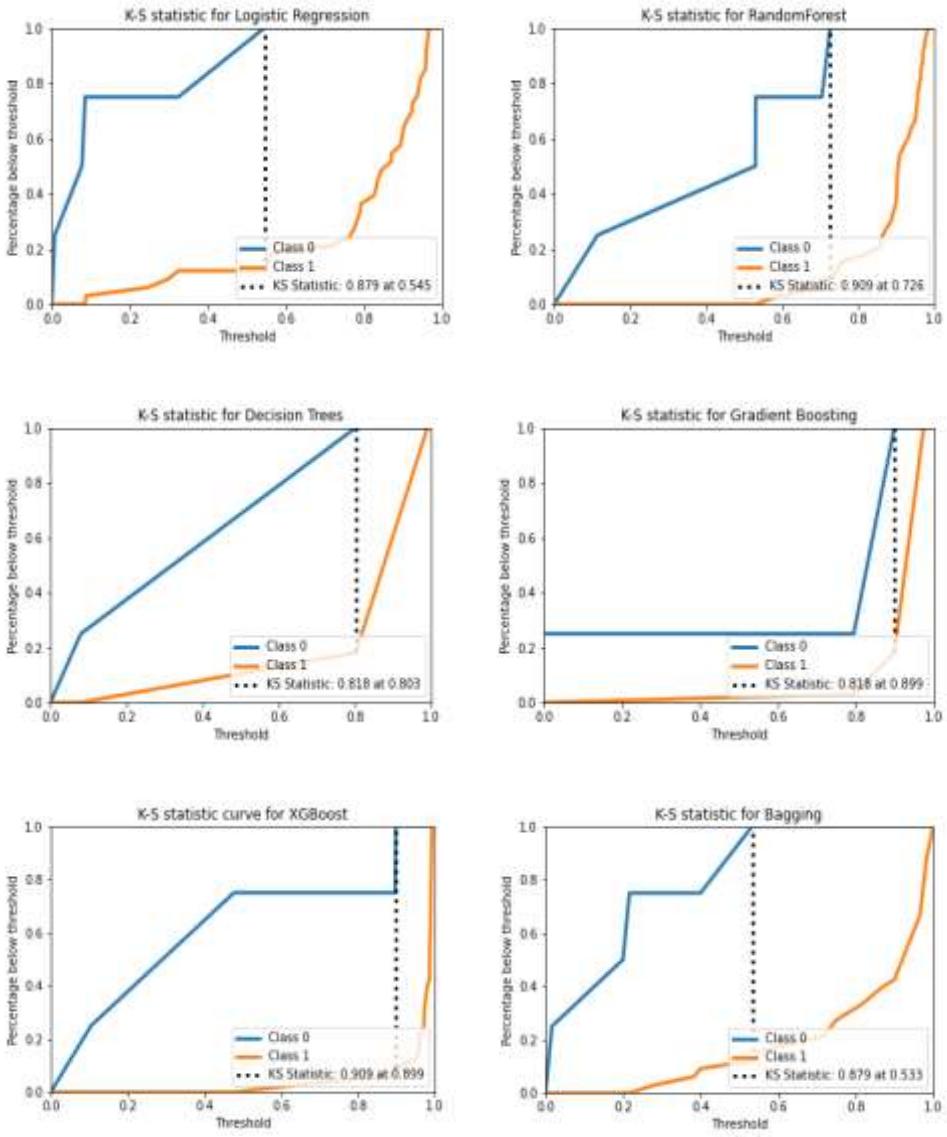


Figure 36: Gain curves for 90:10 training-testing data split for all the models

In all the K-S charts for 90:10 split, XGBoost and Random forest give the maximum separation between both the classes at K-S statistic value of 0.909 followed by Bagging which is highest across all the split ratios. This can also be due to small testing set size and more examples from the minority class provided during the training improved the performance of the model.

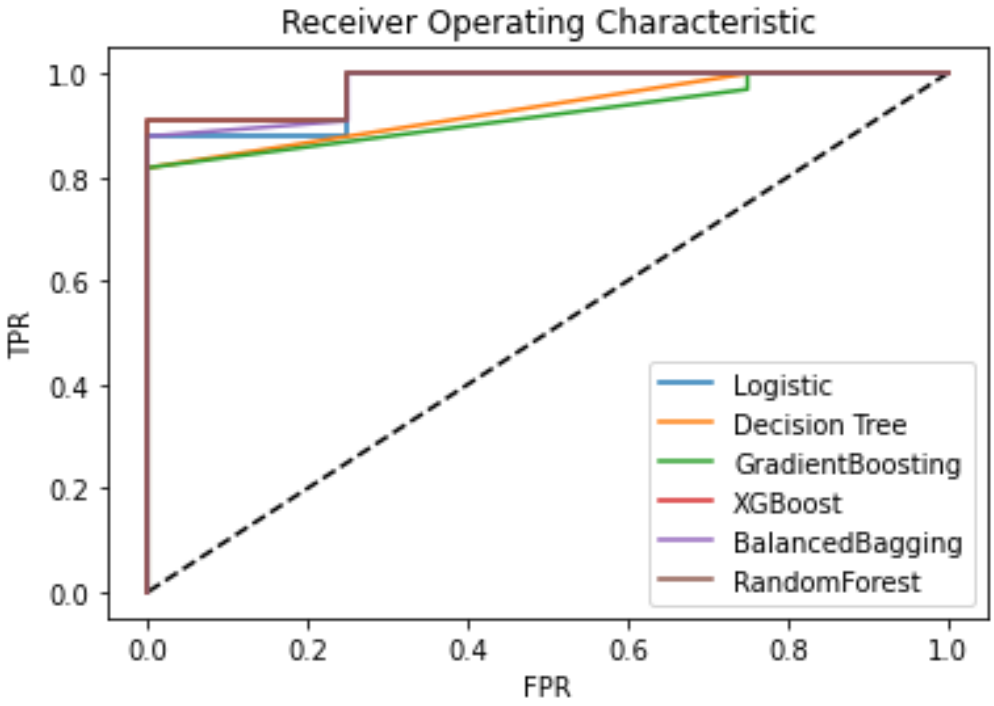


Figure 37: ROC curve

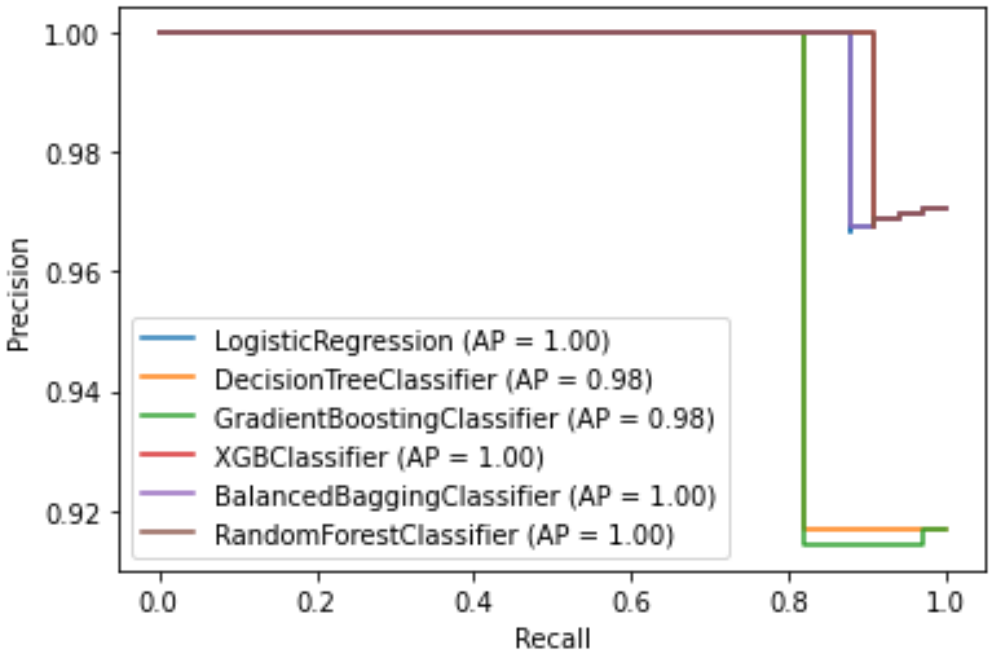


Figure 38: P-R curve

4.3 Model performance after sampling training dataset

From previous results for different combination of splits, we saw that a higher accuracy does not necessarily confirm model performance, whereas accurate predictions for both the outcomes in our case is equally important. For example, in above results for a split of 90:10 decision tree had a high accuracy for majority class as the algorithm itself does not handle the minority class well resulting in low specificity. Hence, for an imbalanced dataset specificity ,sensitivity and F1 score are appropriate metrics to choose a more balanced model. Furthermore, the model that performs good on both of these metrics such as Balanced bagging and XGBoost out of which, one has in-built sampling and the latter uses second order approximation of the scoring function. This approximation allows

XGBoost to calculate the optimal “if” condition and can then store these in its memory for the next decision tree to save time in recomputing it. In other words, it progressively adds more and more "if" conditions to the decision tree to build a stronger model.

Since some algorithms in our results are good at handling imbalanced dataset, this also led us to an inference that sampling the training dataset can be beneficial to accurately predict both the outcomes. We will be oversampling and undersampling our training dataset for different splits and measure model performance.

4.3.1 Oversampling

Oversampling in signal processing is sampling of a signal with sample rate higher than the Nyquist rate. Similarly, in machine learning oversampling is selecting more samples by duplication from the minority class to compensate for the imbalance. However, in machine learning these samples are data and not actual signal.

4.3.1.1 Train test split ratio 50:50

	Before oversampling	After oversampling
Training data shape	{1: 160, 0: 22}	{0: 160, 1: 160}

Oversampling increased the samples of the minority class from 22 to 160. We have used random oversampling which means selecting random examples from

the minority class with replacement and supplementing the training data with multiple copies of this instance, which can lead to overfitting as a single instance may be selected multiple times.

As per the evaluation metrics defined in Chapter 3, Table 7 shows the results of metrics for all the models followed by Gain charts , Lift curves and K-S statistic charts.

Methods	**Accuracy**	**Precision**	**Recall**	**F1score**	**Specificity**	**Mis-classification**
Logistic regression	86%	68%	88%	77%	68%	14%
Decision Tree	81%	100%	79%	88%	100%	19%
Gradient Boosting	94%	68%	98%	80%	68%	6%
XGBoost	95%	64%	99%	77%	64%	5%
Balanced Bagging	93%	55%	99%	70%	55%	7%
Random Forests	93%	50%	99%	66%	50%	7%

Table 7: Metric results for train test split ratio of 50:50

Oversampling for a split of 50:50 performed well on our defined evaluation metrics except precision for some models and became more robust towards minority class. Even though the misclassification rate for Decision trees increased considerably, the model became robust towards minority class with high specificity and F-1 score. However, the increment in precision and accuracy is not significant.

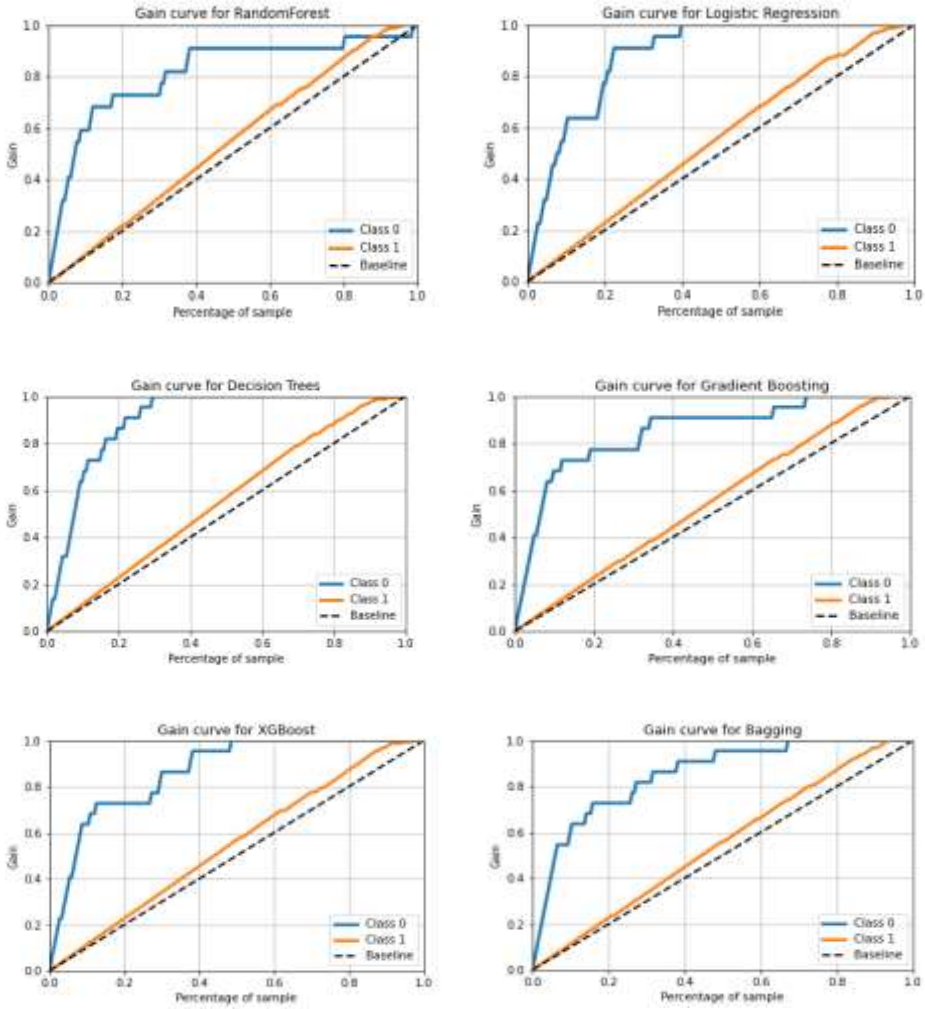


Figure 39: Gain curves for 50:50 training-testing data split for all the models

In all the gain charts for 50:50 split, Decision tree gave the highest gain of ~84% for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set. Undersampling improved performance of decision tree classifier which does not handle imbalanced classes well.

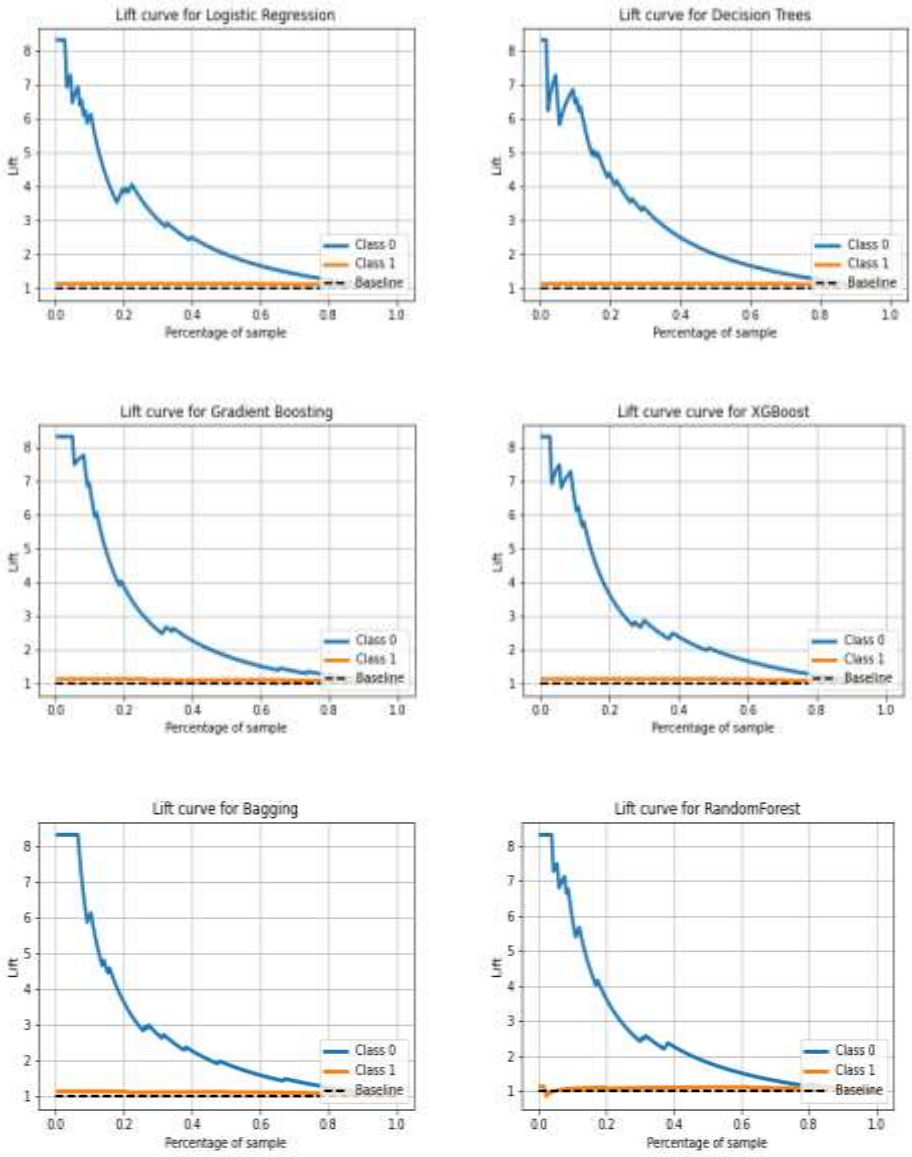


Figure 40: Lift curves for 50:50 training-testing data split for all the models

In all the lift charts for 50:50 split, Decision trees gave the highest lift of greater than 4 followed by Gradient boosting and Random forest for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models.

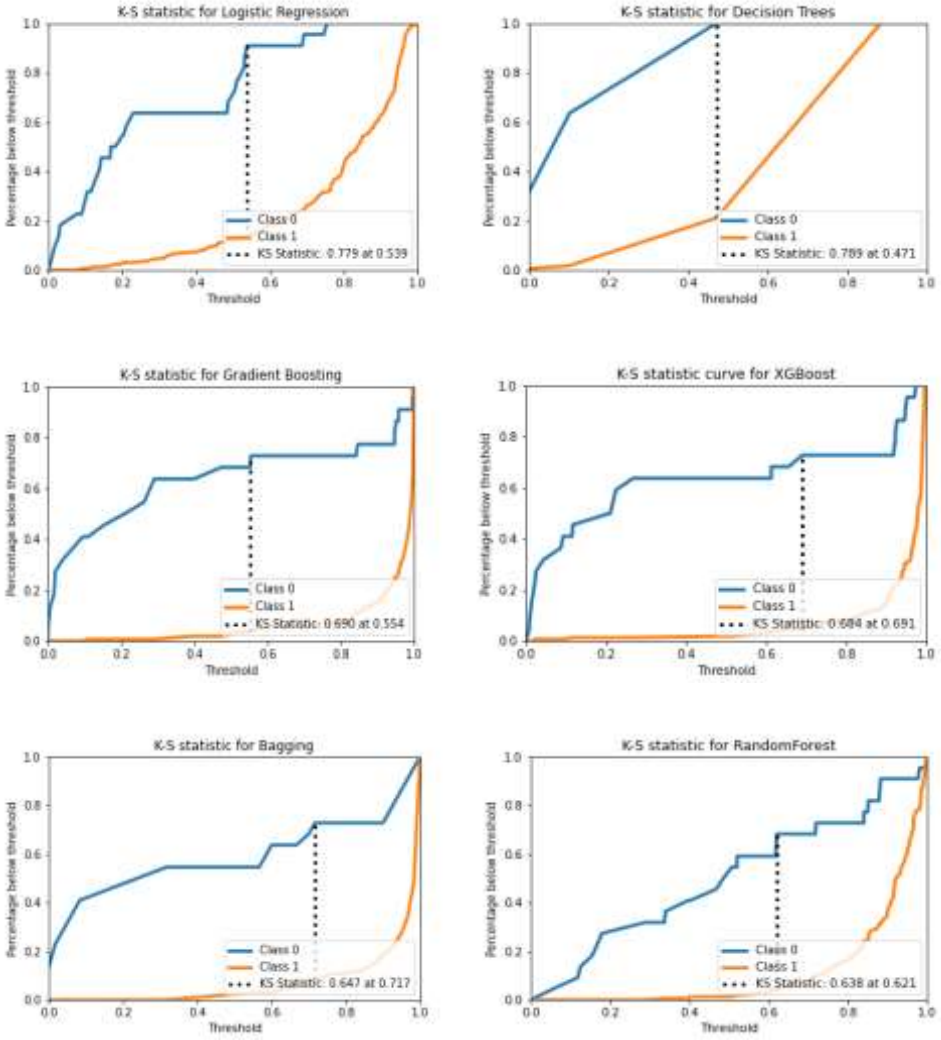


Figure 41: K-S curves for 50:50 training-testing data split for all the models

In all the K-S charts for 50:50 split, Decision trees gave the maximum separation between both the classes at K-S statistic value of 0.789.

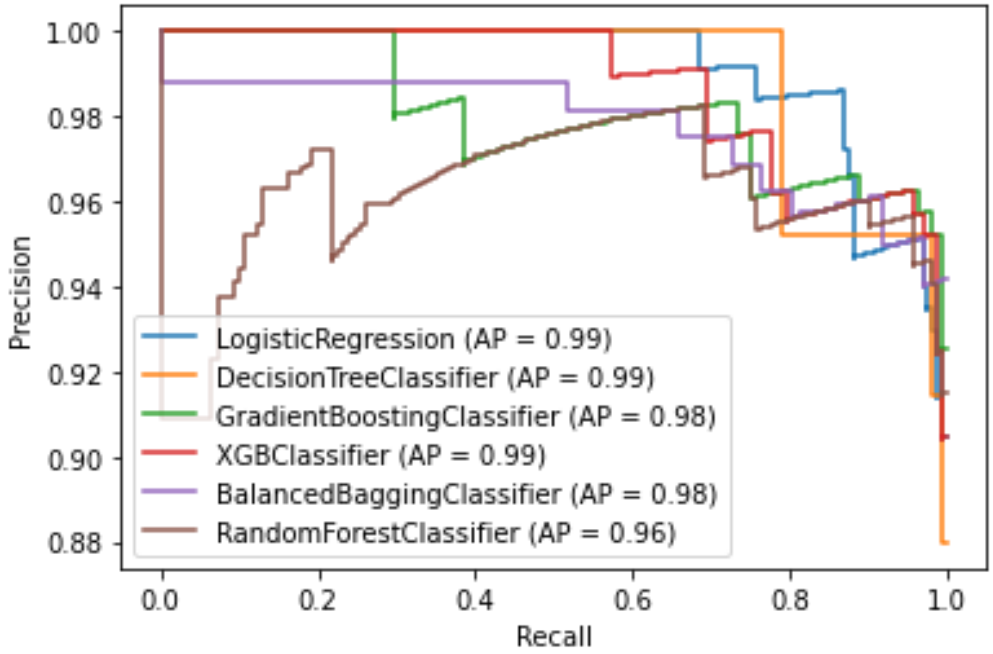


Figure 42: ROC curve

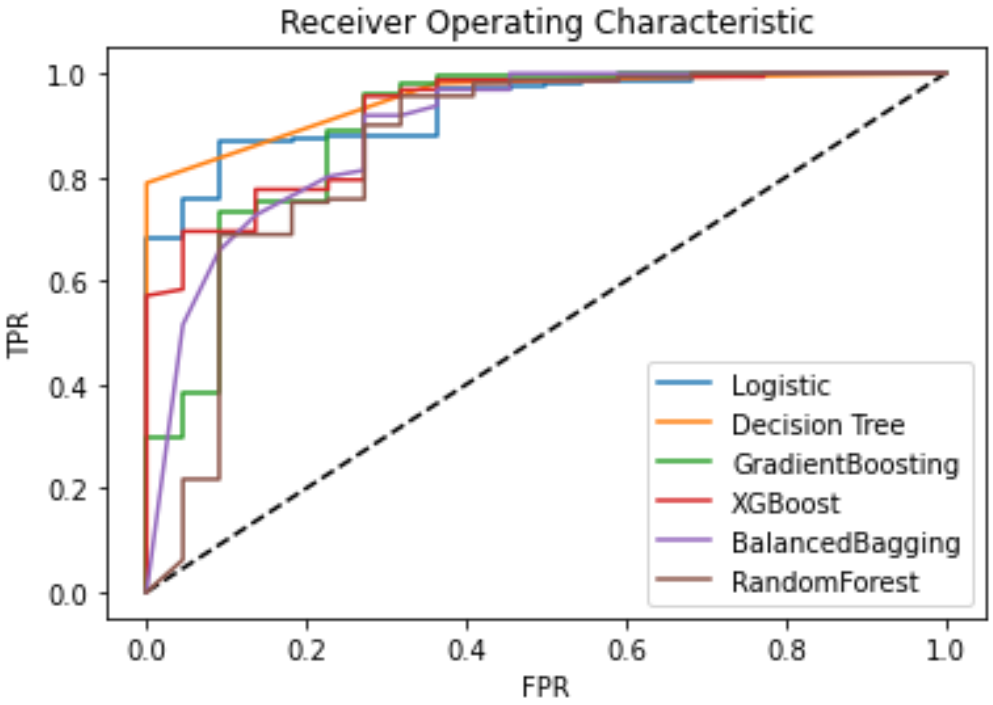


Figure 43: PR curve

4.3.1.2 Train test split ratio 60:40

	Before oversampling	After oversampling
Training data shape	{1: 193, 0: 26}	{1: 193, 0: 193}

As per the evaluation metrics defined in Chapter 3, Table 8 shows the results of metrics for all the models followed by Gain charts , Lift curves and K-S statistic charts.

Methods	**Accuracy**	**Precision**	**Recall**	**F1score**	**Specificity**	**Mis-classification**
Logistic regression	85%	78%	86%	82%	78%	15%
Decision Tree	82%	89%	80%	84%	89%	18%
Gradient Boosting	92%	72%	95%	76%	72%	8%
XGBoost	94%	72%	97%	83%	72%	6%
Balanced Bagging	94%	56%	99%	71%	56%	6%
Random Forests	96%	67%	100%	80%	67%	4%

Table 8: Metric results for train test split ratio of 60:40

Oversampling training dataset for a split of 60:40 did not make an impact on evaluation metrics considerably but it improved precision and F-1 score for models trained on data without sampling. Oversampling training data can be an appropriate choice as it improves metrics for all models except Balanced bagging, which performed well on imbalanced data without performing oversampling.

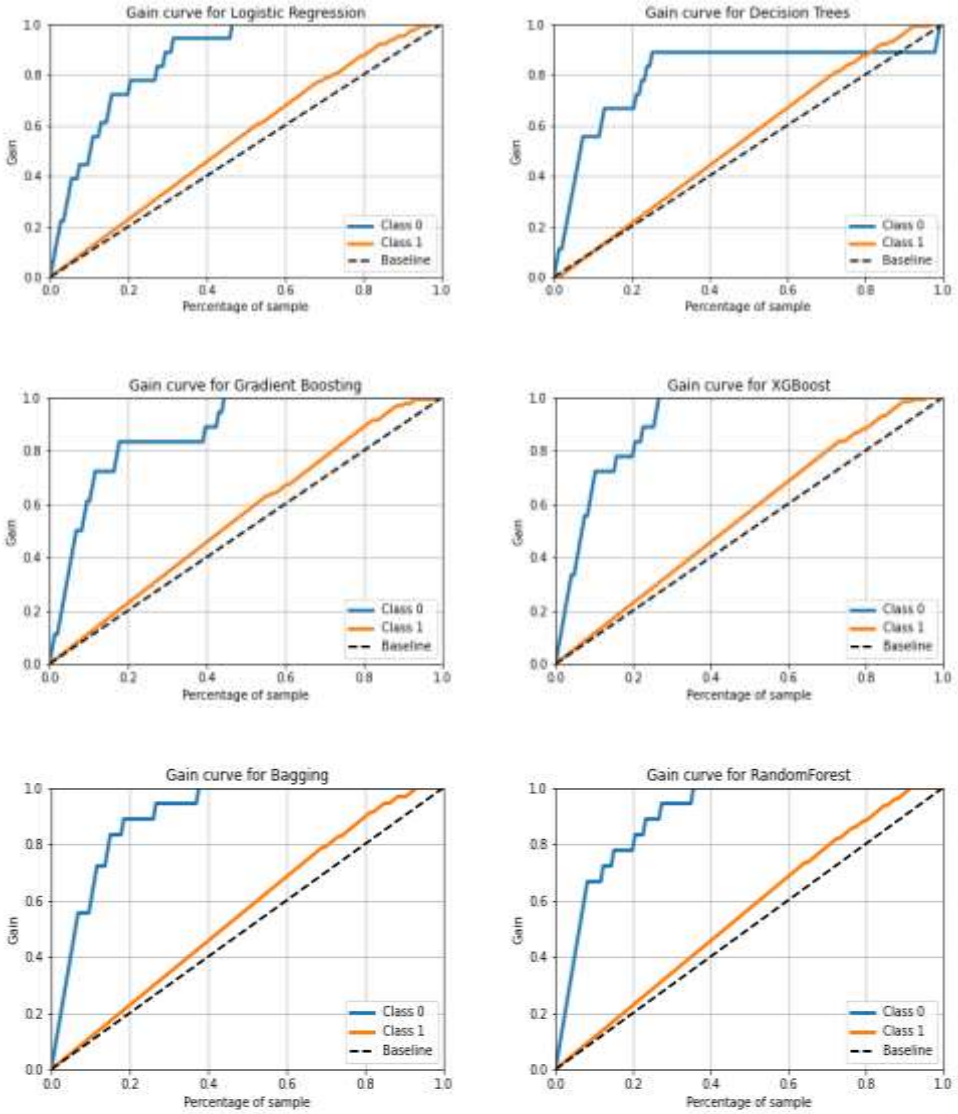


Figure 44: Gain curves for 60:40 training-testing data split for all the models

In all the gain charts for 60:40 split, Bagging gave the highest gain of ~90% followed by Gradient boosting for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set. On comparing this with previous experiment when the data was not sampled, Gradient boosting had the highest gain although it has low F-1 score.

Random forests and XGBoost performed well on all the evaluation metrics and are more robust to class imbalance.

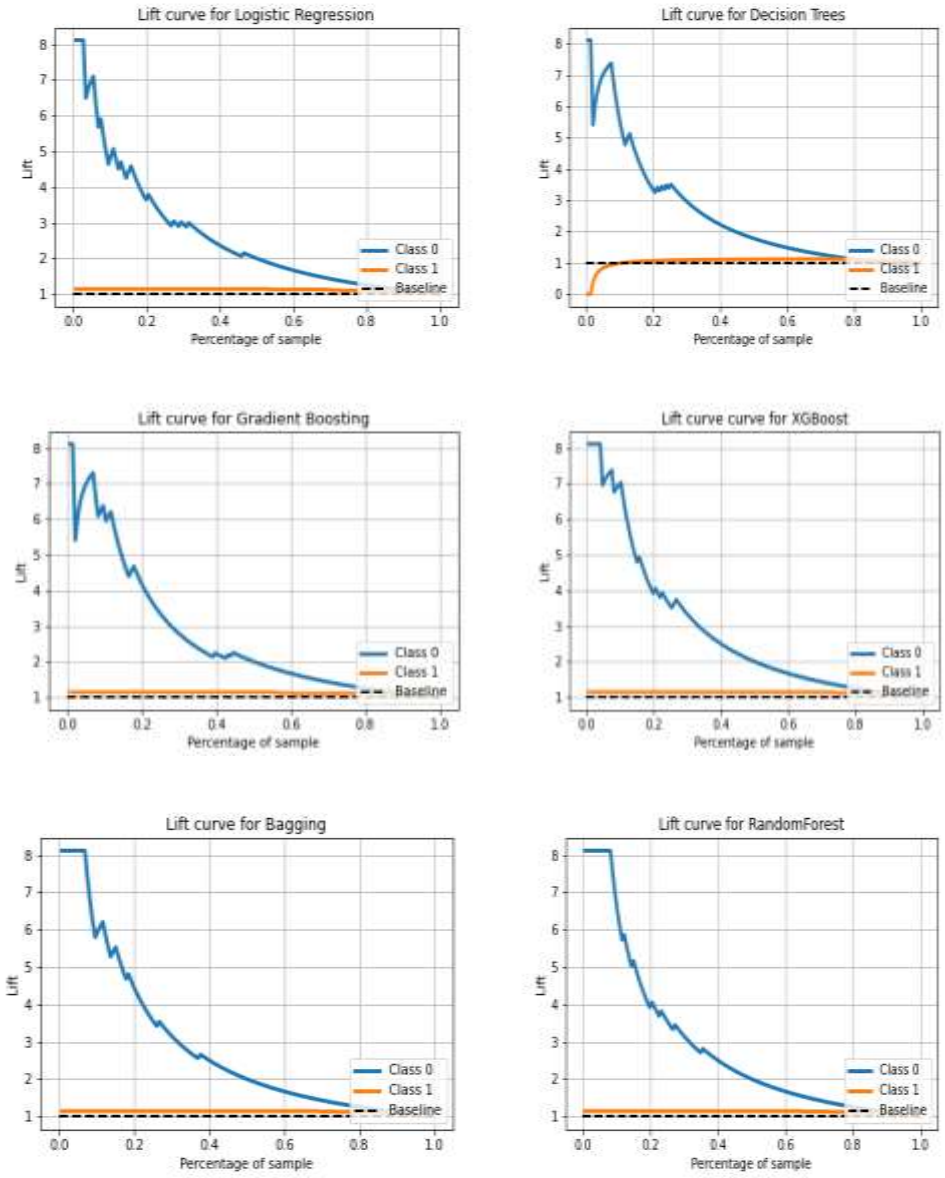


Figure 45: Lift curves for 60:40 training-testing data split for all the models

In all the lift charts for 60:40 split, Bagging gave the highest lift of ~ 4.2 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models.

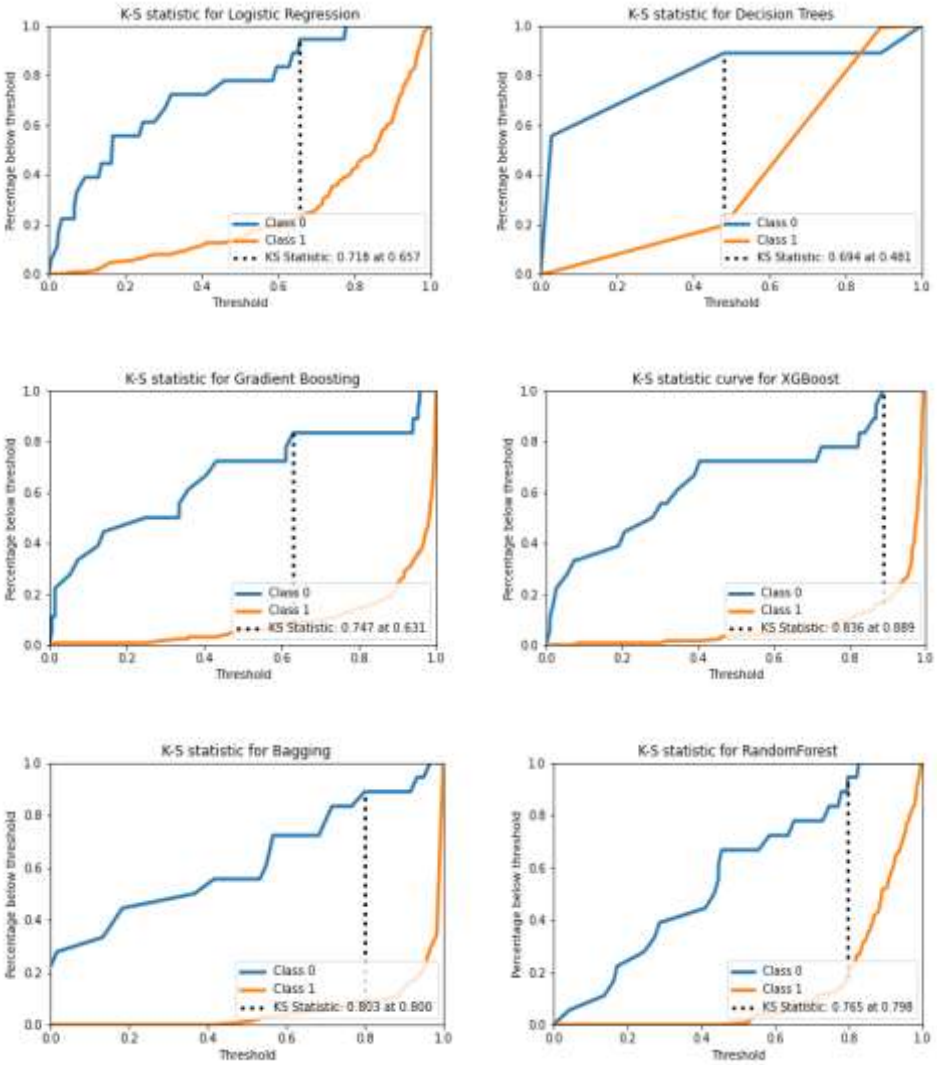


Figure 46: K-S curves for 60:40 training-testing data split for all the models

In all the K-S charts for 60:40 split, XGboost gave the maximum separation between both the classes at K-S statistic value of 0.83 which also explains the high F1 score and specificity of the model. As discussed previously, Gradient

boosting had the highest gain but suffered from low F-1 score and XGBoost is an appropriate choice for more balanced model.

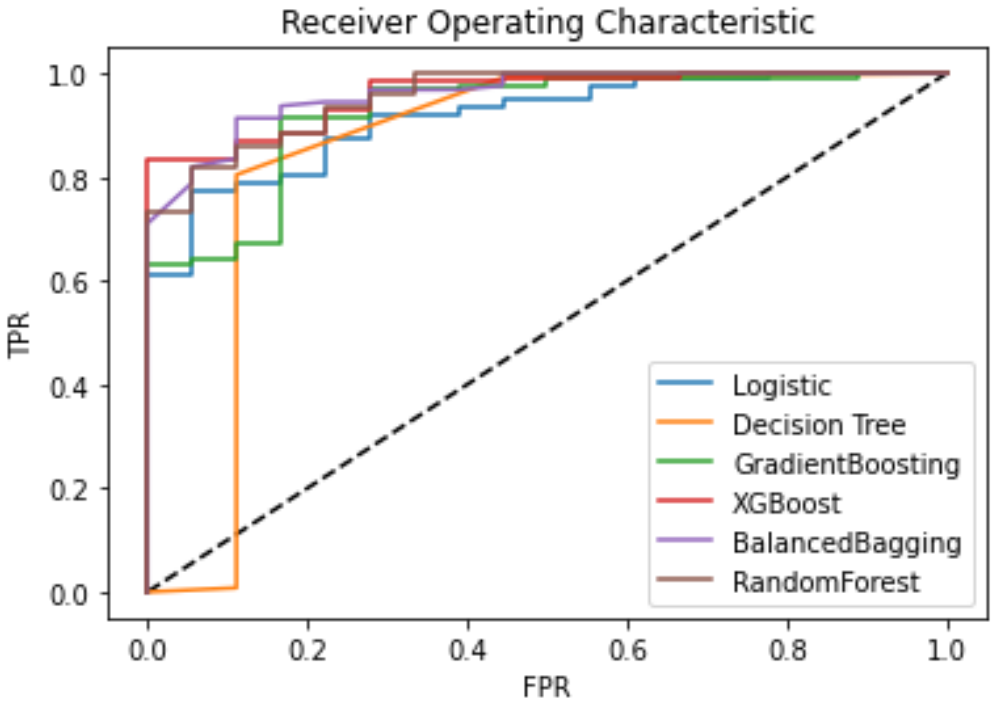


Figure 47: ROC curve

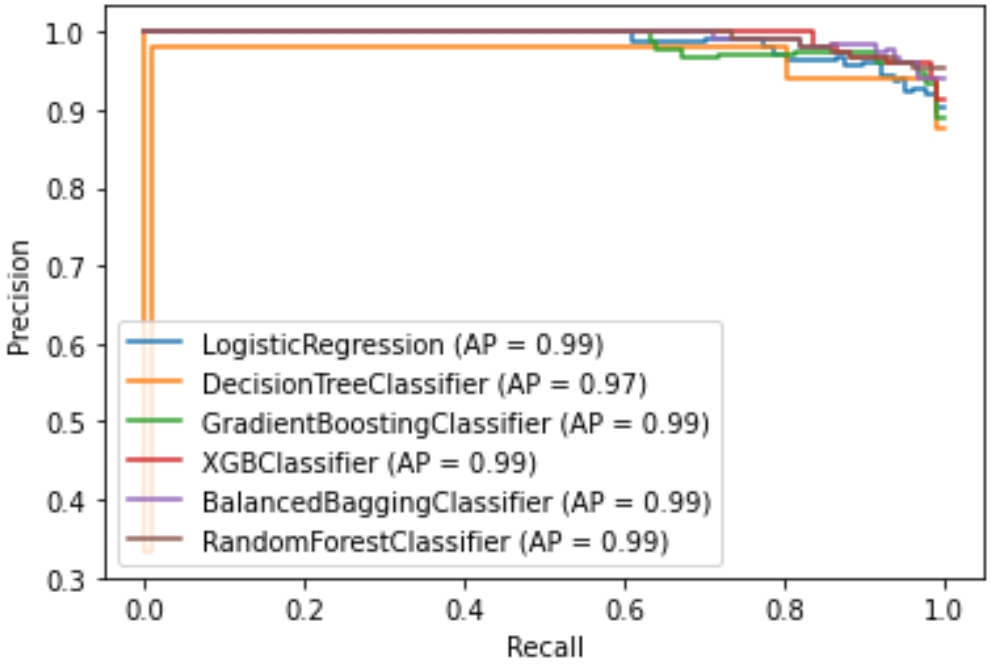


Figure 48: P-R curve

4.3.1.3 Train test split ratio 80:20

	Before oversampling	After oversampling
Training data shape	{1: 257, 0: 35}	{0: 257, 1: 257}

As per the evaluation metrics defined in Chapter 3, Table 9 shows the results of metrics for all the models followed by Gain charts , Lift curves and K-S statistic charts.

Methods	**Accuracy**	**Precision**	**Recall**	**F1score**	**Specificity**	**Mis-classification**
Logistic regression	85%	89%	84%	87%	89%	15%
Decision Tree	84%	100%	81%	90%	100%	16%

Gradient Boosting	95%	89%	95%	92%	89%	5%
XGBoost	96%	89%	97%	93%	89%	4%
Balanced Bagging	96%	78%	98%	87%	78%	4%
Random Forests	97%	89%	98%	93%	89%	3%

Table 9: Metric results for train test split ratio of 50:50

Oversampling improved specificity and precision of the models as compared to models with no sampling, ultimately making them more robust towards minority class. The highest change can be seen in Decision trees, Gradient boosting and Random forests models.

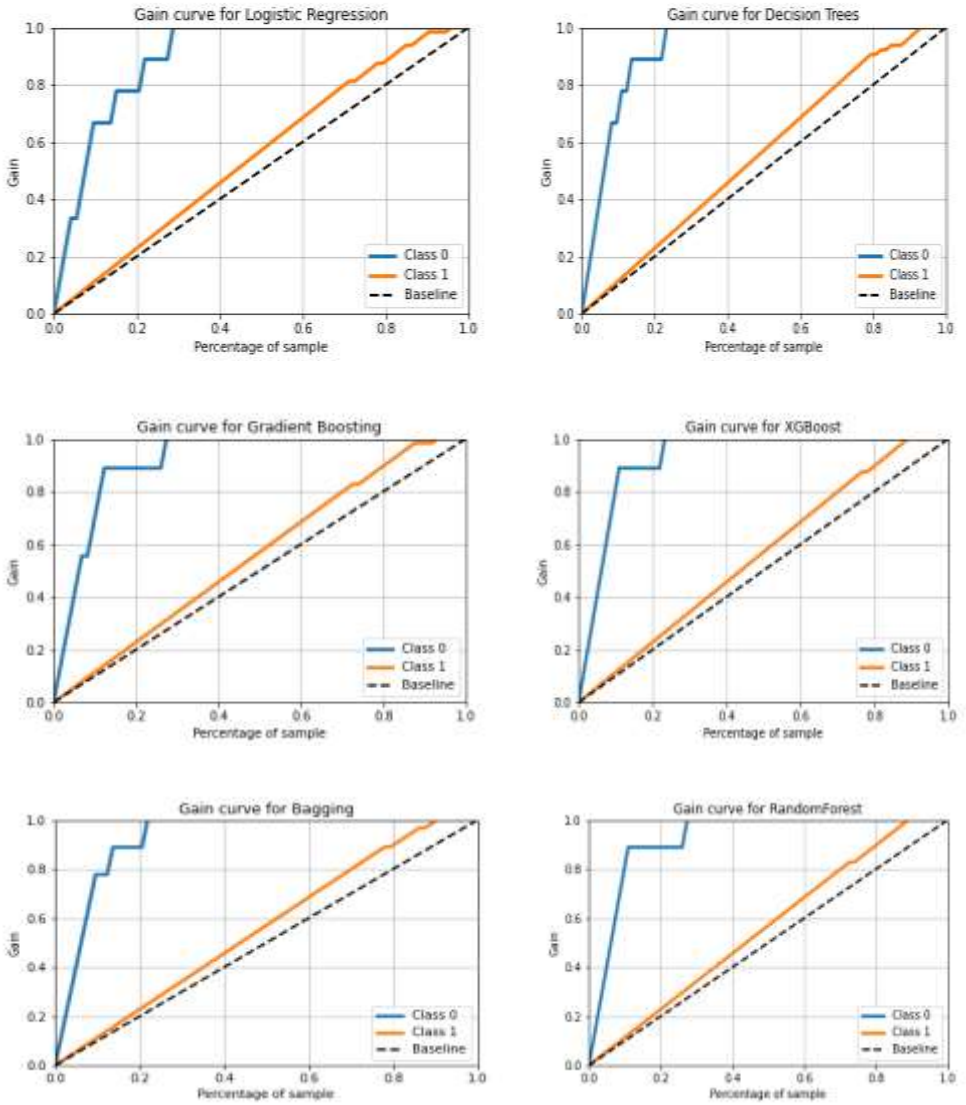


Figure 30: Gain curves for 80:20 training-testing data split for all the models

In all the gain charts for 80:20 split, all models except logistic regression gave gain of greater than ~90% for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set. Random forest model performs well on all the evaluation metrics with lowest misclassification percentage and is an appropriate choice for medical practice.

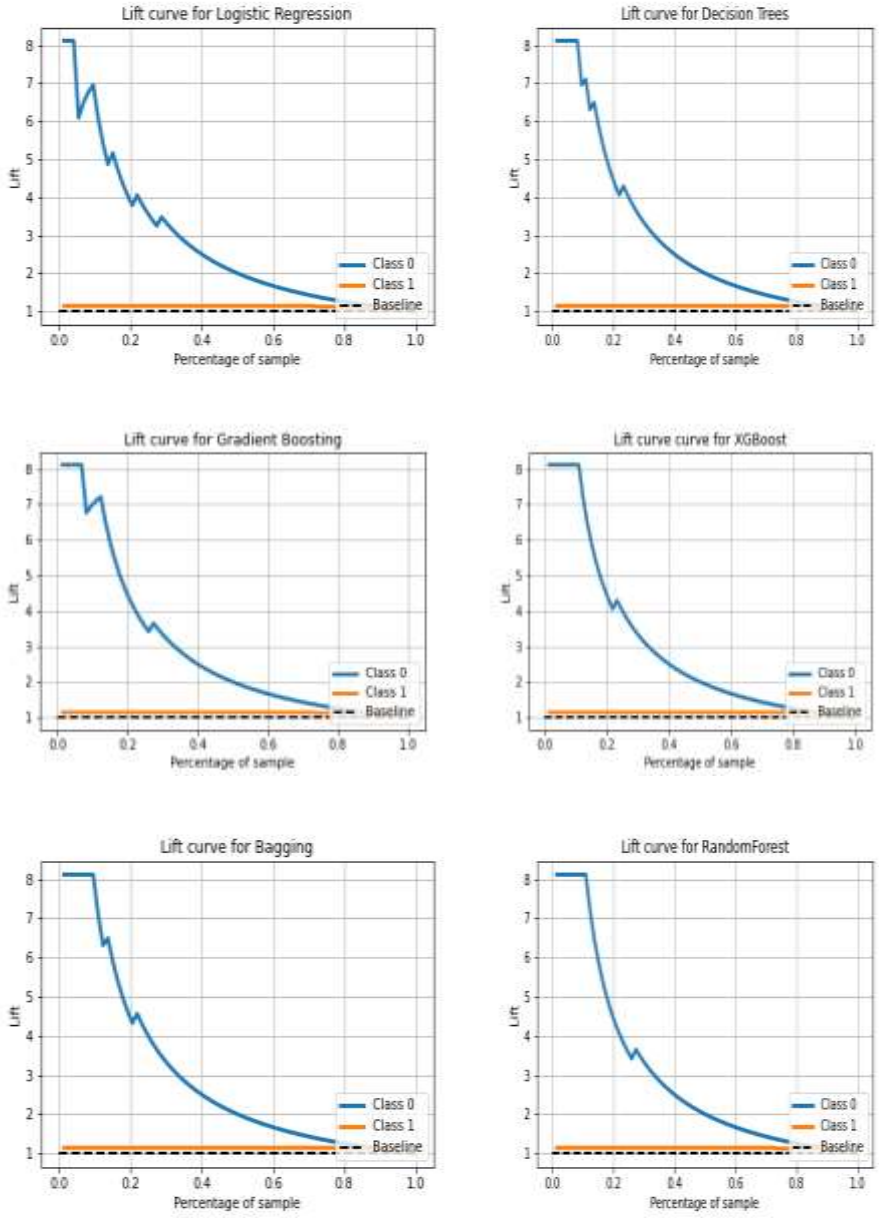


Figure 49: Lift curves for 80:20 training-testing data split for all the models

In all the lift charts for 80:20 split, all models except logistic regression gave a high lift of greater than ~ 4.2 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models.

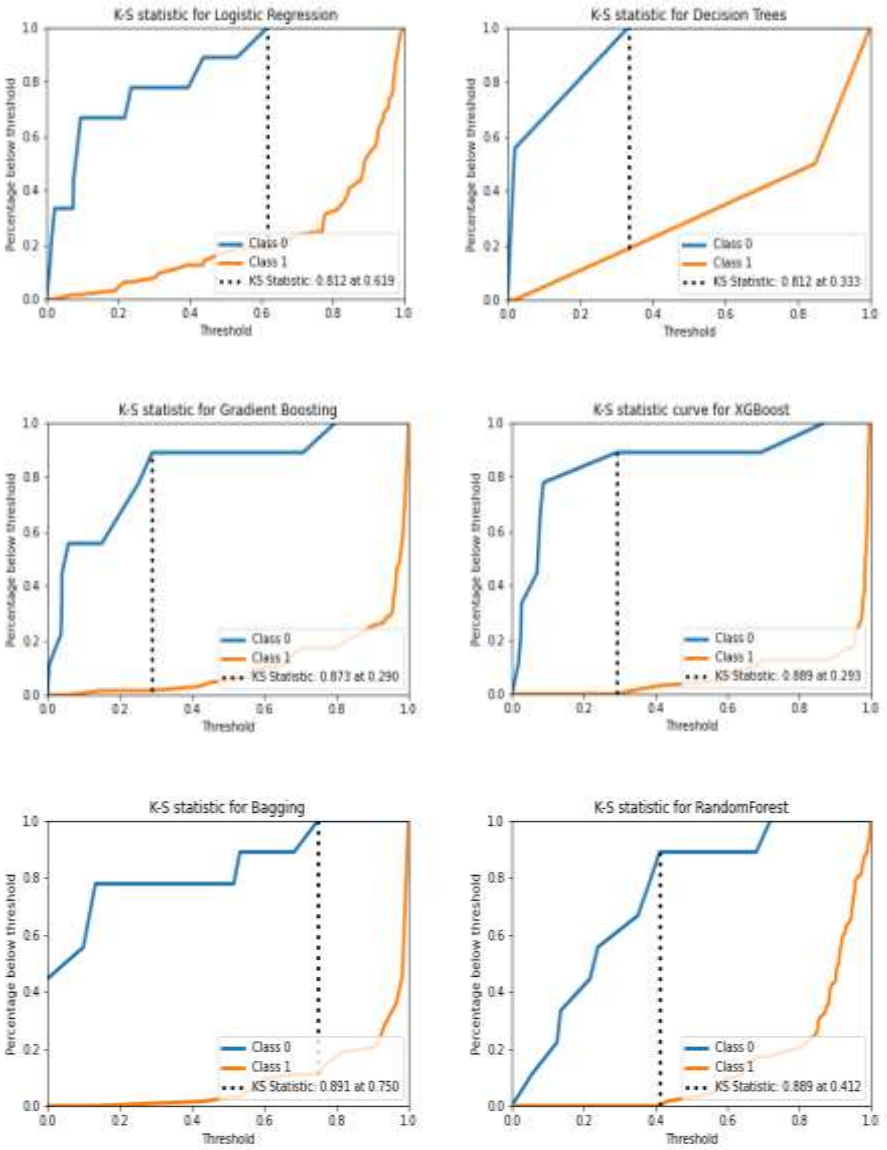


Figure 50: K-S curves for 80:20 training-testing data split for all the models

In all the K-S charts for 80:20 split, bagging gave the maximum separation between both the classes at K-S statistic value of 0.891 followed by Random forests and XGBoost with K-S statistic value of 0.889. However, Random forests and XGBoost achieves this separation at top 41.2% and 29.3% of the patient population as compared to the top 75% of the population in Bagging.

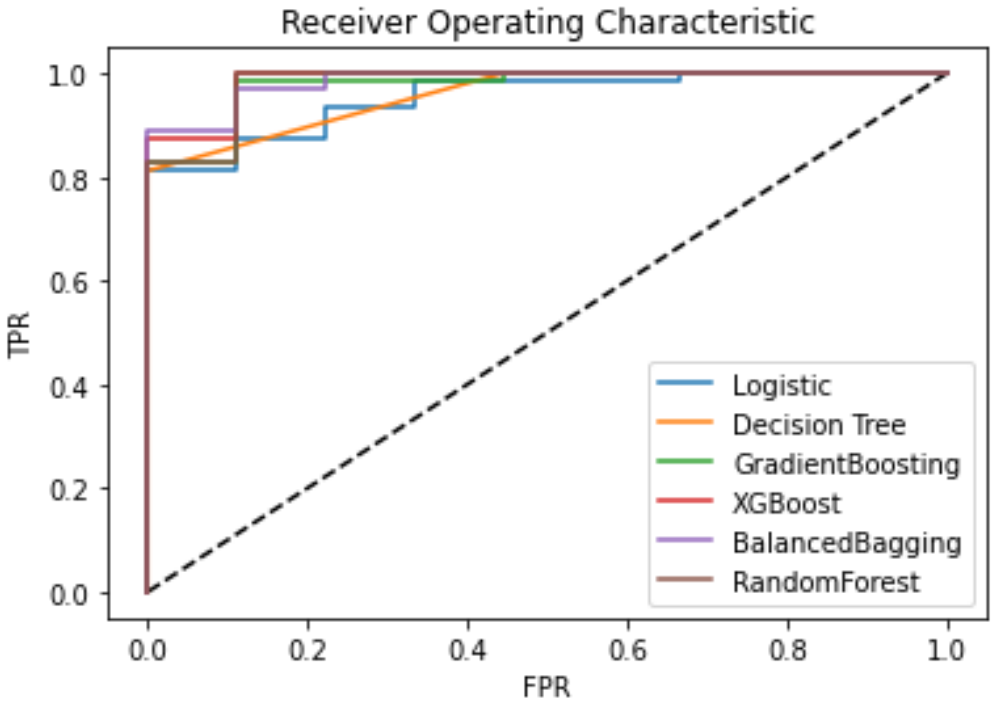


Figure 51: ROC curve

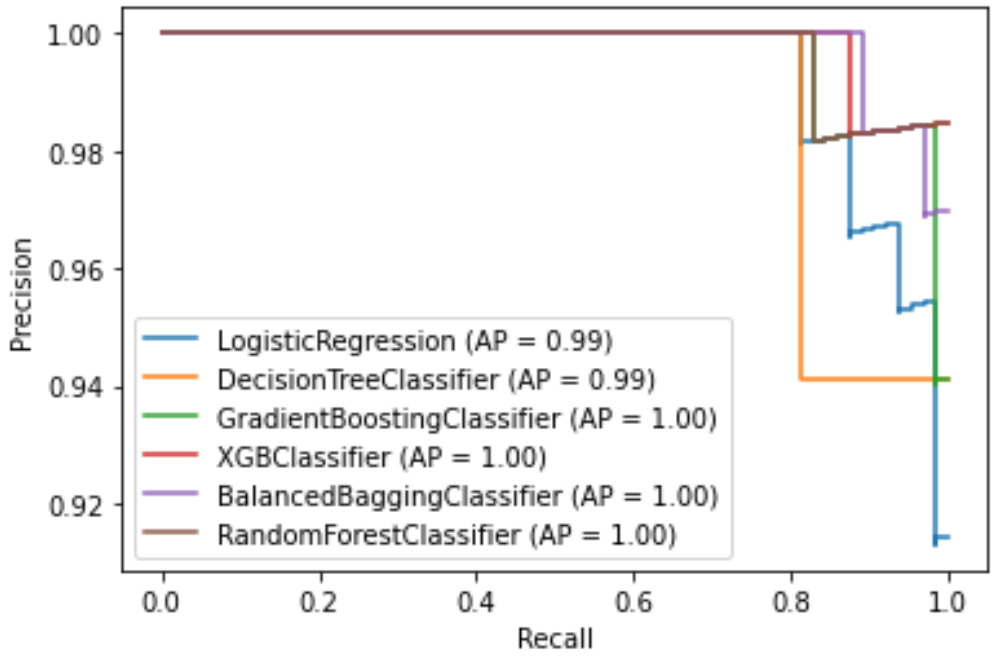


Figure 52: P-R curve

4.3.1.4 Train test split ratio 90:10

	Before oversampling	After oversampling
Training data shape	{1: 288, 0: 40}	{1: 288, 0: 288}

As per the evaluation metrics defined in Chapter 3, Table 10 shows the results of metrics for all the models followed by Gain charts , Lift curves and K-S statistic charts.

Methods	**Accuracy**	**Precision**	**Recall**	**F1score**	**Specificity**	**Mis-classification**
Logistic regression	81%	75%	82%	78%	75%	19%
Decision Tree	84%	100%	82%	90%	100%	16%
Gradient Boosting	92%	75%	94%	83%	75%	8%
XGBoost	95%	75%	97%	85%	75%	5%
Balanced Bagging	97%	75%	100%	86%	75%	3%
Random Forests	97%	75%	100%	86%	75%	3%

Table 10: Metric results for train test split ratio of 90:10

Oversampling training data for a split of 90:10 lowered the specificity and F-1 score of the models making the model prone to inaccurate predictions on minority class. However, precision increases considerably indicating overfitting towards majority class.

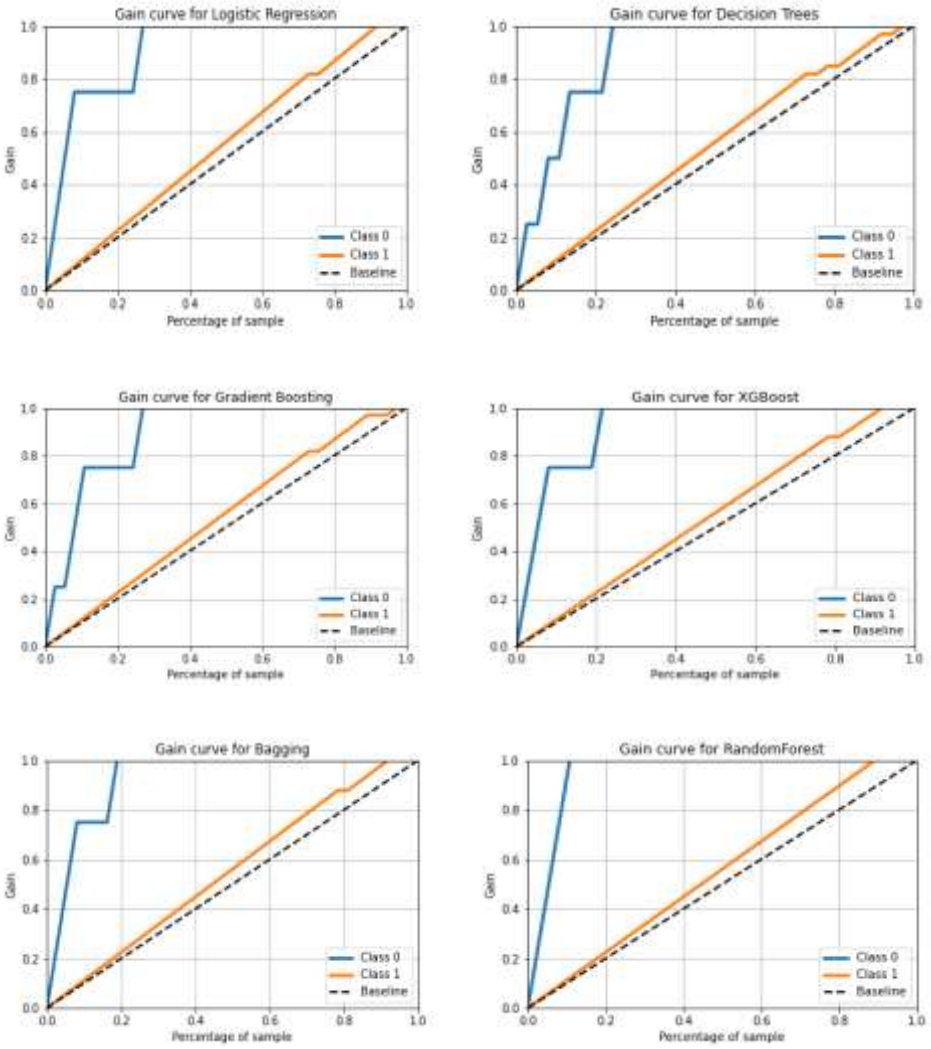


Figure 53: Gain curves for 90:10 training-testing data split for all the models

In all the gain charts for 90:10 split, Random forests and Bagging gave highest gain for capturing patients whose survival status at a given time is alive, when targeting less than top 20% of the patient population in the testing set. These two models perform well on all the evaluation metrics and have the least misclassification percentage.

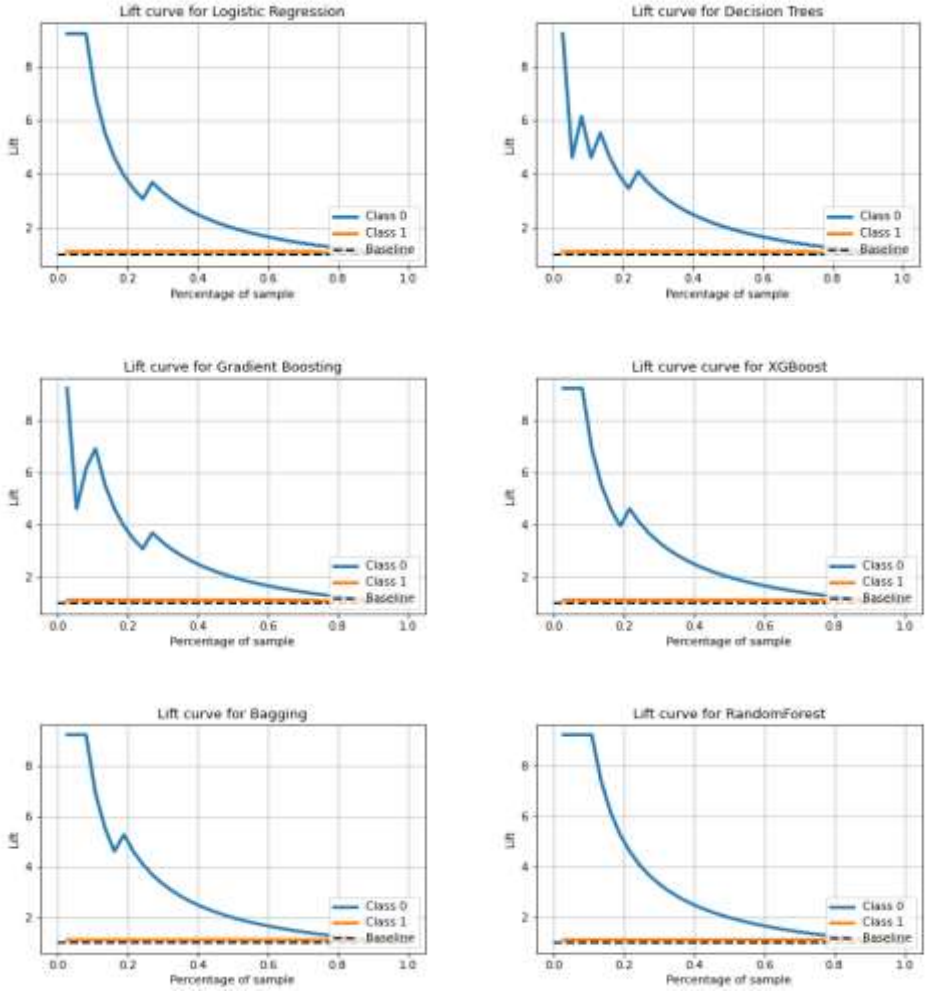


Figure 54: Lift curves for 90:10 training-testing data split for all the models

In all the lift charts for 90:10 split, Bagging and Random forests gave a high lift of greater than ~ 4 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models.

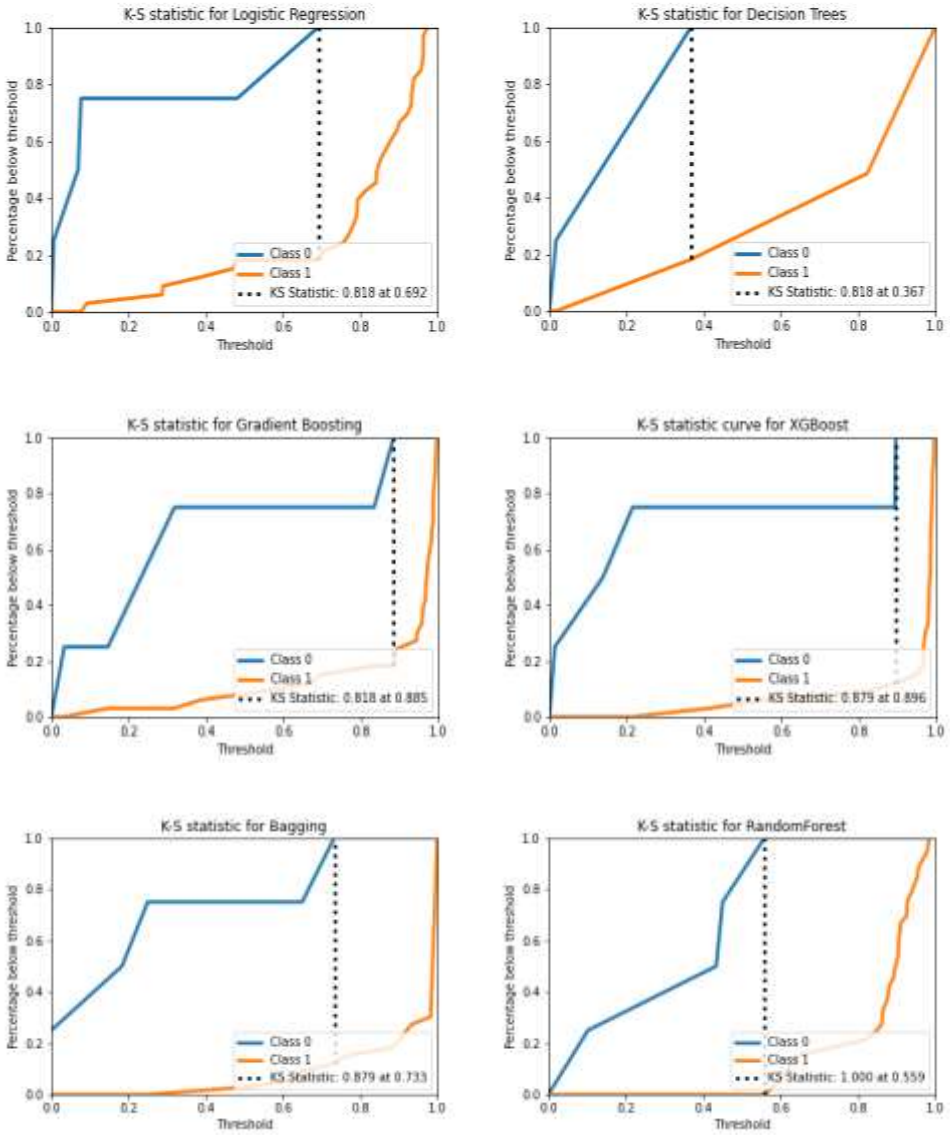


Figure 55: K-S curves for 90:10 training-testing data split for all the models

In all the K-S charts for 90:10 split, Random forests gave the maximum separation between both the classes at K-S statistic value of 1 at top 55% of the population.

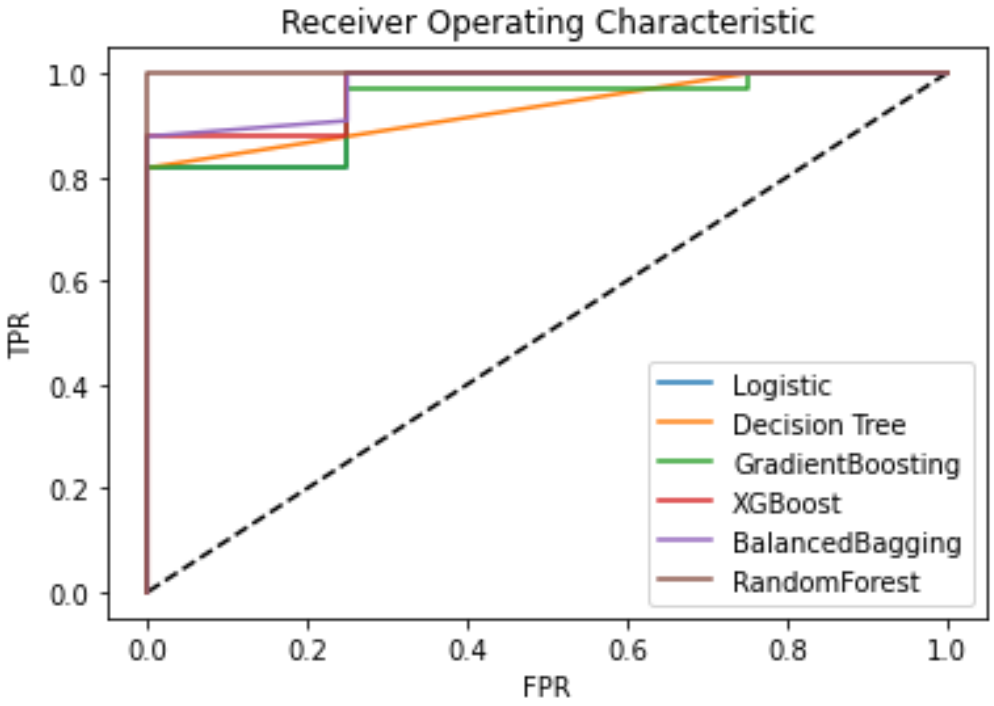


Figure 37: ROC curve

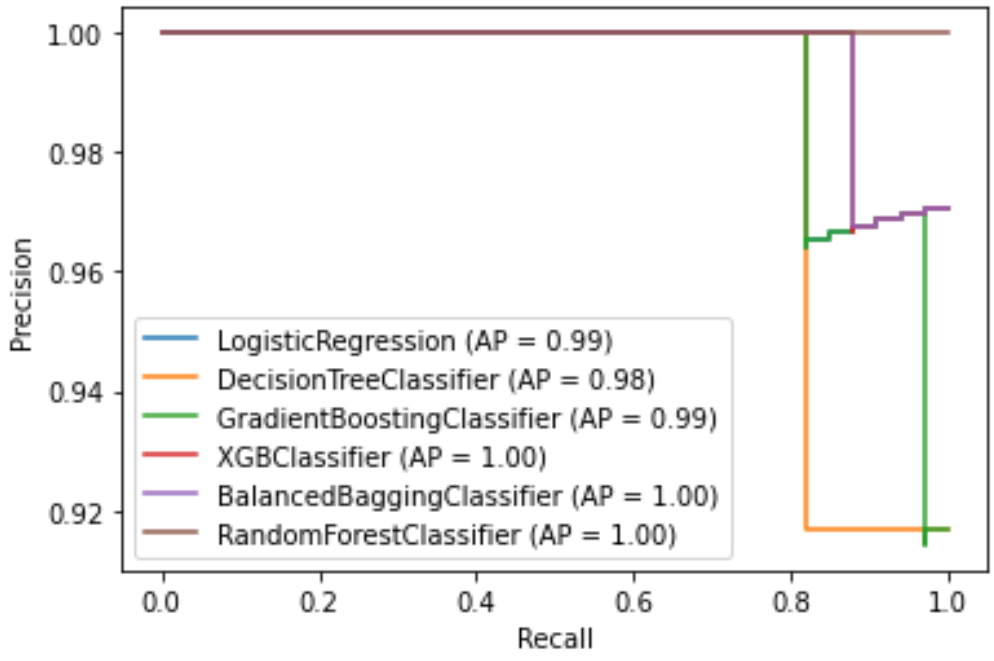


Figure 56: P-R curve

4.3.2 Undersampling

Undersampling involves removing some observations from the majority class until the majority and minority class is balanced out.

4.3.2.1 Train test split ratio 50:50

	Before undersampling	After undersampling
Training data shape	{1: 160, 0: 22}	{0: 22, 1: 22}

As per the evaluation metrics defined in Chapter 3, Table 11 shows the results of metrics for all the models followed by Gain charts , Lift curves and K-S statistic charts.

Methods	**Accuracy**	**Precision**	**Recall**	**F1score**	**Specificity**	**Mis-classification**
Logistic regression	84%	86%	83%	85%	86%	16%
Decision Tree	85%	91%	84%	87%	91%	15%
Gradient Boosting	80%	86%	80%	83%	86%	20%
XGBoost	79%	82%	79%	80%	82%	21%
Balanced Bagging	85%	91%	84%	88%	91%	15%
Random Forests	83%	82%	83%	82%	82%	17%

Table 11: Metric results for train test split ratio of 50:50

Undersampling the training dataset for a split of 50:50 decreased the accuracy, recall of all the models but increased the specificity, precision and F-1 scores making the models more robust towards both the classes when compared to the

models without sampling. On comparing these results with oversampling these results give promising response for minority class as specificity for all the models is greater than 80% whereas for oversampling it goes as low as 50%.

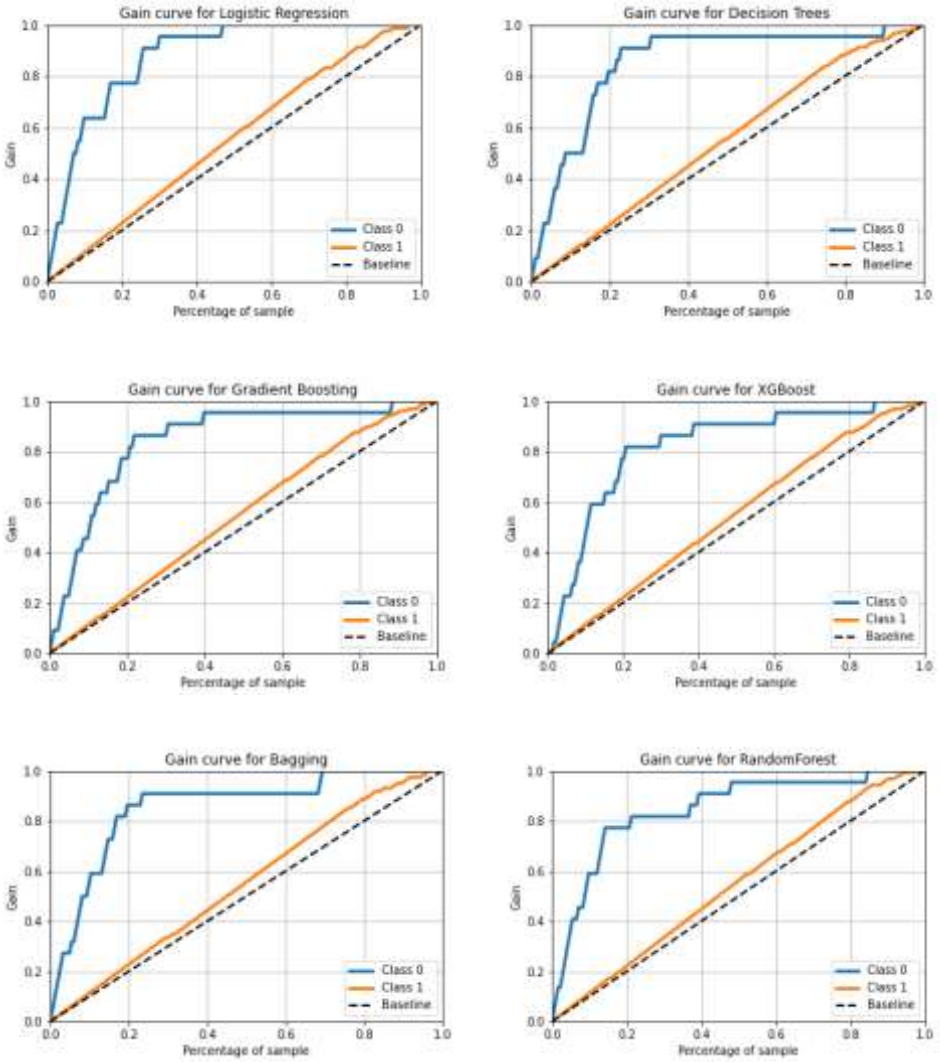


Figure 57: Gain curves for 50:50 training-testing data split for all the models

In all the gain charts for 50:50 split, Bagging has the highest gain for capturing patients whose survival status at a given time is alive, when targeting less than top 20% of the patient population in the testing set. This explains for highest

accuracy for balanced bagging, ultimately leading to lowest misclassification rate across all models.

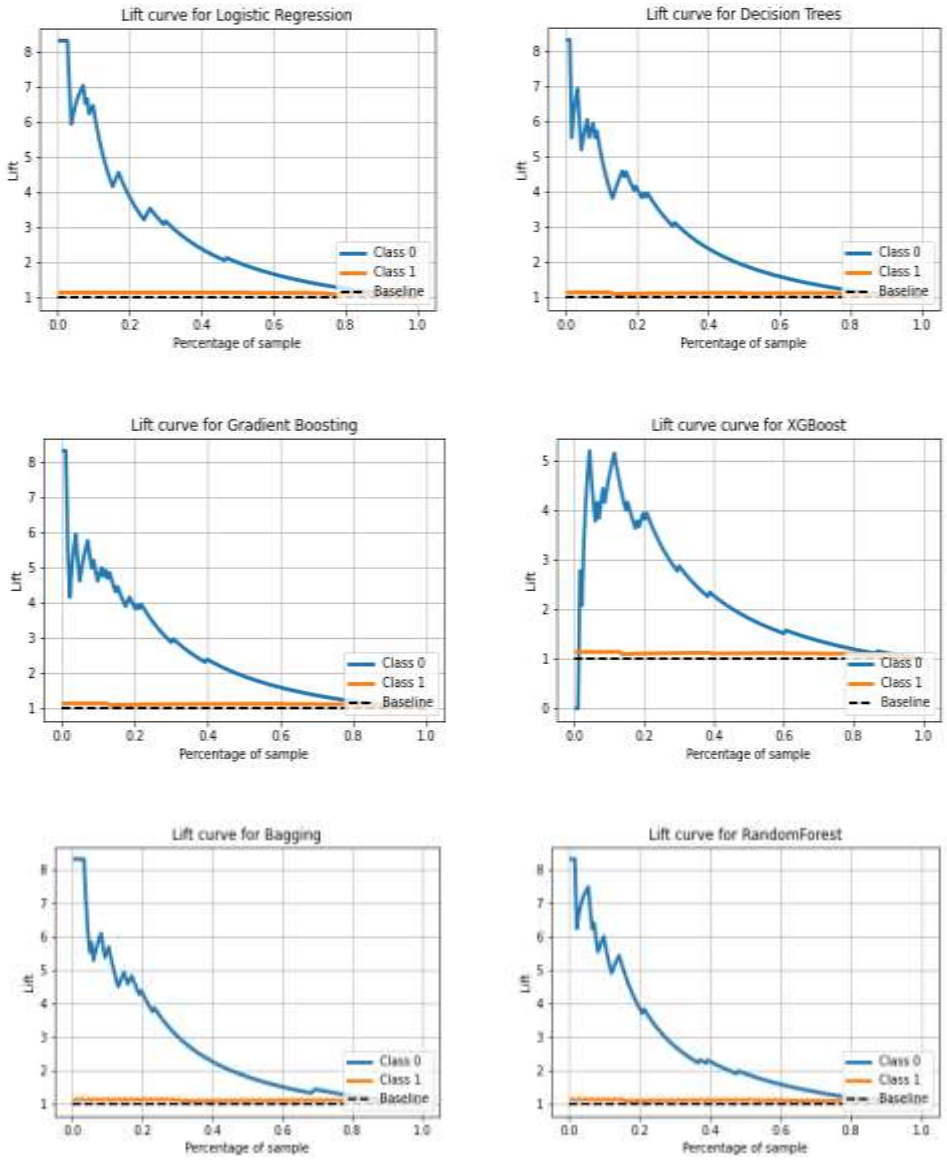


Figure 58: Lift curves for 50:50 training-testing data split for all the models

In all the lift charts for 50:50 split, Bagging gave a high lift of greater than ~ 4 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models.

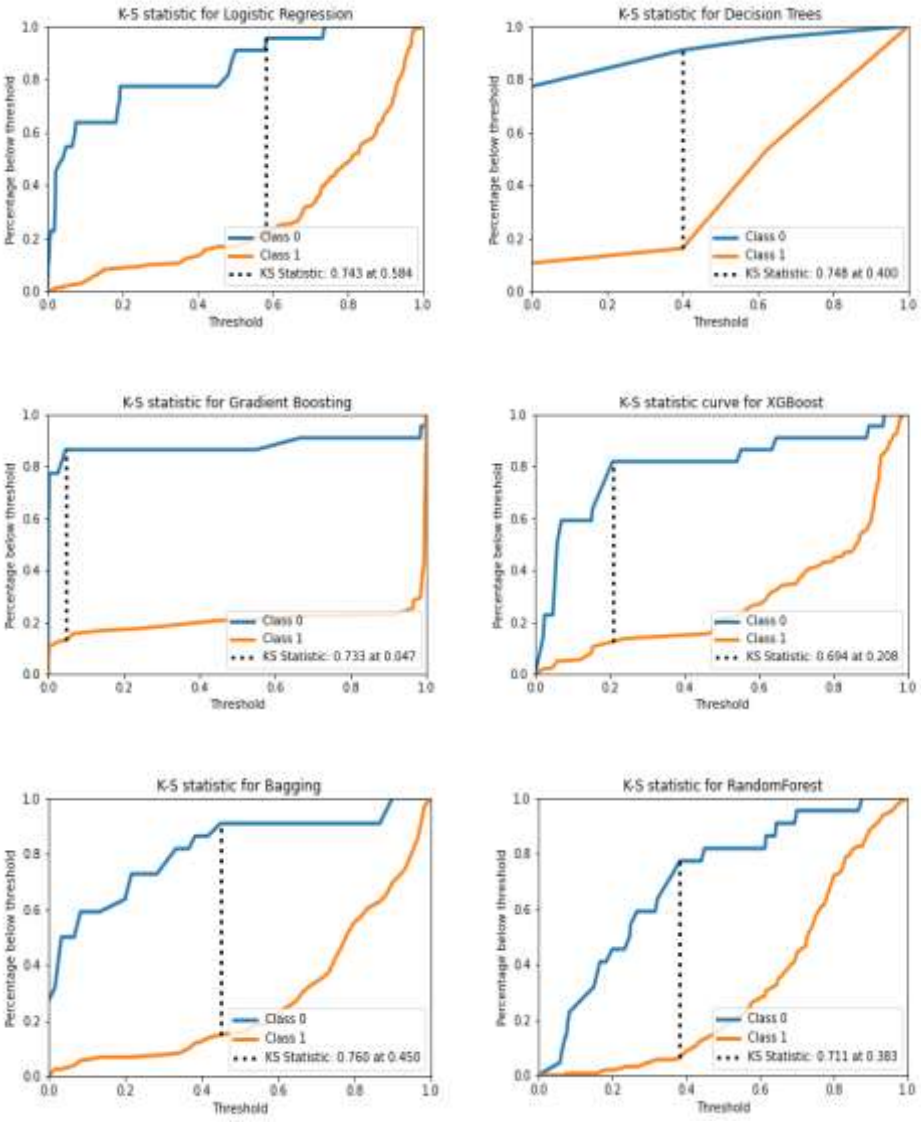


Figure 59: K-S curves for 50:50 training-testing data split for all the models

In all the K-S charts for 50:50 split, Bagging gave the maximum separation between both the classes at K-S statistic value of 0.76 at top 45% of the population.

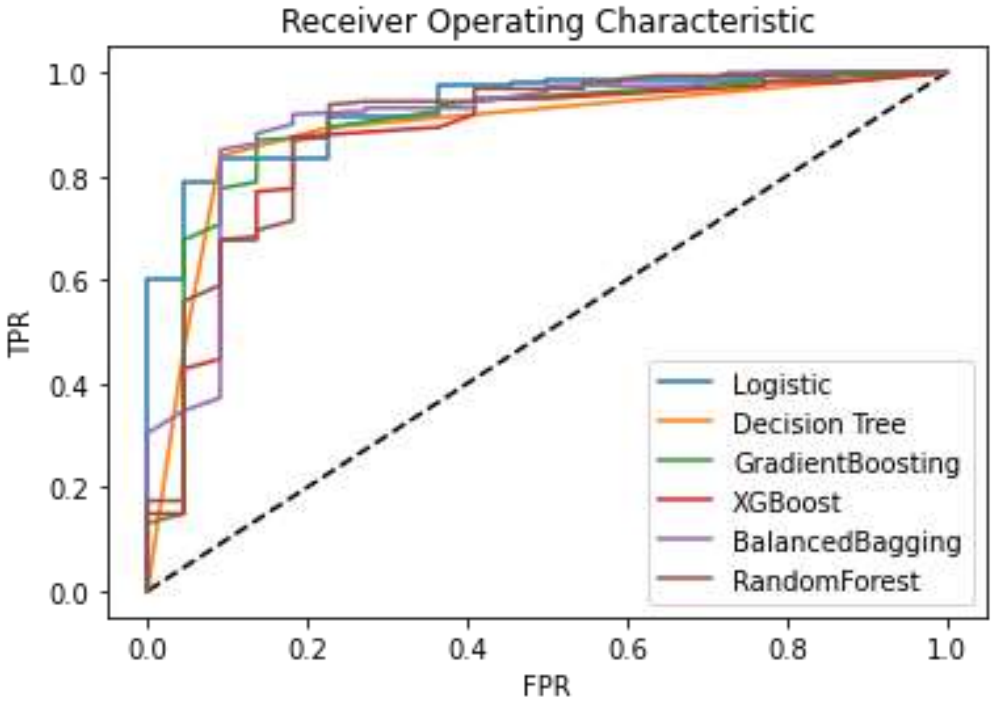


Figure 60: ROC curve

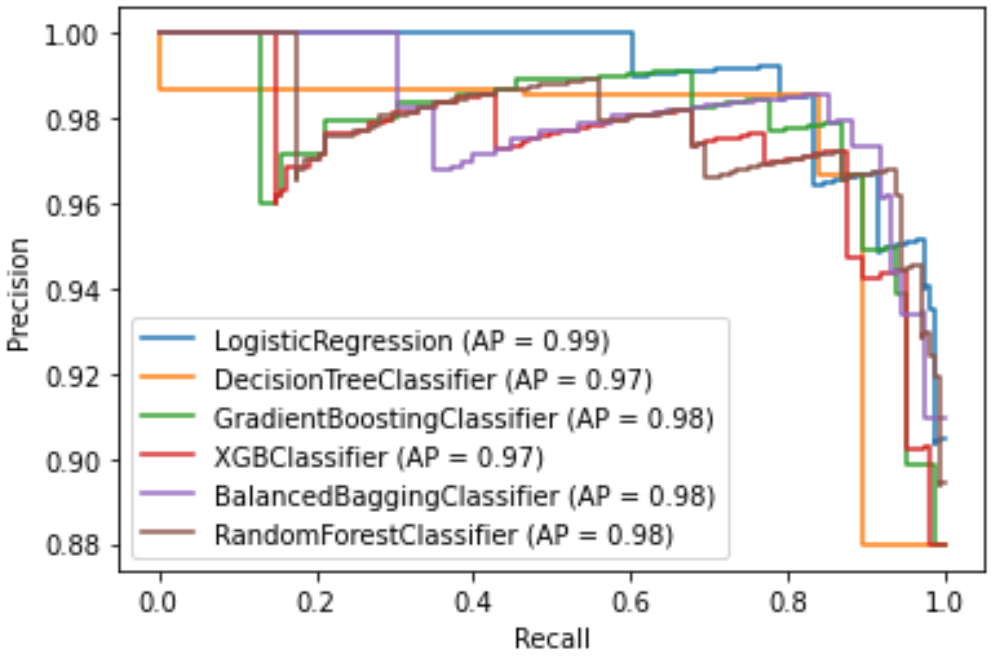


Figure 61: PR curve

4.3.2.2 Train test split ratio 60:40

	Before undersampling	After undersampling
Training data shape	{1: 193, 0: 26}	{0: 26, 1: 26}

As per the evaluation metrics defined in Chapter 3 Table 12 shows the results of metrics for all the models followed by Gain charts , Lift curves and K-S statistic charts.

Methods	**Accuracy**	**Precision**	**Recall**	**F1score**	**Specificity**	**Mis-classification**
Logistic regression	88%	78%	89%	83%	78%	12%
Decision Tree	82%	100%	80%	89%	100%	18%

Gradient Boosting	85%	72%	87%	79%	72%	15%
XGBoost	82%	89%	81%	85%	89%	18%
Balanced Bagging	84%	94%	82%	88%	94%	16%
Random Forests	86%	89%	86%	87%	89%	14%

Table 12: Metric results for train test split ratio of 60:40

Undersampling strategy for a split of 60:40 increased precision, recall, F-1 score and specificity considerably as compared to a training data without undersampling. However, there is a decrease in overall accuracy of the model.

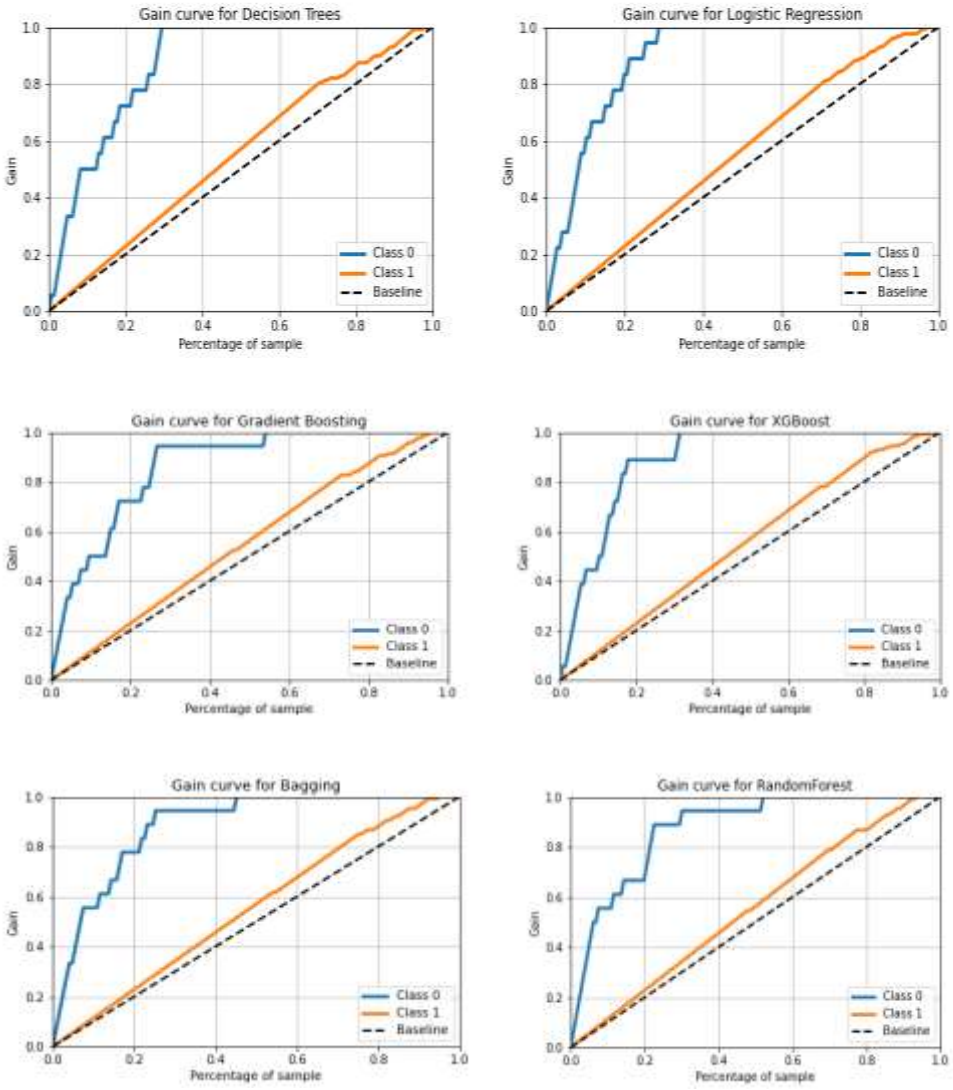


Figure 62: Gain curves for 60:40 training-testing data split for all the models

In all the gain charts for 60:40 split, XGBoost had the highest gain ~90% followed by Logistic regression ~85% for capturing patients whose survival status at a given time is alive, when targeting less than top 20% of the patient population in the testing set. Random forest performs well on all metrics and can be an appropriate choice for this split.

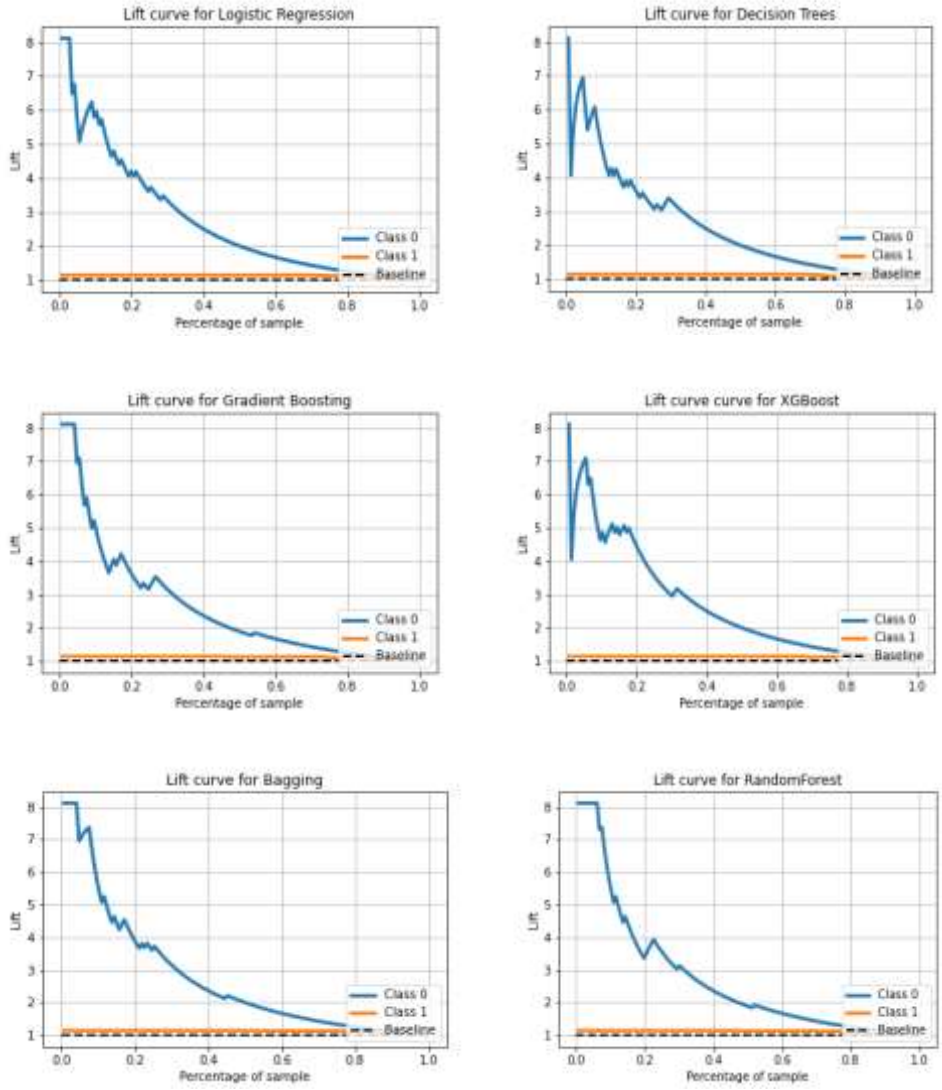


Figure 63: Lift curves for 60:40 training-testing data split for all the models

In all the lift charts for 60:40 split, XGBoost model gave a high lift of ~ 4.5 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models.

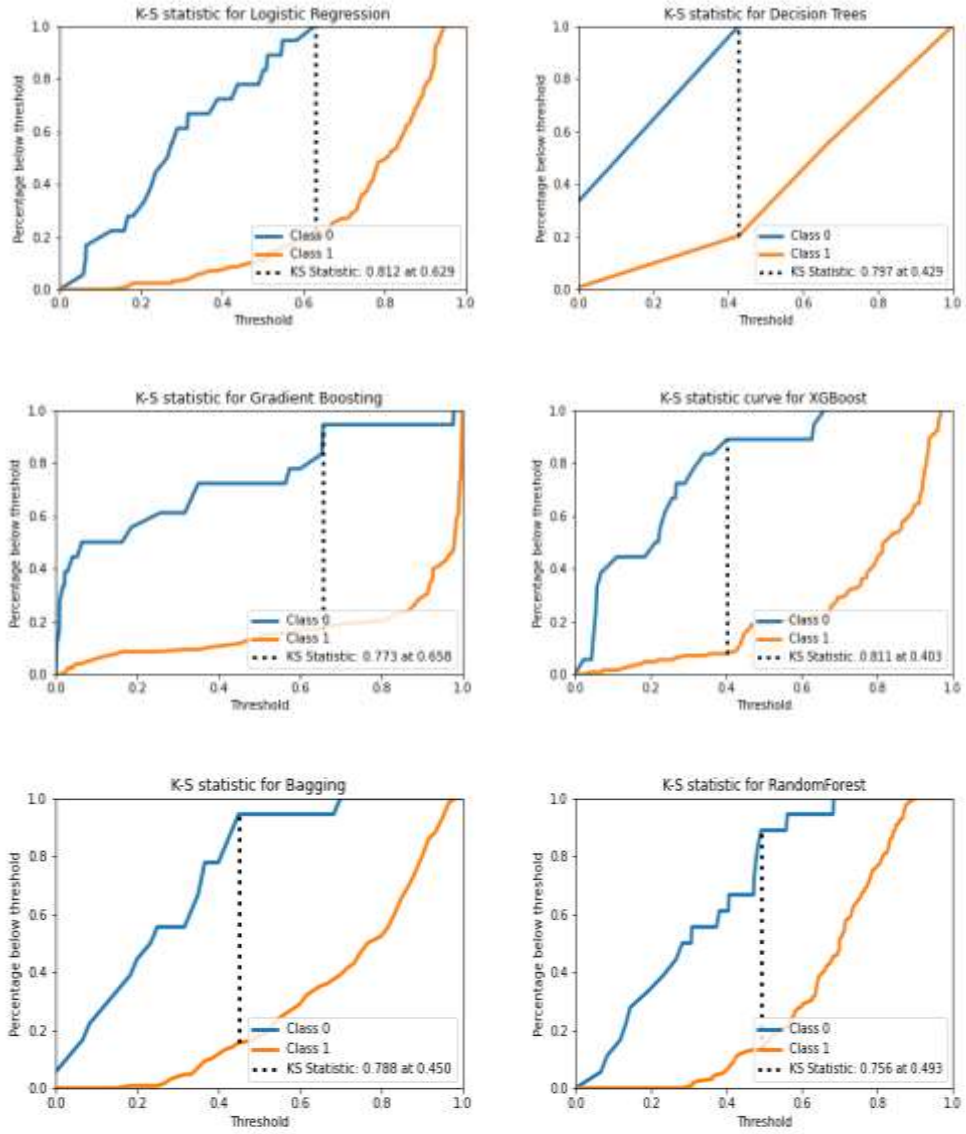


Figure 64: K-S curves for 60:40 training-testing data split for all the models

In all the K-S charts for 60:40 split, XGBoost and Logistic regression gave the maximum separation between both the classes at K-S statistic value of 0.81 and 0.812 at top 40% and top 62% of the population.

4.3.2.3 Train test split ratio 80:20

	Before undersampling	After undersampling
Training data shape	{1: 257, 0: 35}	{1: 35, 0: 35}

As per the evaluation metrics defined in Chapter 3, Table 13 shows the results of metrics for all the models followed by Gain charts , Lift curves and K-S statistic charts.

Methods	**Accuracy**	**Precision**	**Recall**	**F1score**	**Specificity**	**Mis-classification**
Logistic regression	85%	100%	83%	91%	100%	15%
Decision Tree	73%	100%	69%	81%	100%	27%
Gradient Boosting	81%	100%	78%	88%	100%	19%
XGBoost	85%	100%	83%	91%	100%	15%
Balanced Bagging	82%	100%	80%	89%	100%	18%
Random Forests	84%	100%	81%	90%	100%	16%

Table 13: Metric results for train test split ratio of 80:20

Undersampling the training data for a split of 80:20 yields major improvements in all the metrics except accuracy. The model is very confident in classifying both the classes.

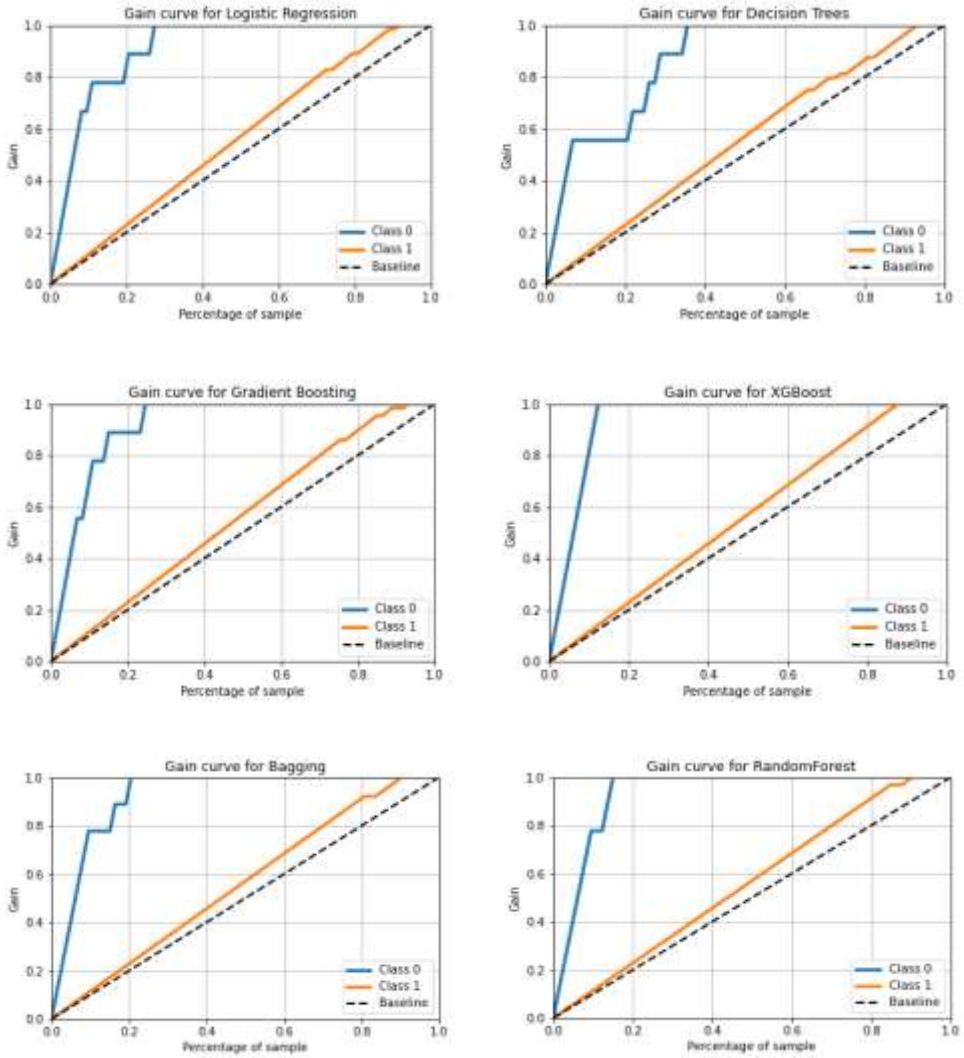


Figure 65: Gain curves for 80:20 training-testing data split for all the models

In all the gain charts for 80:20 split, XGBoost had the highest gain ~90% followed by Logistic regression ~85% for capturing patients whose survival status at a given time is alive, when targeting less than top 20% of the patient population in the testing set. Both the models are robust towards both the classes. However, XGBoost gives low accuracy as compared to a case with oversampling and no sampling. Hence undersampling may not be an appropriate

choice as in most cases as misclassification percentage has increased considerably when compared to oversampling and no sampling.

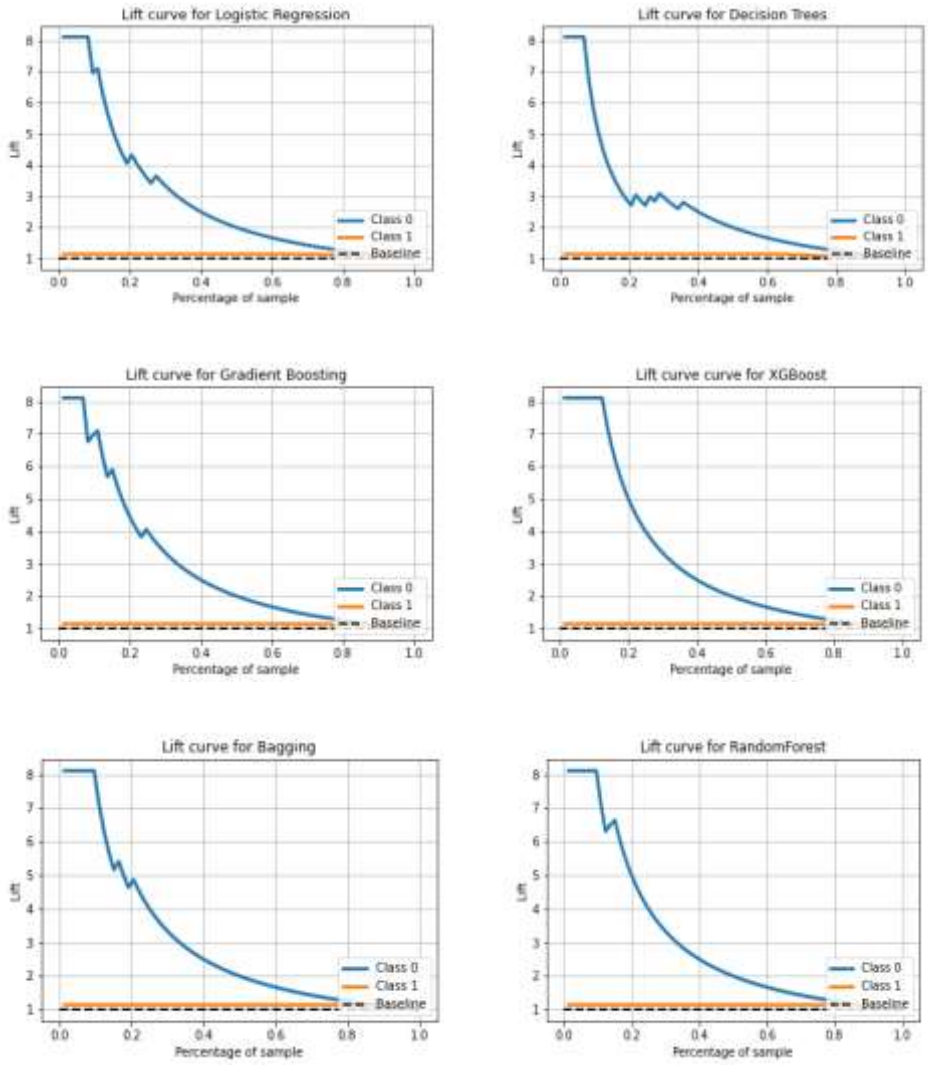


Figure 66: Lift curves for 80:20 training-testing data split for all the models

In all the lift charts for 80:20 split, XGBoost model gave a high lift of ~5 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models.

As discussed previously, XGBoost performs well on all metrics, but undersampling the dataset may not be the best solution for this split.

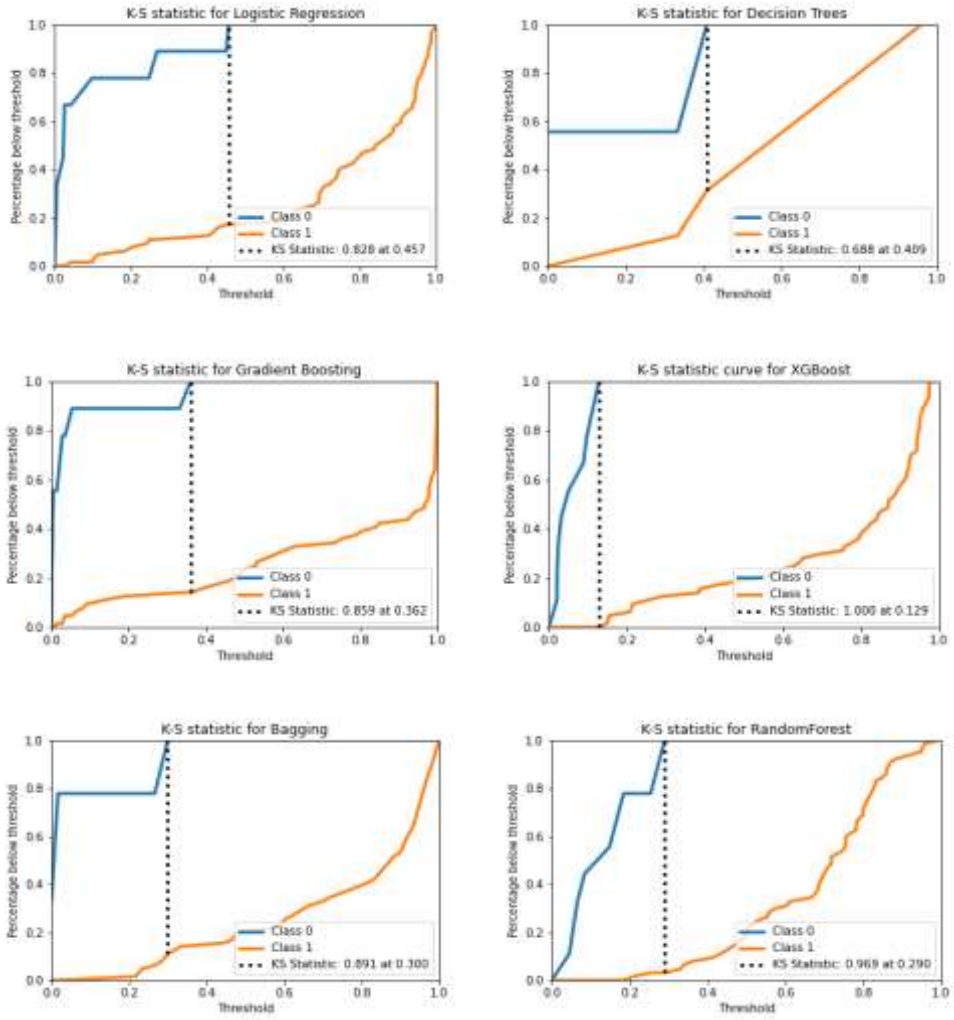


Figure 67: K-S statistic curves for 80:20 training-testing data split for all the models

In all the K-S charts for 80:20 split, XGBoost gave the maximum separation between both the classes at K-S statistic value of 1 at top 12.9% of the population.

4.3.2.4 Train test split ratio 90:10

	Before undersampling	After undersampling
Training data shape	{1: 288, 0: 40}	{1: 40, 0: 40}

As per the evaluation metrics defined in Chapter 3, we will measure the metrics for all the models in Table 14.

Methods	**Accuracy**	**Precision**	**Recall**	**F1score**	**Specificity**	**Mis-classification**
Logistic regression	89%	100%	88%	94%	100%	11%
Decision Tree	84%	100%	82%	90%	100%	16%
Gradient Boosting	86%	100%	85%	92%	100%	14%
XGBoost	92%	75%	94%	83%	75%	8%
Balanced Bagging	78%	100%	76%	86%	100%	22%
Random Forests	86%	100%	85%	92%	100%	14%

Table 14: Metric results for train test split ratio of 90:10

Undersampling for a split of 90:10 increased precision, F-1 score and specificity of all the models. However like other splits, resulted in lowered accuracy ultimately leading to high misclassification percentage.

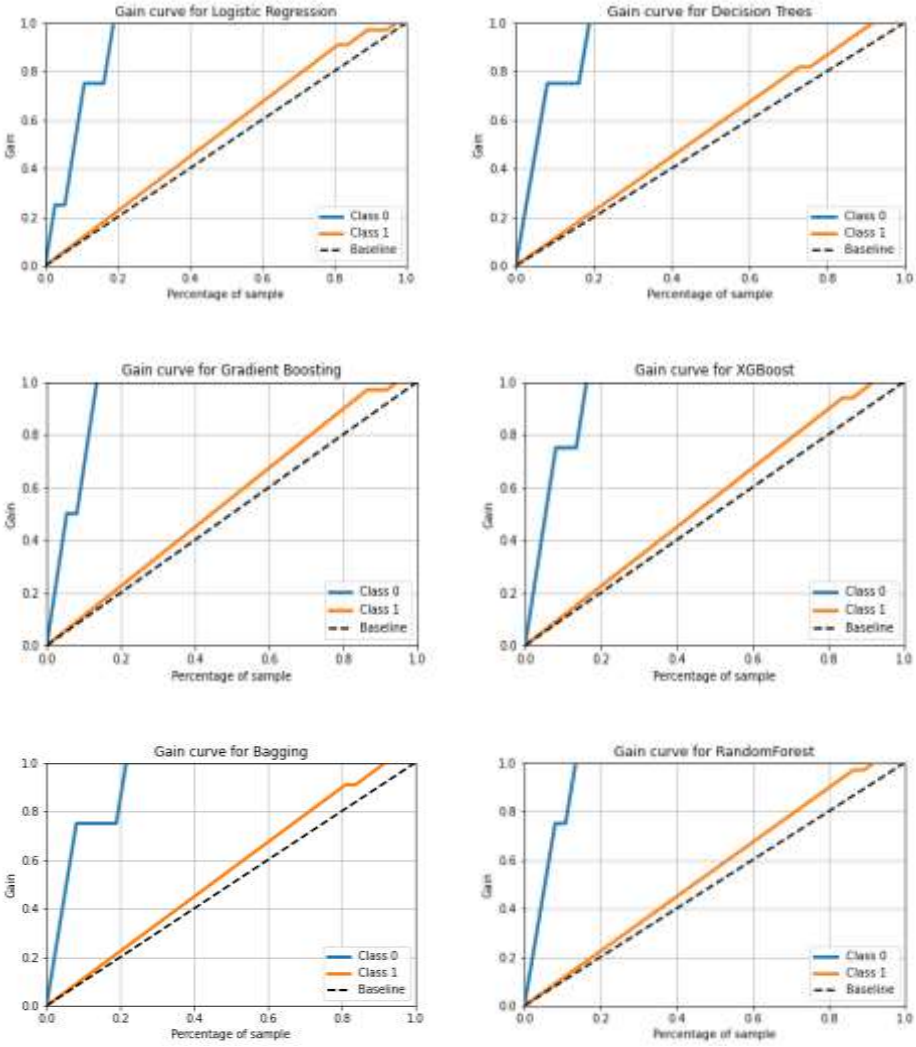


Figure 68: Gain curves for 90:10 training-testing data split for all the models

In all the gain charts for 90:10 split, all models except Bagging had the highest gain achieved when targeting less than top 20% of the patient population in the testing set. XGBoost had low specificity and precision as compared to other models i.e. 75% as compared to 100%. However, other models have low accuracy. Undersampling for a split of 90:10 case is better than no sampling for some models that are highly biased towards majority class i.e. Decision tree,

Gradient boosting and Random forests. On the other hand, choosing an appropriate solution between oversampling and undersampling is a trade-off between recall and precision.

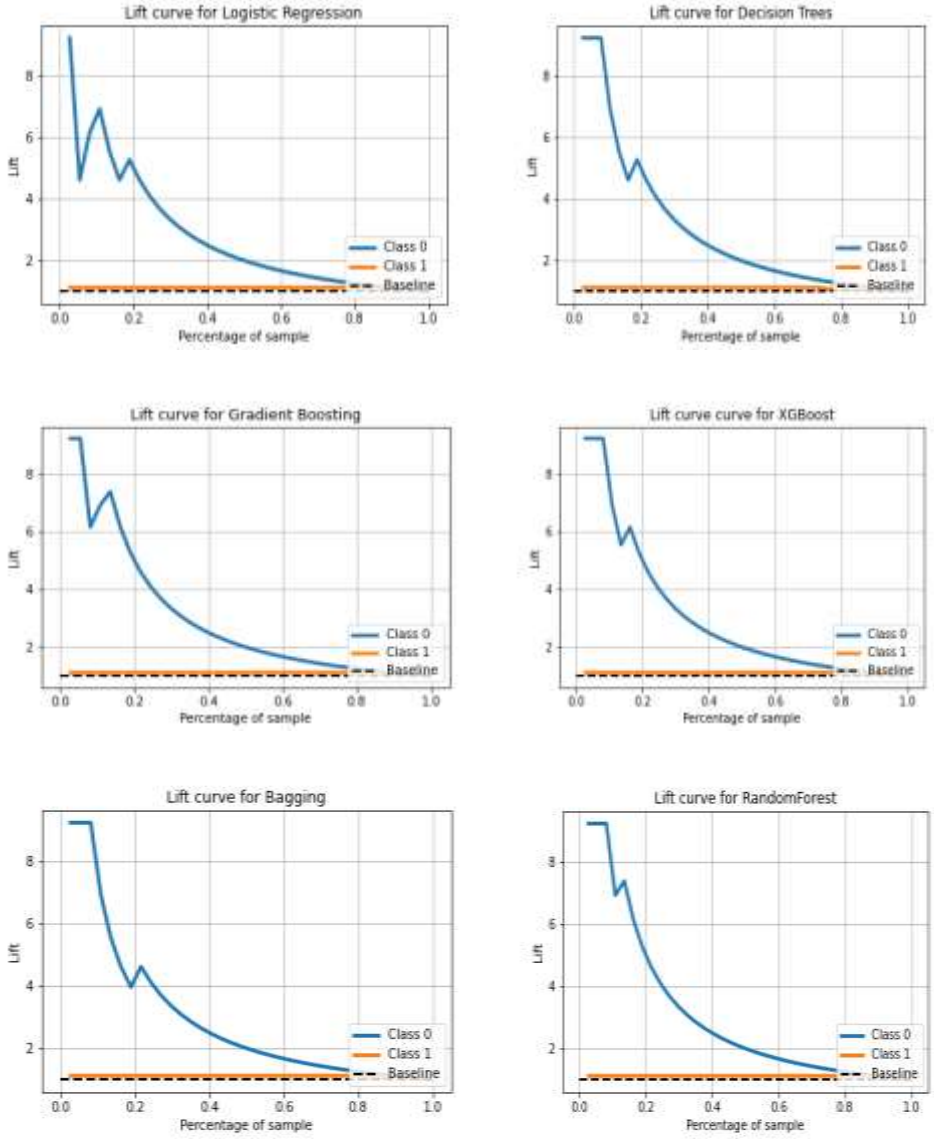


Figure 69: Lift curves for 90:10 training-testing data split for all the models

In all the lift charts for 90:10 split, all models except Bagging gave a high lift of greater than ~ 4.5 for capturing patients whose survival status at a given time is alive, when targeting top 20% of the patient population in the testing set as compared to other models. Balanced bagging also had the highest misclassification percentage.

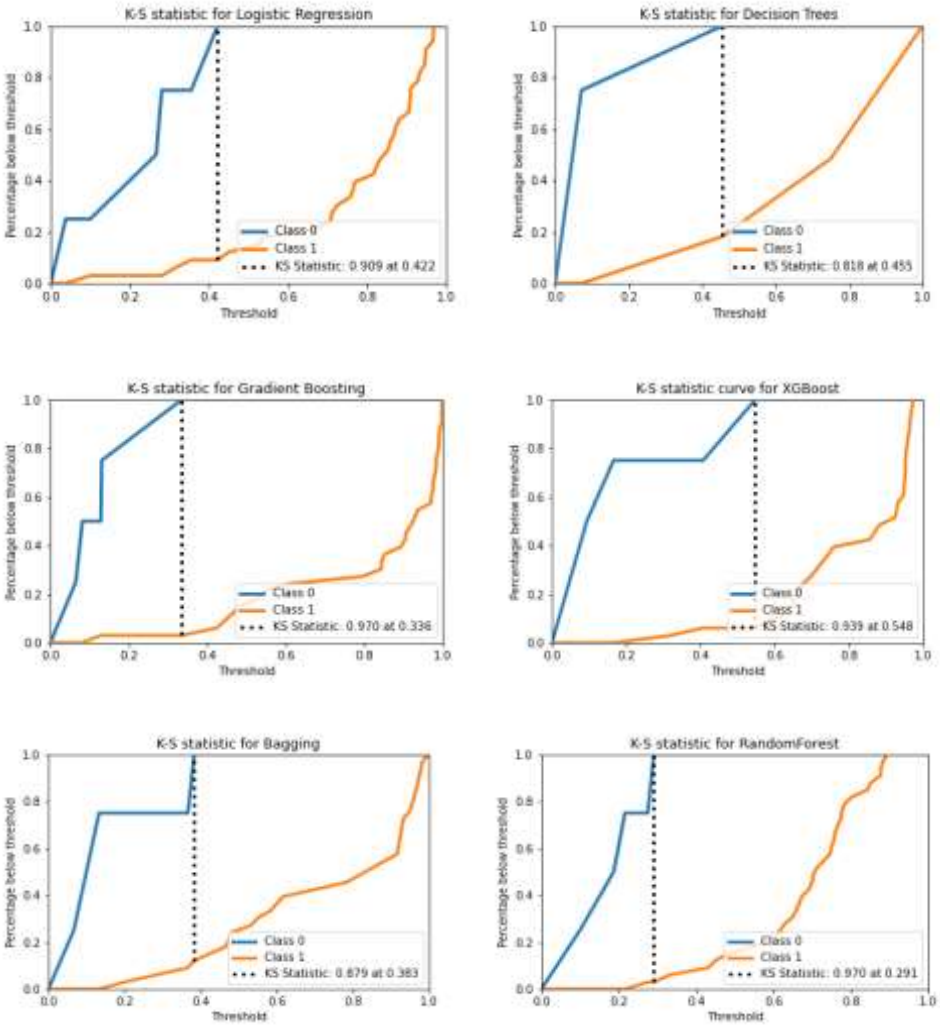


Figure 70: K-S statistic curves for 90:10 training-testing data split for all the models

In all the K-S charts for 90:10 split, Random Forest gave the maximum separation between both the classes at K-S statistic value of 0.97 at top 29% of the population.

Random forest and Logistic Regression are appropriate choices for a split of 90:10 undersampled training dataset.

4.4 Model performance on N-1 testing

In total there were 365 patient records, for N-1 testing model was trained on N-1 records and tested on 1 observation in a rotation.

Methods	**Overall Accuracy**	**Accuracy Class: Alive**	**Accuracy Class: Death**	**Mis-classification**
Logistic Regression	87.39%	84.09%	87.85%	12.61%
Decision Tree	94.24%	65.90%	98.13%	5.76%
Gradient Boosting	93.97%	59.09%	98.75%	6.03%
XGBoost	93.42%	63.63%	97.50%	6.58%
Balanced Bagging	84.10%	79.54%	84.73%	15.9%
Random Forests	93.42%	65.90%	97.19%	6.58%

Table 15: N-1 testing evaluation

Decision trees give highest accuracy of 94.24% overall as seen in Table 15. However, they are highly biased towards majority class giving low accuracy for minority class. A class weight balanced Logistic regression is highly robust towards minority class without performing any sampling.

4.5 Unsupervised learning

We have used K-means clustering for our evaluation in unsupervised learning. Since the dataset has two outcome of patients i.e. Alive and Death. We have used Elbow method and Silhouette analysis for measuring the performance of the model. This can help us determine if the dataset needs to be modelled in a way where we can introduce a third class where model is not able to make a decision.

4.5.1 Elbow method

Elbow method gives us an estimate of what a good *k* number of clusters would be based on the sum of squared distance (SSE) between data points and their clusters centroids. An appropriate value of k can be found at a position where SSE starts to flatten out and forming an elbow.

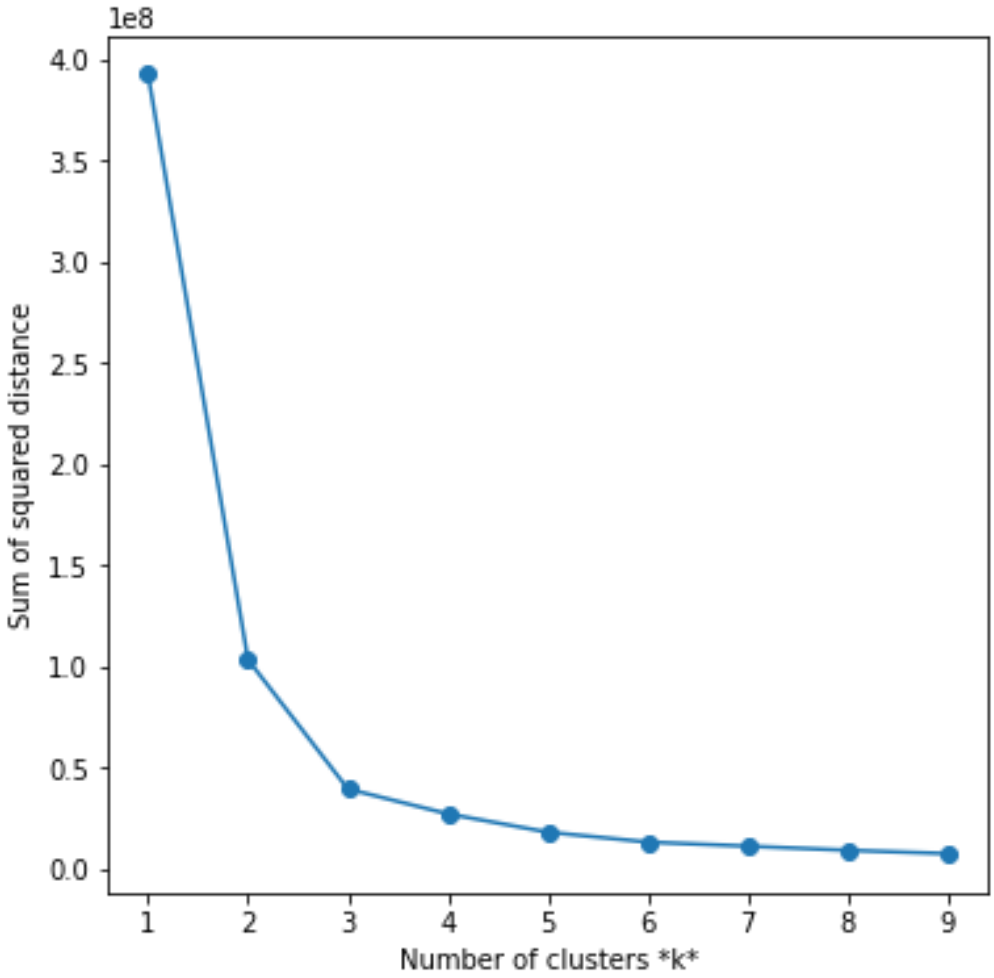
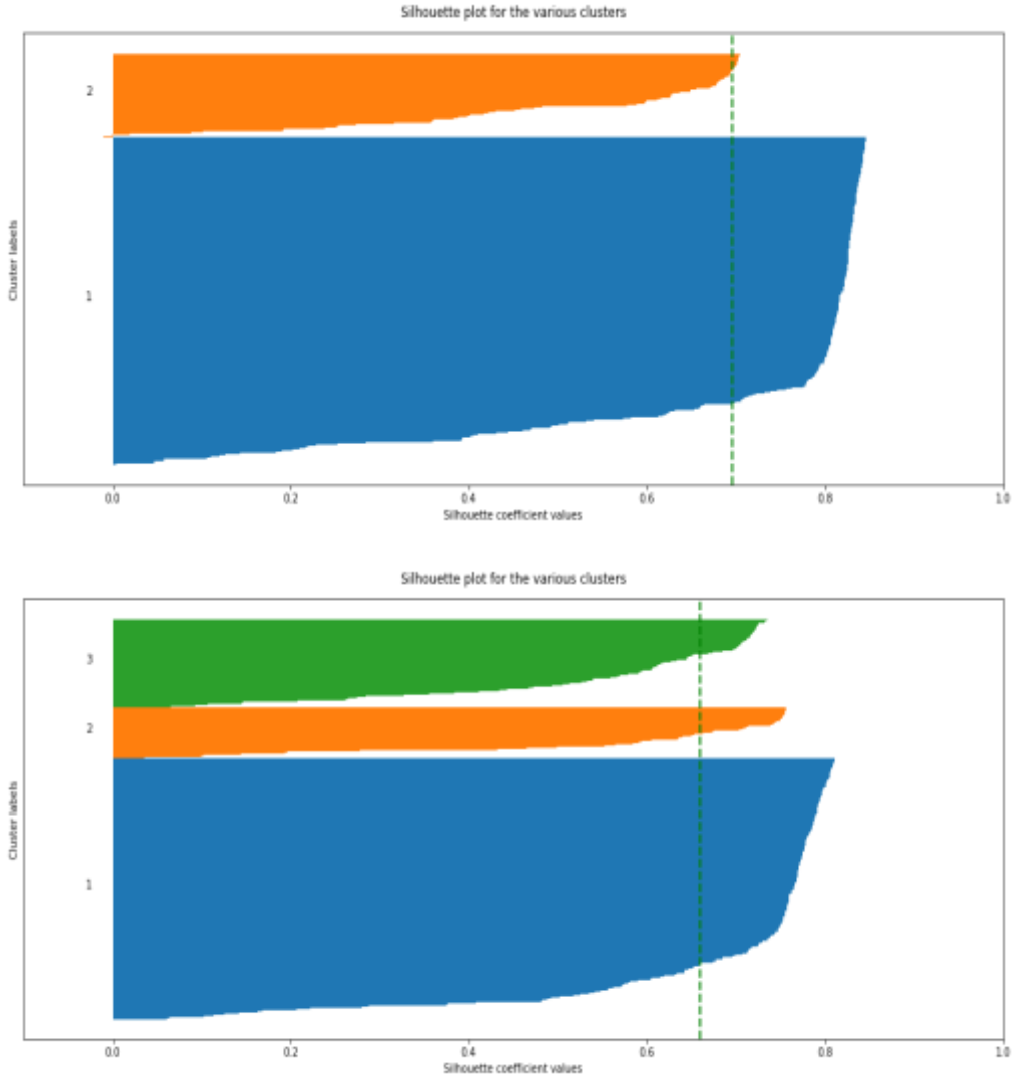


Figure 71: Number of clusters based on Sum of squared distance

The graph above shows that k=2 can be a good choice, but so can be 3. It is however hard to figure out a good number of clusters as the curve is monotonically decreasing and may not show any elbow or has an obvious point where the curve starts flattening.

4.5.2 Silhouette Analysis

We used silhouette analysis to determine the degree of separation between the clusters. The silhouette scores for different number of clusters are below as seen in Figure 72.



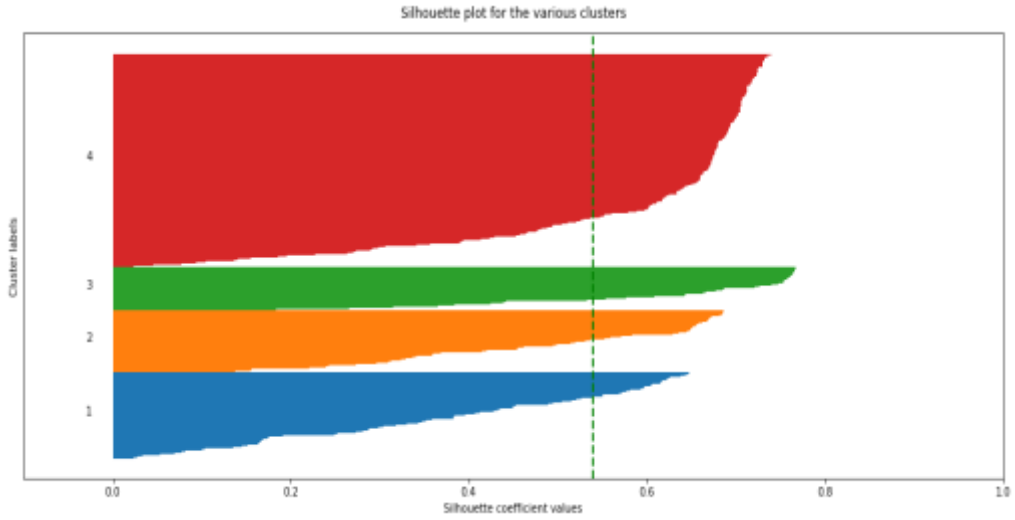


Figure 72: Silhouette plots for different number of clusters 2, 3 and 4

From the above plots, we can see having 2 clusters has the best average silhouette score of around 0.7. Also, the thickness of silhouette plot gives an indication of how much samples are present in the clusters. As we increased number of clusters to 3 and 4, the average silhouette score decreased to around 0.64 and 0.58 respectively. Moreover, the thickness of silhouette plot shows wide fluctuations. Appropriate number of clusters will have above 0.5 silhouette average score and all our clusters have higher than the average score.

Chapter 5
Conclusions

5.1 Research summary

The main objective of this research was to apply different machine learning methods and provide an analysis of approaches to predictive modelling when the nature of dataset is imbalanced. In NSCLC dataset, our objective was to highlight machine learning classifiers that perform well on all the evaluation metrics and are most robust towards minority classes. One of the classifiers used in our analysis that performed well and is robust towards class imbalance in data is XGBoost giving 99% accuracy and 89% specificity.

Generally, Decision trees are commonly used classifiers because of interpretability but are often susceptible to bias. From our analysis, they can be improved by performing sampling. In addition to performing sampling, one can also use ensemble classifiers which make the model more robust towards both the classes for an imbalanced dataset.

Furthermore, we highlighted the performance analysis after undersampling and oversampling. In our analysis, we saw using undersampling or oversampling gave a significant improvement in precision and specificity for models that are biased towards majority class. However, in undersampling misclassification percentage increased for most models.

In addition to that, we performed N-1 testing to measure performance for both the classes. Logistic regression gave highest accuracy 84.09% for minority class and 87.85% for majority class.

At last, we used unsupervised learning approach to measure if the dataset can be modelled differently for instances where model might be confused and to test if introducing another class can be a beneficial for such datasets. From silhouette scores we were able to see that having two clusters gave the best average score.

5.2 Future work

In this thesis, we propose different approaches for using machine learning classifiers in medical practice when the very nature of disease is such that majority of patient population are likely to be dead. Suggestions for future work is to train and test these algorithms on different detailed datasets with similar class imbalance. One example is to use datasets which include more feature columns such as smoking information, number of diagnostic tests and accurate information of the survival time. Another suggestion, is to use these datasets

and perform unsupervised learning for introducing a third class which indicates that the classifier is not sure on a particular record.

Bibliography

[Flow chart of gradient boosting]. (2020). Retrieved from geeksforgeeks: https://media.geeksforgeeks.org/wp-content/uploads/20200721214745/gradientboosting.PNG

Adam Yala, P. G.-L. (2021). Toward robust mammography-based models for breast cancer risk. *Science translational medicine, 13*(578).

Aerts, H. J. (2019). *NSCLC-Radiomics* . Retrieved from The Cancer Imaging Archive: https://doi.org/10.7937/K9/TCIA.2015.PF0M9REI

Camilla Mattiuzzi, G. L. (2019). Current Cancer Epidemiology. *Journal of Epidemiology and Global Health, 9*(4), 217–222.

Christelle Clément-Duchêne, C. C. (2010). How Accurate Are Physicians in the Prediction of Patient Survival in Advanced Lung Cancer? *The Oncologist, 15*(7), 782–789.

Christopher J. Kelly, A. K. (2019). Key challenges for delivering clinical impact with artificial intelligence. *BMC Medicine, 17*(1), 1-9.

Elyne Scheurwegs, B. C. (2017). Selecting relevant features from the electronic health record for clinical code prediction. *Journal of Biomedical Informatics, 74*, 92-103.

Food and Drug Administration. Proposed Regulatory Framework for Modifications to Artificial Intelligence/Machine Learning (AI/ML)-Based Software as a Medical Device (SaMD). (2019). Retrieved from FDA: https://www.regulations.gov/document?D=FDA-2019-N-1185-0001

Gulraiz Khan, A. S. (2019). *(Iteration diagram for bagging)[Diagram]*. Retrieved from IET Journals: https://ietresearch.onlinelibrary.wiley.com/cms/asset/0d45f150-52ee-47d0-87da-d216686758f3/ipr2bf02096-fig-0009-m.jpg

Harshad Hegde, N. S. (2019). MICE vs PPCA: Missing data imputation in healthcare. *Informatics in Medicine Unlocked, 17*.

Irene Chen, F. D. (2018). Why Is My Classifier Discriminatory? *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (pp. 3543-3554). Montreal: Curran Associates Inc.

Janus Christian Jakobsen, C. G. (2017). When and how should multiple imputation be used for handling missing data in randomised clinical

trials – a practical guide with flowcharts. *BMC Medical Research Methodology, 17*(1).

Jerzy Blaszczynski, J. S. (2017). Actively Balanced Bagging for Imbalanced Data. *Foundations of Intelligent Systems. 10352*, pp. 271-281. Springer International Publishing.

Joseph A. Cruz, D. S. (2007). Applications of Machine Learning in Cancer Prediction and Prognosis. *Cancer Informatics, 2*, 59-77.

Joshua C Mandel, D. A. (2016). SMART on FHIR: a standards-based, interoperable apps platform for electronic health records. *Journal of the American Medical Informatics Association, 23*(5), 899-908.

Kate Crawford, R. C. (2016). There is a blind spot in AI research. *Nature, 538*(7625), 311-313.

Kathryn C. Arbour, G. J. (2019). Systemic Therapy for Locally Advanced and Metastatic Non–Small Cell Lung Cancer: A Review. *JAMA, 322*(8), 764-774. doi:10.1001/jama.2019.11058

Kelly CJ, K. A. (2019 Oct). Key challenges for delivering clinical impact with artificial intelligence. *BMC Medicine*, 17(1).

Lever, J. K. (2017). Principal component analysis. *Nature methods*, 641-642.

Liang, H. T. (2019). Evaluation and accurate diagnoses of pediatric diseases using artificial intelligence. *Nature medicine*, 433–438.

Melissa J. Azur, E. A. (2011). Multiple imputation by chained equations: what is it and how does it work? *International journal methods of psychiatric research, 20*(1), 40-49.

Omer Gottesman, F. D.-w.-V. (2018). Evaluating reinforcement learning algorithms in observational health settings. *ArXiv, abs/1805.12298*, 1-16.

Pearse A. Keane, E. J. (2018). With an eye to AI and autonomous diagnosis. *npj Digital medicine, 1*.

Richard O. Duda, P. E. (2001). *Pattern classification.* New York: Wiley.

Rodvold DM, M. D. (2001). Introduction to artificial neural networks for physicians: taking the lid off the black box. *Prostate*, 39-44.

Somorjai RL, D. B. (2003). Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, caveats, cautions. *Bioinformatics*, 1484-91.

Wanyue Zhao, Y. G. (2020). Fast intelligent cell phenotyping for high-throughput optofluidic time-stretch microscopy based on the XGBoost algorithm. *Journal of biomedical optics, 25*(6), 1-12.

William R Hersh, M. G. (2013). Caveats for the use of operational electronic health record data in comparative effectiveness research. *Medical care, 51*(8 Suppl 3), S30-S37.