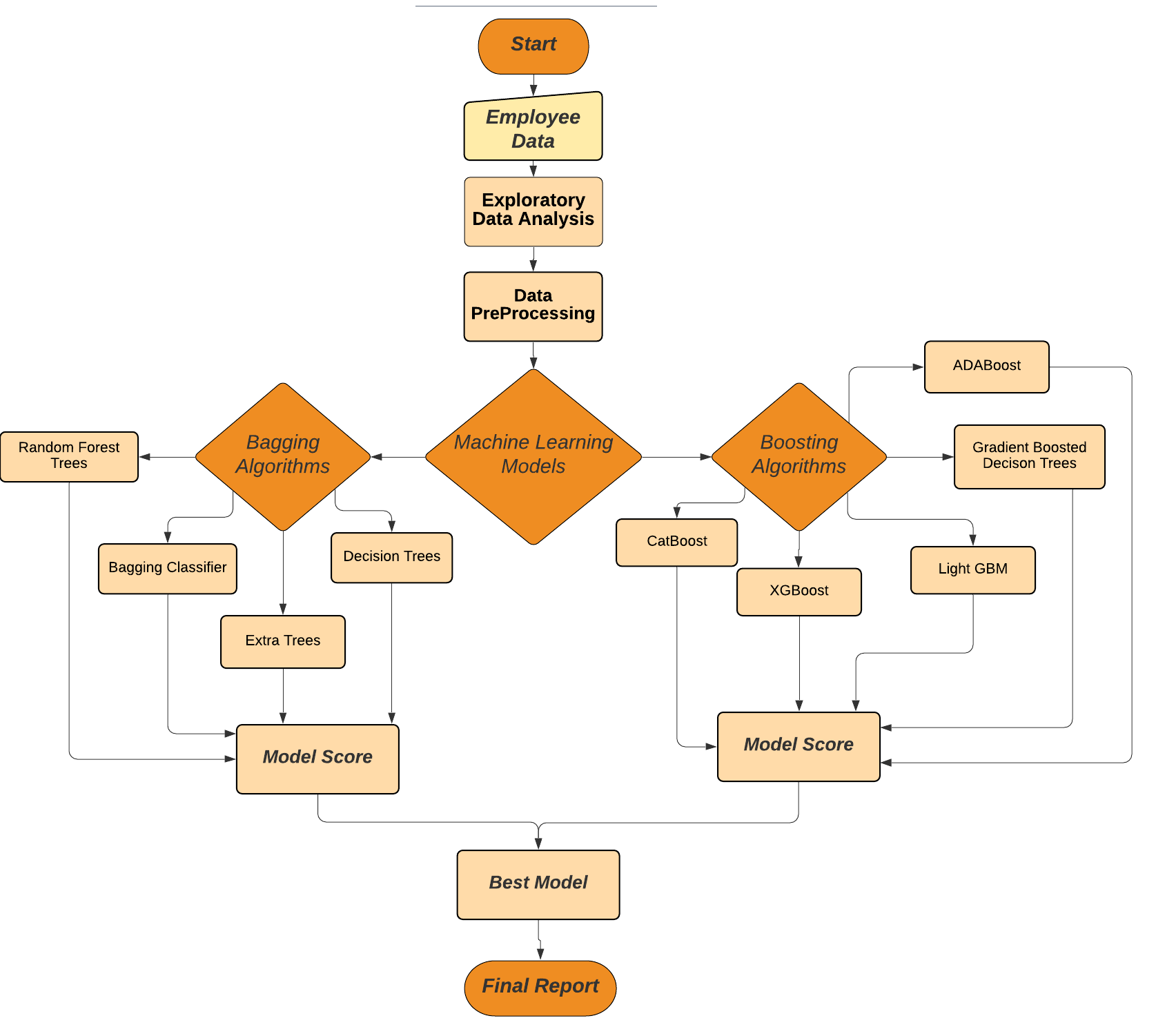
**Employee Attrition Analysis**

# Chapter 3: Methodology

The design flow of the application is shown in the figure below.



**Figure: Design Flow Chart**

(Source: Self-created)

The design starts by importing the employee data downloaded from kaggle.com into the python notebook environment in local machine. The libraries of python that are required for all the data processing model implementation and evaluation are imported into python notebook.

Then the task of performed an exploratory data analysis. This step it is checked for its structure, null/missing values, class balancing, and outliers. All these issues are treated in the step of data pre-processing, where data is processed to be clean and ready for the execution in the machine learning algorithms and encoding the data for the classification models. The first task in the data pre-processing step is to remove the missing/null values. This can be done in two ways by either removing the missing values or by adding a default value.

Then the task performed next is feature selection. This is an important step as the accuracy of the model depends a lot on the features that it is built on. The overfitting/underfitting of the models, which means in overfitting running the machine learning with too many features and some of them are insignificant. Underfitting is training the model with too few features that it is not able to perform with the right accuracy. This is treated by checking the features for their level of significance and the features that are best for training the model will be the only features kept in the data.

The classifications models work best on categorical data in form of numeric labels rather than string labels. Then encoding on the data is done, to convert such features so the model efficiency is improved. Encoding the data also helps in the fast processing of the models, when the data is huge model training part can run up to days.

Then the implementation of all the five models proposed namely: CatBoost, ADABoost, Gradient Boosted Decision Tree, Light GBM and XGBoost as boosting models, and Bagging Classifier, Decision Tree, Random Forest Trees and Extra Tree as Bagging Models. For every algorithm, saving the accuracy score, F1 score, precision and recall in a separate table so comparative analysis can be done effortlessly. The predictions made by the models are also compared with the actual values. The output result of each step is discussed in the next section of the finding analysis.

## Chapter 4: Finding Analysis

## 4. 1 Dataset Description

In this paper, There are three files having employee data general data, manager survey data and employee survey data. The general data has information regarding the employee and the target variable regarding the churn of the employee. The manager survey data consists of the information of manager reviews on the employee and similarly, employee survey data has information about the employee view and satisfaction level of the employee. All these factors of employee and manager viewpoints also play a vital role when it comes to deciding between staying or leaving the organisation. These files are then concatenated on the employee id as it was the common column in all files. The dataset that is used is consisting of shape 4410 x 29 where the number of rows is 4410 and 29 columns with the data of the employes. This data includes features as ' Age', ' Attrition', ' BusinessTravel', ' Department', ' DistanceFromHome ', ' JobLevel ', JobRole ', ' MaritalStatus ', ' MonthlyIncome ', 'NumCompaniesWorked ', ' PercentSalaryHike ', StandardHours ', ' TotalWorkingYears ', 'YearsAtCompany ', ‘YearsSinceLastPromotion’, ‘EnvironmentSatisfaction’, ‘JobSatisfaction’, ‘WorkLifeBalance’, ‘JobInvolvement’, ‘PerformanceRating’ as shown in the Table 1.

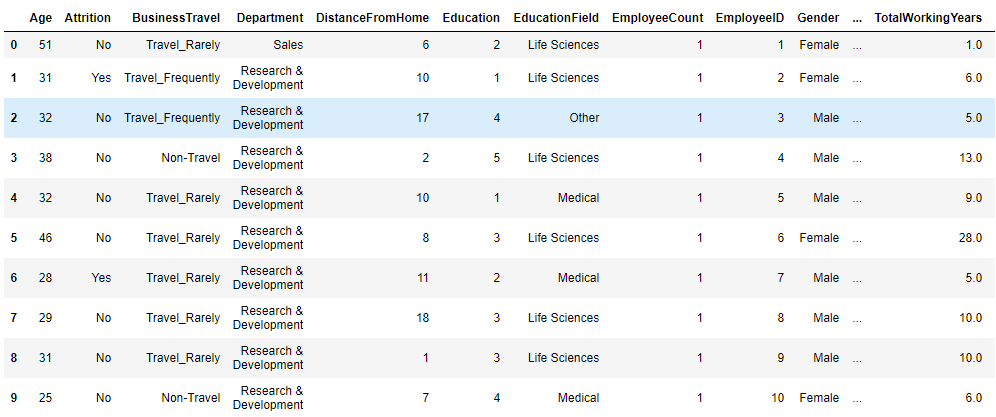
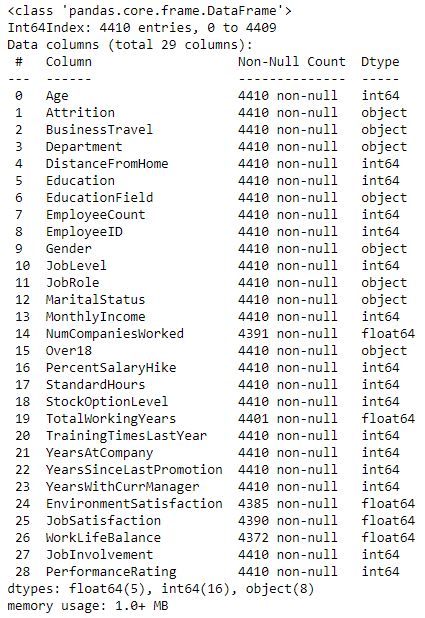


Table 1: Employee Data

(Source: Data file downloaded from kaggle.com)

## 4.2Exploratory Data Analysis

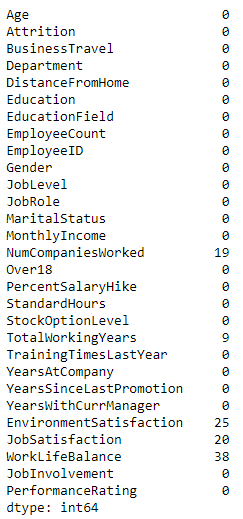
To do the data exploratory analysis, jupyter notebook is used in the local environment. Firstly, the information of the data is checked whereby the complete summary of data related to all column data type, range of data points, and null values. The Age, DistanceFromHome, Education, EmployeeCount, JobLevel, MonthlyIncome, PercentSalaryHike, StandardHours, StockOptionLevel, TrainingTimesLastYear, YearsAtCompany, YearsSinceLastPromotion, YearsWithCurrManager, JobInvolvement and PerformanceRating features are of type integer. The NumCompaniesWorked, TotalWorkingYears, EnvironmentSatisfaction, JobSatisfaction and WorkLifeBalance is of type float and all the remaining fields are of type object. In python string (or character) type is treated as type object. In the column Non-null count, some of the fields like NumCompaniesWorked, EnvironmentSatisfaction, JobSatisfaction and WorkLifeBalance have different values. This implies that the field has missing values present in the data.



**Table 2: Data information**

(Source: Code Output)

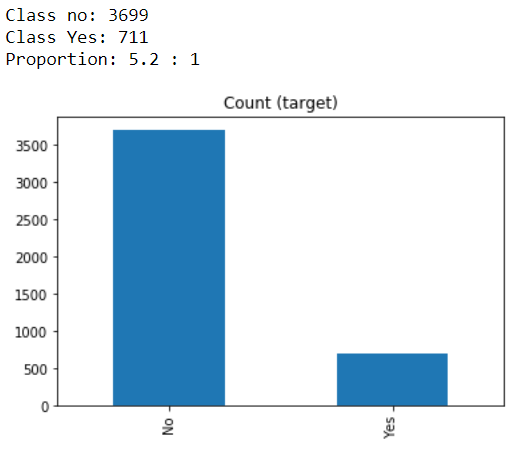
Then all the columns are checked for the exact count of missing values. Table 3 below shows the count.



**Table 3: Count of missing values**

(Source: Code Output)

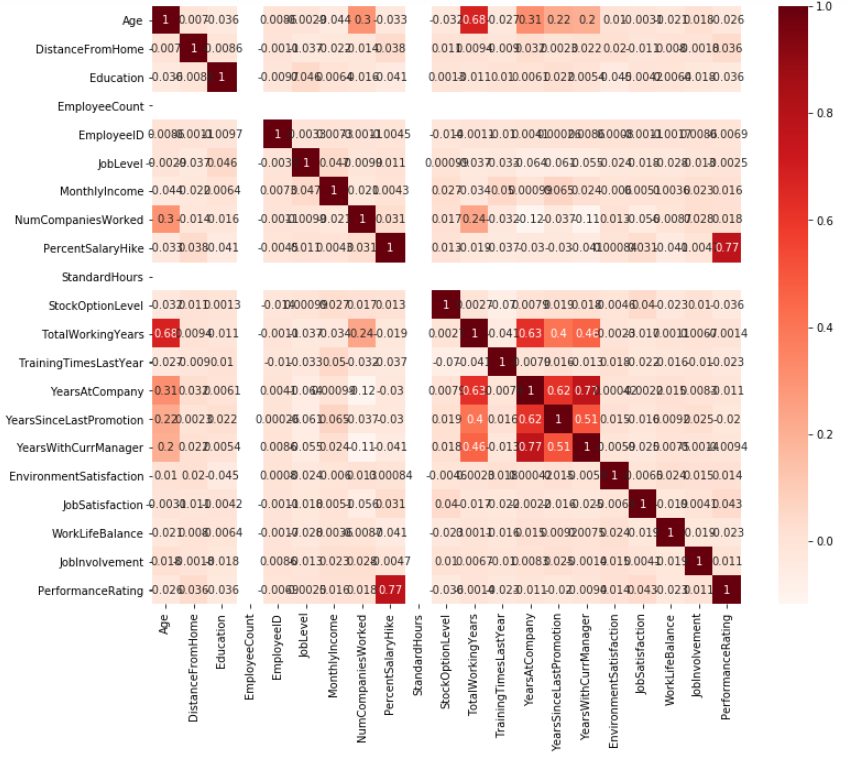
Then the target class Attrition is checked for the total number of values in each class. There are two classes which means this is the problem of binary classification. The classes are Yes and No. The number Yes values in the dataset are 711 and the total number of No values is 3699. This number is also plotted in the histogram as shown in figure below.



**Figure: Attrition target class value count**

(Source: Code Output)

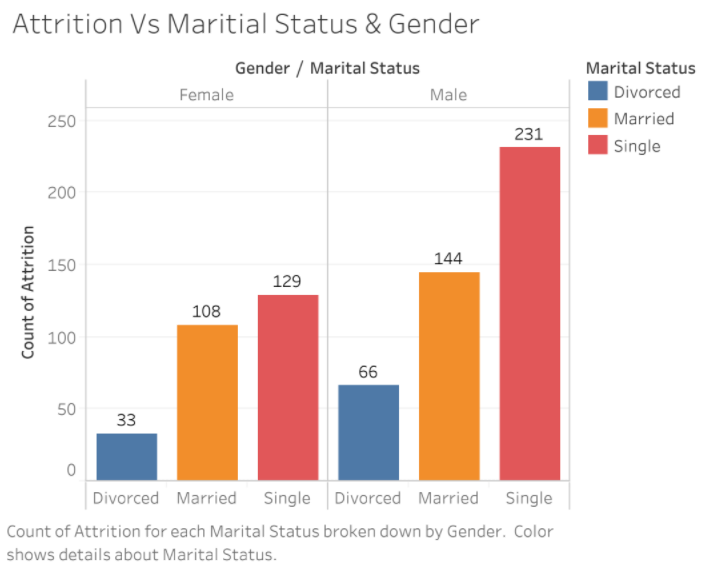
Then the Pearson correlation test is performed and the strength of the relationship of all the features is checked. This is then plotted in the seaborn heatmap as shown in the figure below. The figure shows the darker colour for the features which has a high correlation.



**Figure: Pearson Correlation test heatmap**

(Source: Code Output)

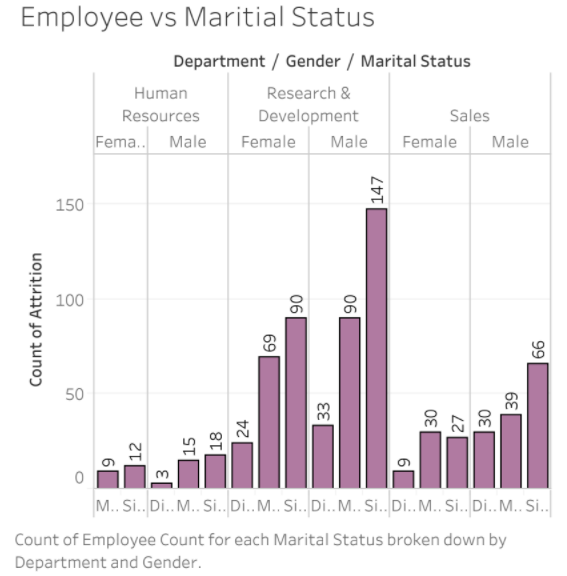
The exploratory data analysis is also performed on Tableau. It is a tool for generating insights about the data. Figure below show the attrition v/s marital status plot separated by gender. This shows that the male person who is single are churning more from the company, also that the female churn is not much affected by their marital status.



**Figure: Attrition v/s marital status**

(Source: Code Output)

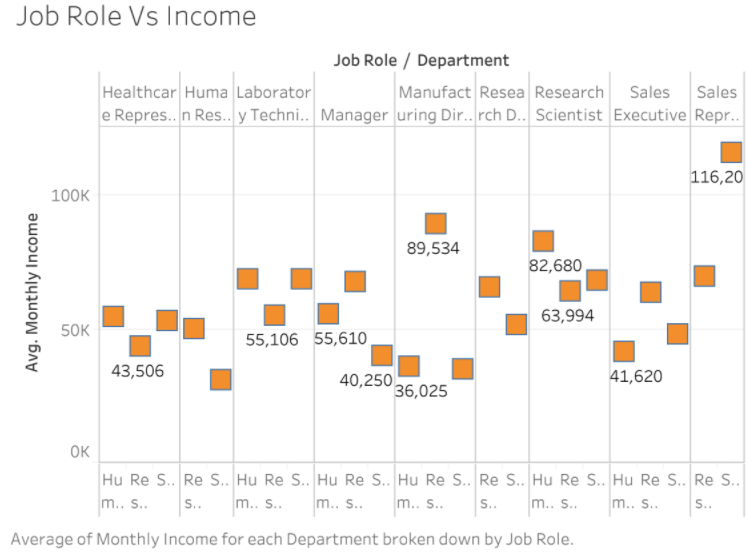
Figure below show the attrition v/s marital status plot separated by gender and the department. This shows that the male person who is single are churning more from the company, also that the female churn is not much affected by their marital status.



**Figure: Attrition vs/ marital status by gender**

(Source: Code Output)

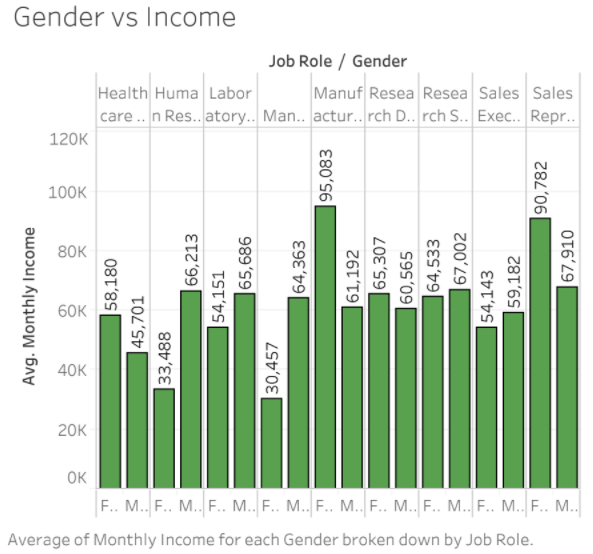
The figure below shows the plot of the salary earned by the employees with respect to their job role. It can be seen that job title/ department of Sales earns most than the rest of the employees.



**Figure: Jon role/ Department with Income**

(Source: Code Output)

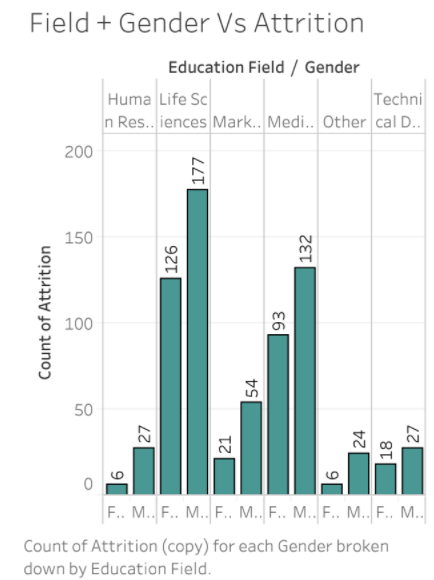
The figure below shows the plot of the salary earned by the employees with respect to their job role and gender. It can be seen that the female employees in most of the department are earning more than the male employees. This can be said about the organisation that they are not gender baised.



**Figure: Gender v/s Income plot**

(Source: Code Output)

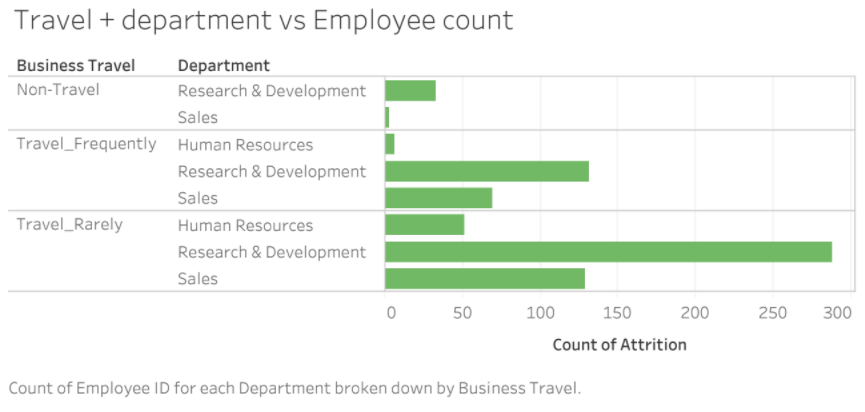
The figure below shows the plot of the count of the attrition employees with respect to their educational background and gender. It can be seen that Life Science based employees are churning more for both the genders but the count of male employees churning is still mopre for all the backgrounds.



**Figure: Attrition v/s education field / gender**

(Source: Code Output)

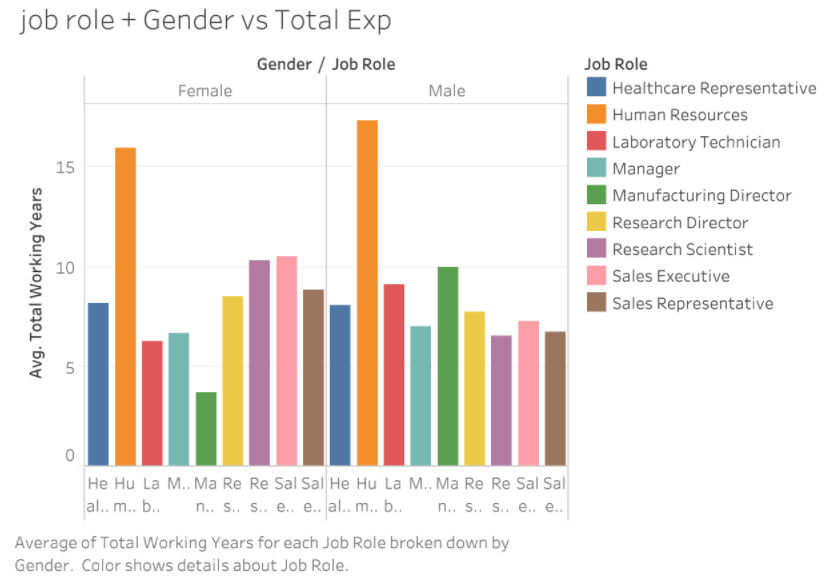
The figure below shows the plot of the attrition based on the travel history of the employees. It can be seen that the employees who rarely travel are more likely to churn. The count is maximum for Research and Development department.



**Figure: Pearson Correlation test heatmap**

(Source: Code Output)

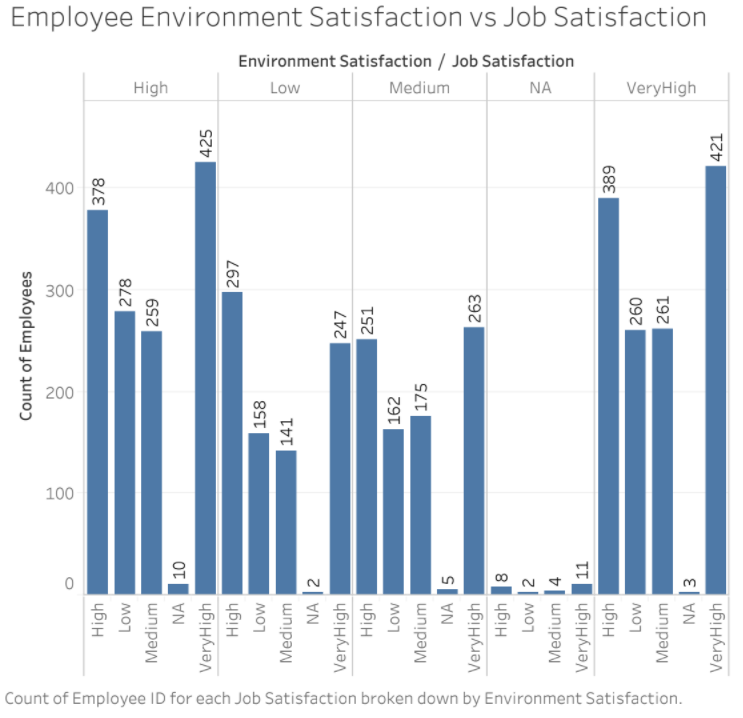
The figure below shows the plot of the total experience in years v/s job role and gender. It can be seen that for both the gender the Human Reseource Department has employees with maximum experience.



**Figure: Job role + gender v/s total experience**

(Source: Code Output)

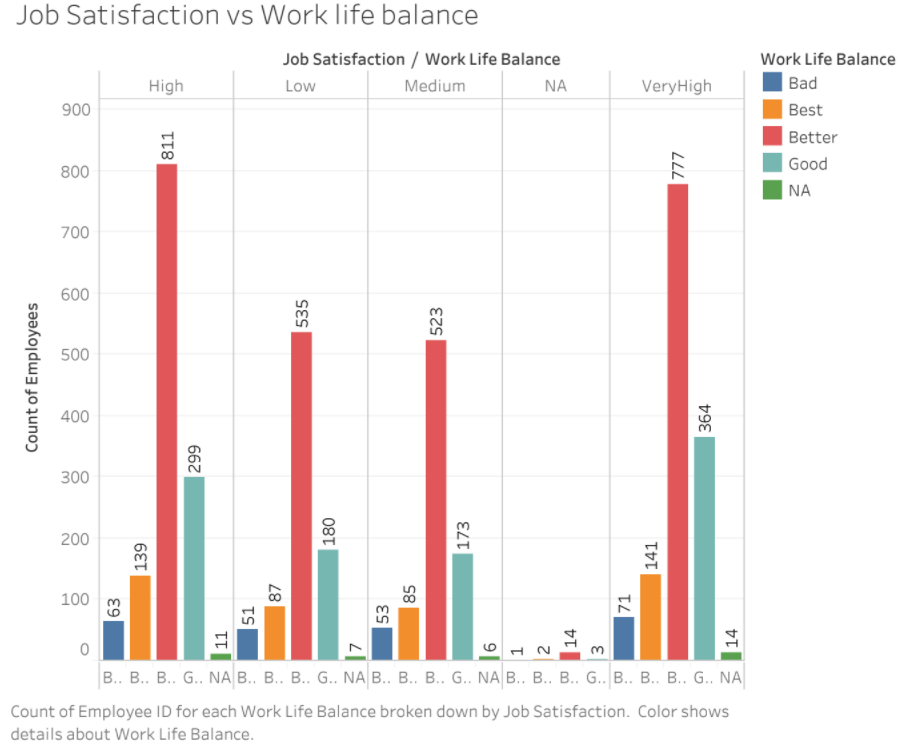
The figure below shows the plot of the employees organisation environment satisfaction with respect to their job satisfaction. It can be seen that most of the employees are satisfied.



**Figure: Employees environment satisfaction v/s job satisfaction**

(Source: Code Output)

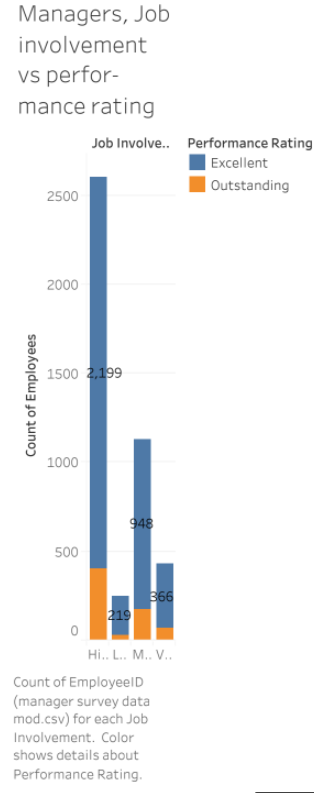
The figure below shows the plot of the employees work life balance with respect to their job satisfaction. It can be seen that there are a majority of employees who has a better work life balance.



**Figure: Job Satisfaction v/s work life balance**

(Source: Code Output)

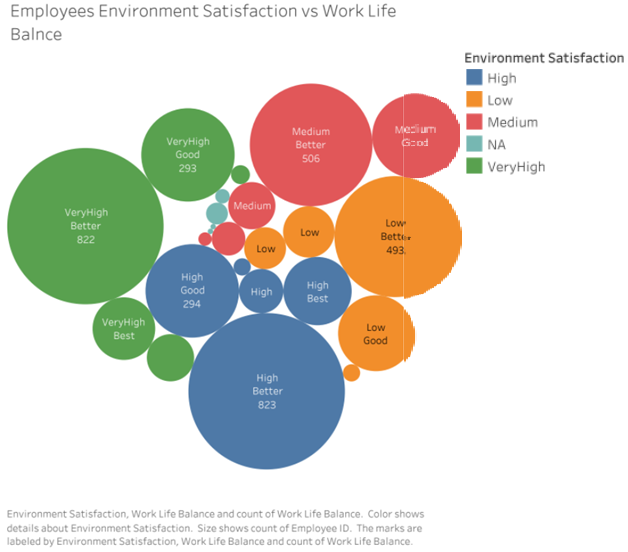
The figure below shows the plot of the managers, job involvement with respect to the employees performance rating. It can be seen that the performance rating of the employees is high when the managers are involved in the job.



**Figure: Managers, job involvement v/s performance rating**

(Source: Code Output)

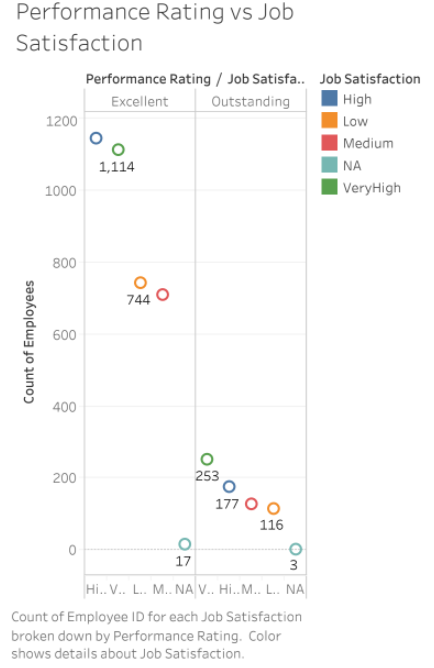
The figure below shows the plot of the employee environment satisfaction and work life balance. It can be seen that high better and very high better is the highest values in respect for satisfaction of the employees based on the work environment and work like balance.



**Figure: Employee environment satisfaction and work life balance**

(Source: Code Output)

The figure below shows the plot of the performance rating v/s job satisfaction. It can be seen that the employees performance ratings are much higher than their job satisafaction.



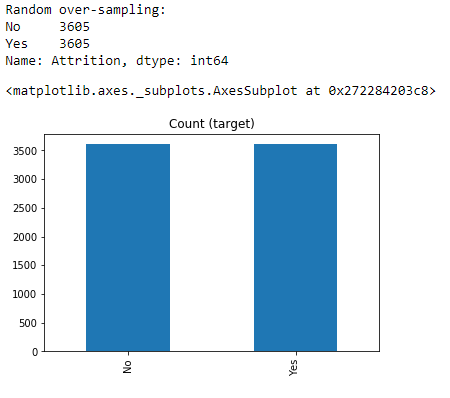
**Figure: Performance ratings v/s job satisfaction**

(Source: Code Output)

## 4.3 Data Pre-processing

The first step that is performed is data Pre-processing and it is also considered the most crucial phase in data analytics, where the unrefined information is cleaned and converted into an understandable format. Data cleaning, data enrichment, and data normalization is part of data pre-processing techniques. Table 3 above in section 4.2 Exploratory Data Analysis shows the total number of missing values in each feature set. WorkLifeBalance has the maximum number of missing columns 38 followed by the EnvironmentSatisfaction, JobSatisfaction and NumCompaniesWorked. As treatment of these values can’t be by filling the missing values with some default mean or median, so such rows are dropped. This is because if they are filled with any default values data credibility will be lost.

Then class balancing is performed to balance out the Attrition class Yes and No. To balance the classes, Random oversampling method is used. There are two types of classes in our data Yes which has a lower count of values 711 hence Yes is minority class and No is the majority class. In random oversampling the data from the majority class is used to populate the data with new entries in the minority class. This will generate an equal count of the values in both the classes. The data bar plot after the oversampling is shown below in Figure.



**Figure: Class values count after Random Oversampling**

(Source: Code Output)

Then performed label encoding using LabelEncoder in Python. In this step, converted all the values that are qualitative into quantitative. The data values of type string are encoded into their numeric form For example taking Attrition as a feature, encode No as 0 and Yes as 1. This is done by the label encoder and can then also decode these to test for the actual and observed values by each value, for comparing them as well. Table 4 below shows the label encoded data after encoding Attrition feature.

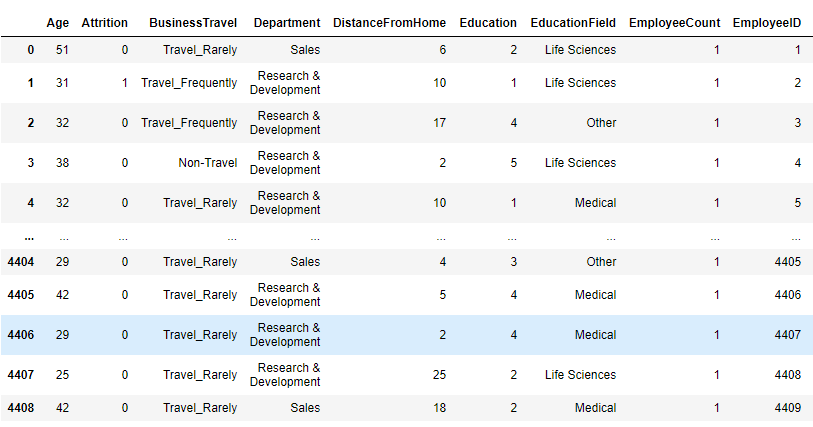


Table 4: Label encoded data values

(Source: Code Output)

For this implementation, we created two label encoders one for the target Attrition and other for the feature set. For this, the data column names are first listed to find the columns which are of type object. Then these column names are passed through the for loop and all columns are label encoded. The loop is used as the label encoder is a scalar function, will take one values one time. Table 5 below shows the data information of the data after label encoding. The data types are integer and float only.

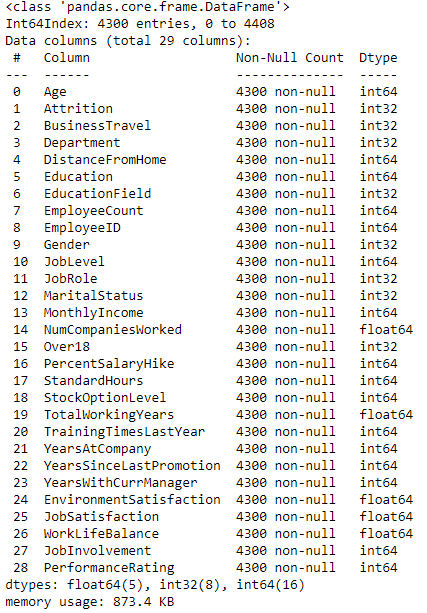


Table 5: Data information on Label encoded data

(Source: Code Output)

The nest step that is performed is feature selection. In this data, there are 29 features. But all of these features may not be significant to help predict the target. The feature selection method applied in this research is the Recursive Feature Selection. This method works by removing n non-significant feature in each recursion. The method works by running a regression model in each loop with the data. After this process, there were 15 significant features.

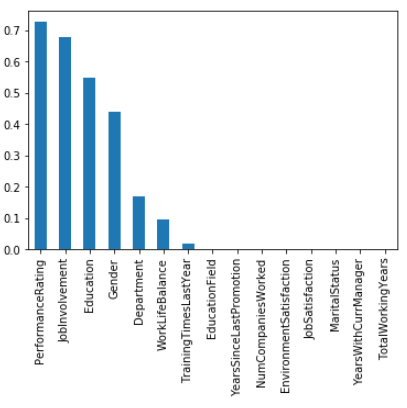
The selected features are then run through a hypothesis testing process called Chi-Square test. A chi-square is a goodness of fit test which validates if that the subset of the data with chosen features matches the entire population. A chi-square test is also a test of independence which compares two variables in a contingency table to check their relationship strength. It show that relationship strength which signifies the change in one variable will affect the change in another. It works by testing the null hypothesis that there is no relationship present among the two variables. If the probability is less that 0.05, there is not enough evidence to support the null hypothesis. This will lead to approve the alternate hypothesis that the variables are significant. The Figure shows the chi-square test results, for the features selected by the PCA.



Table 6: Feature score and probability

(Source: Code Output)

The features probability plot is shown in figure below.



**Figure: Probability plot for Chi-square test**

(Source: Code Output)

The features with probability score less than 0.05 are kept. The table 7 below shows the data information of the data after removing non-significant columns. There are 9 significant features.

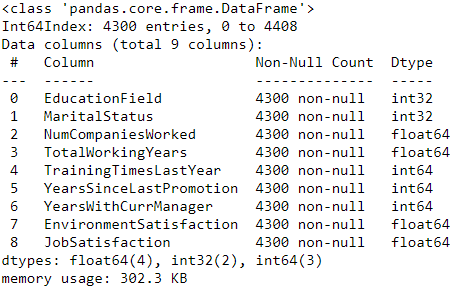
****

Table 6: Data information on Feature selected data

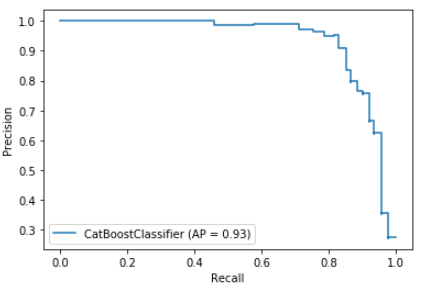
(Source: Code Output)

**4.4 Machine Learning Algorithms**

To implement the machine learning algorithms on data, it is first divided into training and test set. This is done so the training of the model, as well as evaluation of the model for its accuracy, can be done. Accuracy, F1-score, precision and recall are the evaluation metric used to validate model performance. For all the models, accuracy score, F1-score, precision and recall score are stored on a blank pandas data frame for comparative analysis. This paper implements bagging and boosting methods based on machine learning algorithms. The bagging algorithms are the algorithms which consider homogeneous weak learners, learns them independently from each other in parallel and combines them following some kind of deterministic averaging process.

The bagging algorithms are the algorithms which consider homogeneous weak learners, learns them sequentially in a very adaptative way (a base model depends on the previous ones) and combines them following a deterministic strategy.

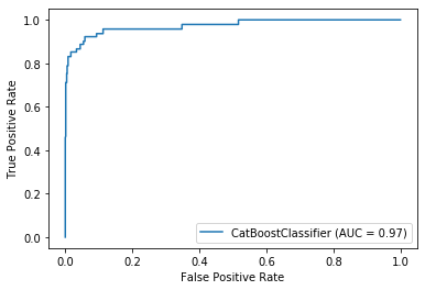
First implemented the CatBoost, Boosting model on the data. CatBoost algorithm is a boosting algorithm for the categorical data. CatBoost comes from Category and Boosting. The model is generated for a learning rate of 0.85 and runs the data in 100 iterations. The learning rate is the ensemble weight on the weak learner features. The model generated the accuracy score of 94.65, F1-Score of 80.83, Precision score of 98.98 and recall rate of 6830.64. The figure below shows the precision-recall curve of CatBoost. A model with perfect skill is depicted as a point at a coordinate of (1,1). A skilful model is represented by a curve that bows towards a coordinate of (1,1). A no-skill classifier will be a horizontal line on the plot with a precision that is proportional to the number of positive examples in the dataset. Catboost model is skilful model.



**Figure: Precision Recall curve of CatBoost**

(Source: Code Output)

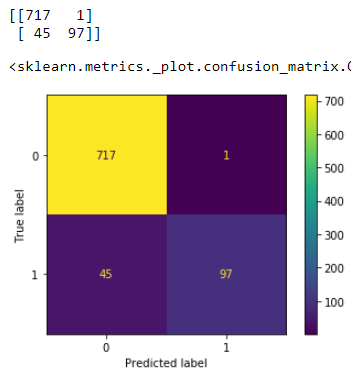
Figure below shows the ROC curve of Catboost Model. A classifier that has no discriminative power between positive and negative classes will form a diagonal line between a False Positive Rate of 0 and a True Positive Rate of 0 (coordinate (0,0) or predict all negative class) to a False Positive Rate of 1 and a True Positive Rate of 1 (coordinate (1,1) or predict all positive class). Models represented by points below this line have worse than no skill. A model with perfect skill is represented at a point (0,1). A model with perfect skill is represented by a line that travels from the bottom left of the plot to the top left and then across the top to the top right. The ROC curve shows that the model has perfect skill. The area under the curve can be calculated to give a single score for a classifier model across all threshold values. This is called the ROC area under curve or ROC AUC or sometimes ROCAUC. The score is a value between 0.0 and 1.0 for a perfect classifier. The model has AUC of 0.97 which implies that the model is perfect.



**Figure: ROC curve of CatBoost**

(Source: Code Output)

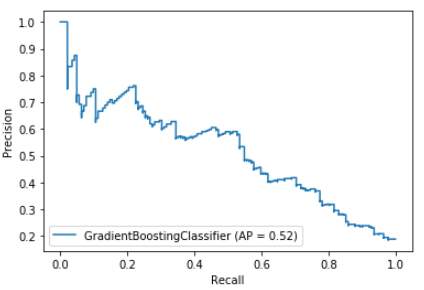
The figure below shows the confusion matrix of the model. The values in confusion matrix are used to generate all the above metrics. The True Positive of the model is 717, False Negative values is 45, False Positive values is 1 and True Negative of the model is 97. This shows that the model is making more true predictions.



**Figure: Confusion matrix and plot of CatBoost**

(Source: Code Output)

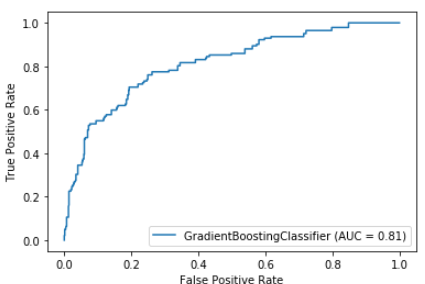
Next implemented the Gradient Boosting model on the data. Gradient boosting algorithm is a boosting algorithm of the decision trees by adding an ensemble. The model is generated for a default initializations. The model generated the accuracy score of 86.04, F1-Score of 34.78, Precision score of 76.19 and recall rate of 22.53. The values for F1 score and recall are too low which indicated that model is not performing efficiently. Figure below show the precision recall curve of Gradient Boosting. Gradient Boosting model is a no skill model as though it’s a curve not a horizontal but it’s not curving towards (1,1).



**Figure: Precision Recall curve of Gradient Boosting**

(Source: Code Output)

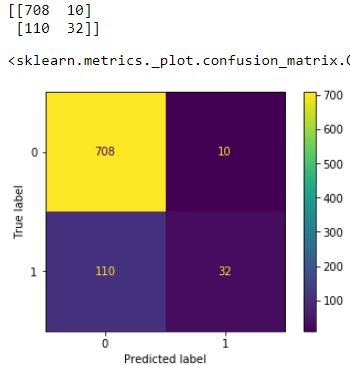
Figure below shows the ROC curve of Gradient Boosting Model. The ROC curve shows that the model has perfect skill. The model has AUC of 0.81 which implies that the model is perfect.



**Figure: ROC curve of Gradient Boosting**

(Source: Code Output)

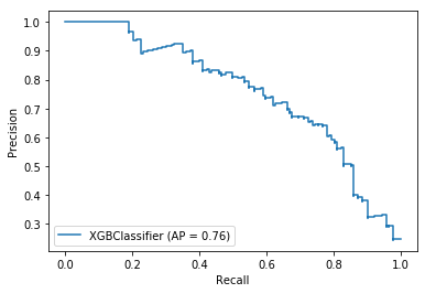
The figure below shows the confusion matrix of the model. The values in confusion matrix are used to generate all the above metrics. The True Positive of the model is 708, False Negative values is 110, False Positive values is 10 and True Negative of the model is 32. This shows that the model is making more negative predictions, and is predicting just a few positive predictions. This shows that the model is making more true predictions.



**Figure: Confusion matrix and plot of Gradient Boosting**

(Source: Code Output)

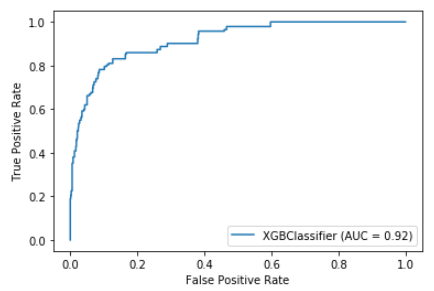
## Next implemented the XG Boost, Boosting model on the data. XG Boost algorithm is a boosting algorithm based on gradient boosting having a high performance and speed. The model generated the accuracy score of 88.93, F1-Score of 54.11, Precision score of 86.15 and recall rate of 39.43. Figure below show the precision recall curve of XG Boost. XG Boost model is skillfull model.



**Figure: Precision Recall curve of XG Boost**

(Source: Code Output)

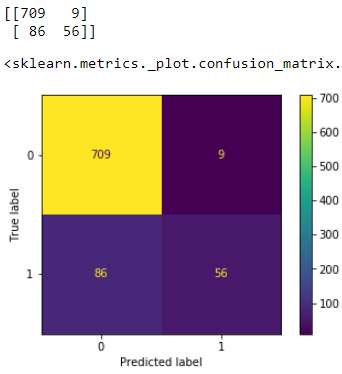
Figure below shows the ROC curve of **XG Boost** Model. The ROC curve shows that the model has perfect skill. The model has AUC of 0.92 which implies that the model is perfect.



**Figure: ROC curve of XG Boost**

(Source: Code Output)

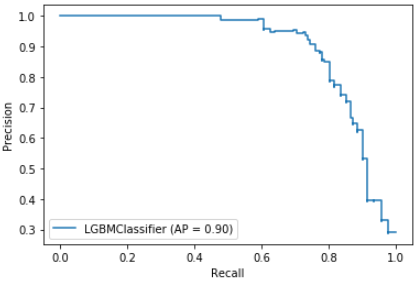
The figure below shows the confusion matrix of the model. The values in confusion matrix are used to generate all the above metrics. The True Positive of the model is 709, False Negative values is 86, False Positive values is 9 and True Negative of the model is 56. This shows that the model is making more negative predictions, and is predicting just a few positive predictions.



**Figure: Confusion matrix and plot of CatBoost**

(Source: Code Output)

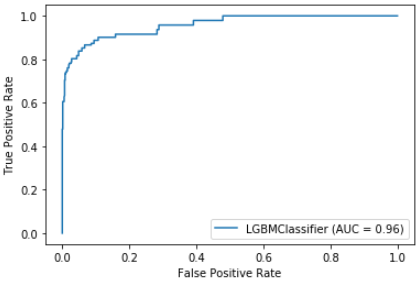
Next implemented the Light GBM, Boosting model on the data. Light GBM algorithm is a boosting algorithm based on gradient boosting having a high performance and speed. It speed is a lot faster than the other versions of gradient boosted machine. The model is generated for default initializations. The model generated the accuracy score of 94.07, F1-Score of 79.01, Precision score of 95.05 and recall rate of 67.6. Figure below show the precision recall curve of Light GBM. Light GBM model is no skill model as the curve is moving toward (0, 0).



**Figure: Precision Recall curve of Light GBM**

(Source: Code Output)

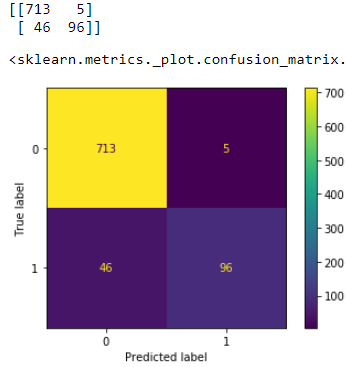
Figure below shows the ROC curve of **Light GBM** Model. The ROC curve shows that the model has perfect skill. The model has AUC of 0.96 which implies that the model is perfect.



**Figure: ROC curve of Light GBM**

(Source: Code Output)

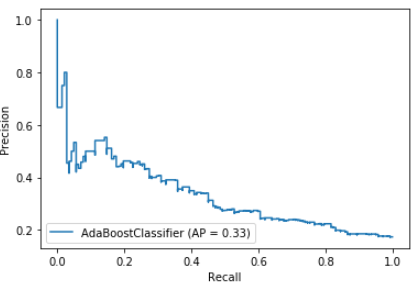
The figure below shows the confusion matrix of the model. The values in confusion matrix are used to generate all the above metrics. The True Positive of the model is 713, False Negative values is 46, False Positive values is 5 and True Negative of the model is 96. This shows that the model is making more negative predictions, and is predicting just a few positive predictions.



**Figure: Confusion matrix and plot of Light GBM**

(Source: Code Output)

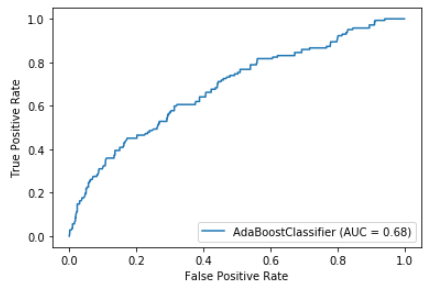
Next implemented the ADABoost, Boosting model on the data. ADABoost algorithm is a meta learning based boosting algorithm for the binary classification. ADABoost iterartively work to convert the weak learners to strong. The model is generated for decision tree of maximum depth of 54 with SAMME algorithm and 200 estimators. The model generated the accuracy score of 83.14, F1-Score of 12.12, Precision score of 43.48 and recall rate of 7.04. Figure below show the precision recall curve of ADABoost. ADABoost model is no skill model.



**Figure: Precision Recall curve of ADABoost**

(Source: Code Output)

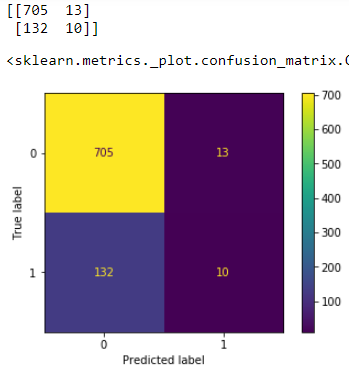
Figure below shows the ROC curve of ADABoost Model. The ROC curve shows that the model has perfect skill. The model has AUC of 0.68 which implies that the model is learning model.



**Figure: ROC curve of ADABoost**

(Source: Code Output)

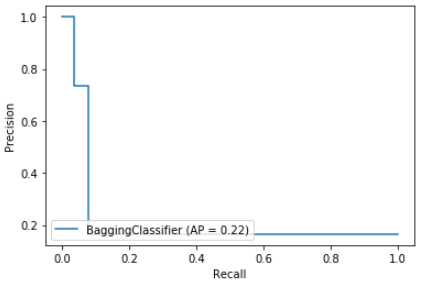
The figure below shows the confusion matrix of the model. The values in confusion matrix are used to generate all the above metrics. The True Positive of the model is 705, False Negative values is 132, False Positive values is 13 and True Negative of the model is 10. This shows that the model is making more negative predictions.



**Figure: Confusion matrix and plot of ADABoost**

(Source: Code Output)

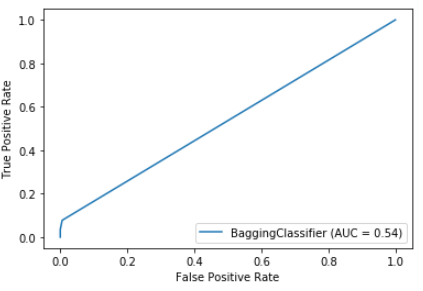
Next implemented the Bagging Classifier, Bagging model on the data. Bagging Classifier algorithm is a meta-learning based ensemble model which works by taking subsets of data, the data points in the random subset chosen randomly. The subsets are then fitted into the base estimator model for the for predictions and then the aggregate of those predictions are taken to make the final prediction. The model is generated with support vector machine as a base classifier and taking 10 estimators and the random state is et to 35. The learning rate is the ensemble weight on the weak learner features. The model generated the accuracy score of 83.38, F1-Score of 0, Precision score of 0 and recall rate of 0. The figure below show the precision recall curve of Bagging Classifier. There is no curve the line is horizontal and moving towards (0,0) which implies that the Bagging Classifier model is no skill model.



**Figure: Precision Recall curve of Bagging Classifier**

(Source: Code Output)

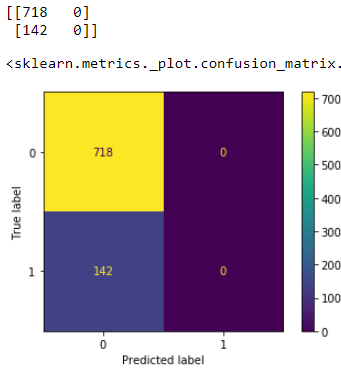
Figure below shows the ROC curve of Bagging Classifier Model. The ROC curve has no discriminative power as a diagonal line between a False Positive Rate of 0 and a True Positive Rate of 0 (coordinate (0,0) is formed. The model has AUC of 0.54 which implies that the model is perfect.



**Figure: ROC curve of** Bagging Classifier

(Source: Code Output)

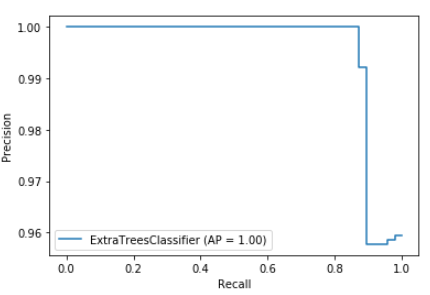
The figure below shows the confusion matrix of the model. The values in confusion matrix are used to generate all the above metrics. The True Positive of the model is 718, False Negative values is 142, False Positive values is 0 and True Negative of the model is 0. This shows that the model is only more negative predictions, and is not predicting a single positive predictions.



**Figure: Confusion matrix and plot of Bagging Classifier**

(Source: Code Output)

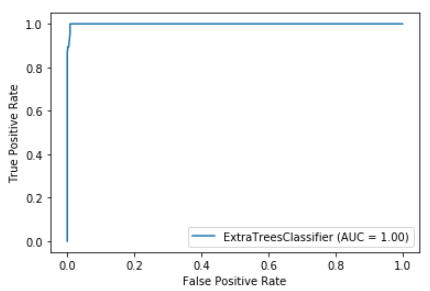
Next implemented the Extra Tree Classifier, a Bagging model on the data. Extra Tree Classifier algorithm is an ensemble model which aggregates the results of multiple de-correlated decision trees collected in a “forest” to output it's classification result. The model is generated for 50 estimators, criterion ‘entropy’ and max features is the square root. The model generated the accuracy score of 97.6, F1-Score of 93.28, Precision score of 99.20 and recall rate of 88.02. Figure below shows the precision recall curve of Extra Tree Classifier. As there is a curve that bows towards a coordinate of (1,1) this imply that the Extra Tree Classifier model is skillfull model.



**Figure: Precision Recall curve of Extra Tree Classifier**

(Source: Code Output)

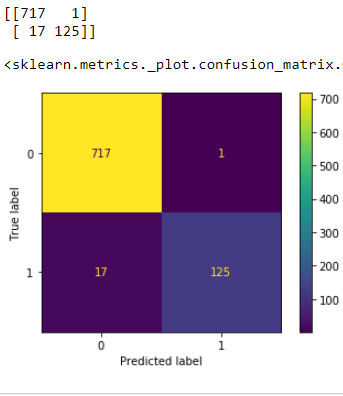
Figure below shows the ROC curve of Extra Tree Classifier Model. A model with perfect skill is represented at a point (0,1). A model with perfect skill is represented by a line that travels from the bottom left of the plot to the top left and then across the top to the top right. This shows that Extra tree classifier is a skilled model. The score is a value between 0.0 and 1.0 for a perfect classifier. The model has AUC of 1 which implies that the model is perfect.



**Figure: ROC curve of Extra Tree Classifier**

(Source: Code Output)

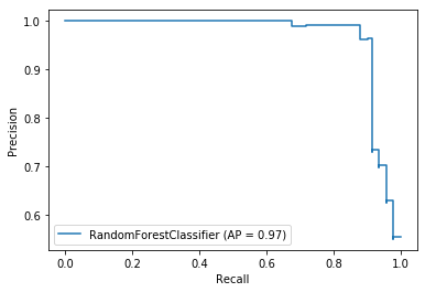
The figure below shows the confusion matrix of the model. The values in confusion matrix are used to generate all the above metrics. The True Positive of the model is 717, False Negative values is 17, False Positive values is 1 and True Negative of the model is 125. This shows that the model is making more true predictions.



**Figure: Confusion matrix and plot of Extra Tree Classifier**

(Source: Code Output)

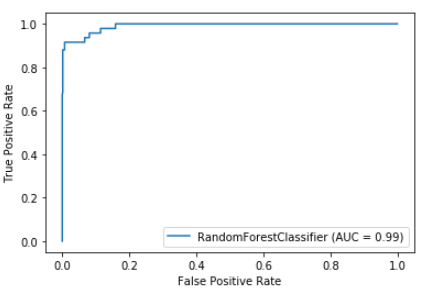
Next implemented the Random Forest Trees, Bagging model on the data. Random Forest Trees algorithm is a bagging algorithm for the categorical data. Random Forest Trees is based on decision tree algorithm. The difference is that it runs the data through various decision trees and then take the majority vote for the classification problems or average for the regression problems. The model is generated for a maximum depth of 15 and the random state of the model is 40. The model generated the accuracy score of 97.21, F1-Score of 90.84, Precision score of 99.17 and recall rate of 83.80. Figure below show the precision recall curve of Random Forest Trees. A model with perfect skill is depicted as a point at a coordinate of (1,1). A skillful model is represented by a curve that bows towards a coordinate of (1,1). A no-skill classifier will be a horizontal line on the plot with a precision that is proportional to the number of positive examples in the dataset. Random Forest Trees model is skillfull model.



**Figure: Precision Recall curve of Random Forest Trees**

(Source: Code Output)

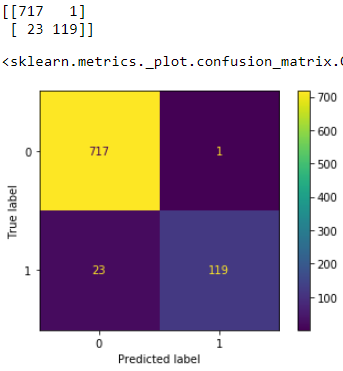
Figure below shows the ROC curve of Random Forest Trees Model. The ROC curve shows that the model has perfect skill. The model has AUC of 0.99 which implies that the model is perfect.



**Figure: ROC curve of Random Forest Trees**

(Source: Code Output)

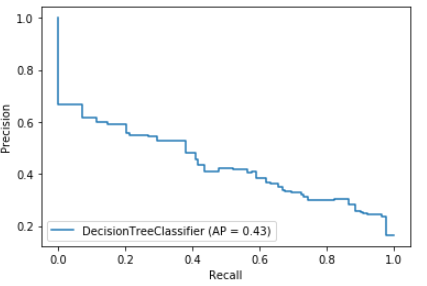
The figure below shows the confusion matrix of the model. The values in confusion matrix are used to generate all the above metrics. The True Positive of the model is 717, False Negative values is 23, False Positive values is 1 and True Negative of the model is 119. This shows that the model is making more true predictions.



**Figure: Confusion matrix and plot of Random Forest Trees**

(Source: Code Output)

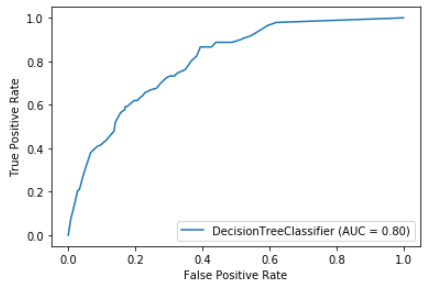
Last implemented the Decision Tree, a Bagging model on the data. Decision Tree is flow/chart or tree like algorithm which is best suited for categorical data as the flow can easily be put to conditions based on the values in each category. The model is generated with the default initializations for random state of 42 and weight fraction on leaf as 0.01. The model generated the accuracy score of 84.3, F1-Score of 36.02, Precision score of 55.07 and recall rate of 26.76. Figure below show the precision recall curve of Decision Tree implying that the model is a no skill model.



**Figure: Precision Recall curve of Decision Tree**

(Source: Code Output)

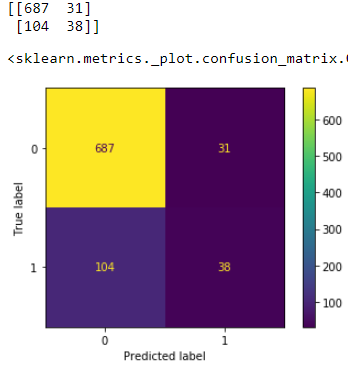
Figure below shows the ROC curve of Decision Tree Model. The ROC curve shows that the model has perfect skill. The model has AUC of 0.8 which implies that the model is perfect.



**Figure: ROC curve of Decision Tree**

(Source: Code Output)

The figure below shows the confusion matrix of the model. The values in confusion matrix are used to generate all the above metrics. The True Positive of the model is 687, False Negative values is 104, False Positive values is 31 and True Negative of the model is 38. This shows that the model is making more negatuve predictions.

****

**Figure: Confusion matrix and plot of Decision tree**

(Source: Code Output)

## 4.5 Evaluation of Metrics

To evaluate the model performance, used accuracy score, F1-Score, precision score and recall score as the performance metrics.  Accuracy is the fraction of predictions that the model predicted correct. F1- Score is the weighted average of precision and recall. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

For binary classification, all these metrics can also be calculated in terms of positives and negatives as follows:

Accuracy=(TP + TN) / TP + TN + FP + FN

F1 Score = 2\*(Recall \* Precision) / (Recall + Precision)

Precision = TP/TP+FP

Recall = TP/TP+FN

Where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

**True Positives (TP)** - These are the correctly predicted positive value which means that the value of the actual class is yes and the value of the predicted class is also yes. E.g. if actual class value indicates that this passenger survived and predicted class tells you the same thing.

**True Negatives (TN)** - These are the correctly predicted negative values which mean that the value of the actual class is no and value of the predicted class is also no. E.g. if the actual class says this passenger did not survive and predicted class tells you the same thing.

False positives and false negatives, these values occur when your actual class contradicts with the predicted class.

**False Positives (FP)** – When actual class is no and the predicted class is yes. E.g. if the actual class says this passenger did not survive but predicted class tells you that this passenger will survive.

**False Negatives (FN)** – When actual class is yes but predicted class in no. E.g. if actual class value indicates that this passenger survived and predicted class tells you that passenger will die.

As shown in Table 7 below shows the performance metrics of all the models. As per the accuracy score ranking for all the models, Extra tree classifer, a bagging algorithm has the highest accuracy score of 97.90%. This is followed by Random Forest Trees 97.2. CatBoost comes third with 94.65% accuracy score. At fourth place we have Light GBM with score of 94.06%. At fifth place we have Light GBM model with score of 88.95% accuracy score. Gradient Boosting model had an accuracy score of 86.04%. At seventh place we have Decision Tree with score of 84.3% accuracy Score. The model has the lowest score is bagging classifier and ADABoost having a score of 83%.

As per the F1-Score ranking for all the models, Extra tree classifer, a bagging algorithm has the highest F1-Score of 93.28%. This is followed by Random Forest Trees having F1-Score of 90.83%. At third place, comes. CatBoost comes third with 80.83% F1-Score. At fourth place we have Light GBM with score of 79.01% F1-Score. At fifth place we have XGBoost model with score of 54.01% F1-Score. At sixth place we have Decision Tree with score of 36.01% F1-Score. At seventh place we have Gradient Boosting with score of 34.78% F1-Score. At eighth place we have ADABoosts with score of 12.12% F1-Score. The model has the lowest F1-Score is bagging classifier having a score of 0%.

As per the Precision Score ranking for all the models, Extra tree classifer, a bagging algorithm has the highest Precision Score of 99.2%. This is followed by Random Forest Trees having Precision Score of 99.16%. At third place, comes. CatBoost comes third with 98.97% Precision Score. At fourth place we have Light GBM with score of 95.04% Precision Score. At fifth place we have XGBoost model with score of 86.15% Precision Score. At sixth place we have Gradient Boosting Tree with score of 76.19% Precision Score. At seventh place we have Decision Trees with score of 55.07% Precision Score. At eighth place we have ADABoost with score of 83.47% Precision Score. The model has the lowest Precision Score is bagging classifier having a score of 0%.

As per the Recall Score ranking for all the models, Extra tree classifer, a bagging algorithm has the highest Recall Score of 88.02%. This is followed by Random Forest Trees having Recall Score of 83.8%. At third place, comes. CatBoost comes third with 68.3% Recall Score. At fourth place we have Light GBM with score of 67.6% Recall Score. At fifth place we have XGBoost model with score of 39.43% Recall Score. At sixth place we have Decision Tree with score of 26.76% Recall Score. At seventh place we have Gradient Boosting with score of 22.53% Recall Score. At eighth place we have ADABoosts with score of 7.04% Recall Score. The model has the lowest F1-Score is bagging classifier having a score of 0%.

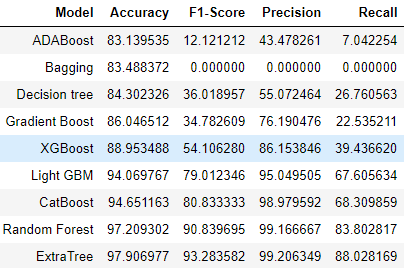


Table 8: Model Evaluation scores sorted by accuracy score

Table 9 below shows the list of the actual values and the values observed by the models. The observed values mean the predictions done by the model for the test data. From the table below it is evident that maximum similarity in the actual and observed values is for extra tree classifier. This accounts for the 97.9% accuracy score.

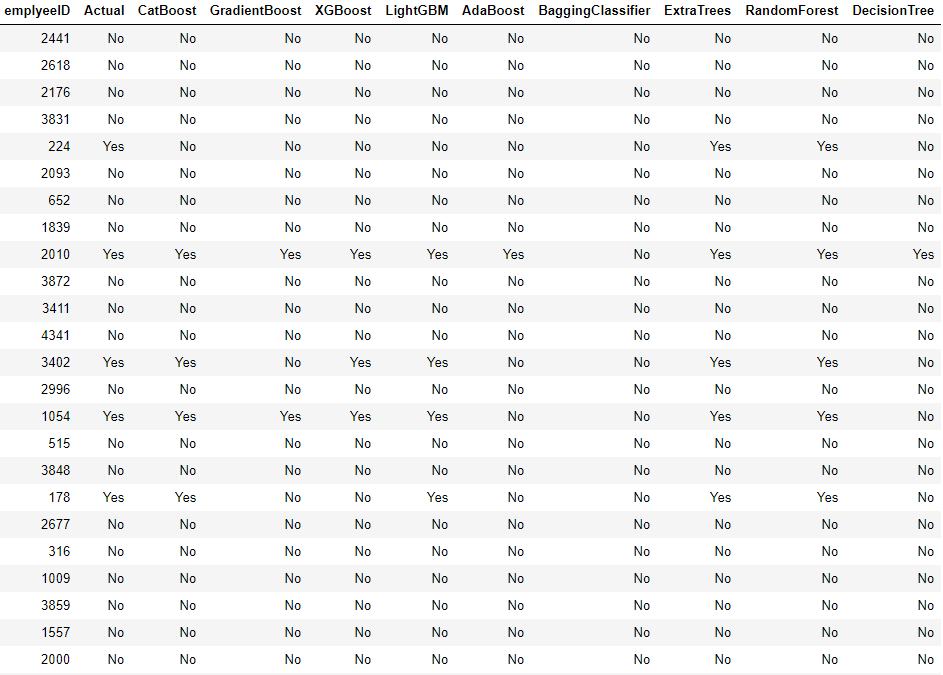
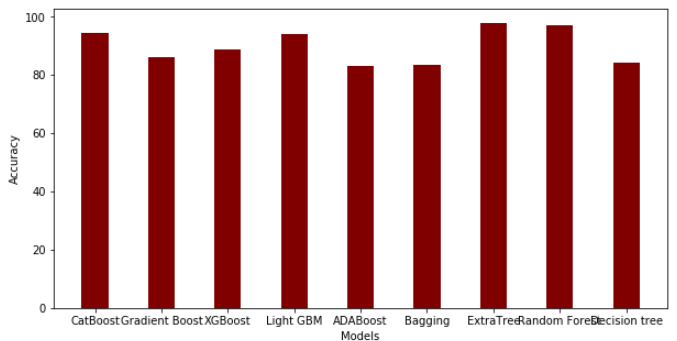


Table 8: Actual values and observed values of models

# Chapter 5: Conclusion and Future Work

* 1. **Conclusion**

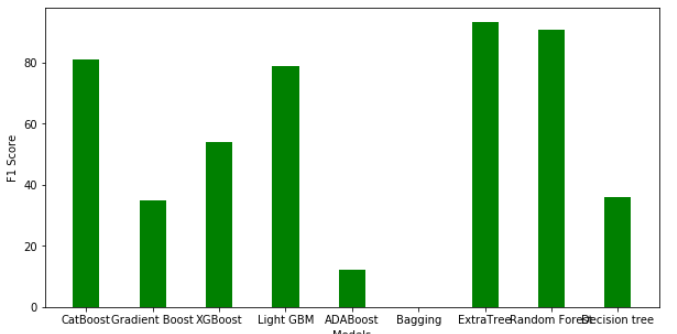
This research aims to find an efficient model to predict employee attrition. The model should predict the employee who is planning to leave the organisation before it happens. It is a challenging task as this involve study of the employee satisfaction rate with the managers as well as the colleagues, also it how his work and personal life is balanced. These are the feature which are captured in three different csv files, which are used for the model training and testing purpose. The research also aimed to evaluate which type of ensemble learning model bagging or boosting is best for the prediction of employee attrition. It is found that the Extra Tree classifier which is a bagging model is the best model to be implemented for the problem statement having the highest performance score in all the evaluation metrics. The figure below shows the plot of the accuracy score by the models. The biggest bar belongs to Extra Tree Classifier and shortest to Bagging Classifier.



**Figure: Accuracy plot**

(Source: Code Output)

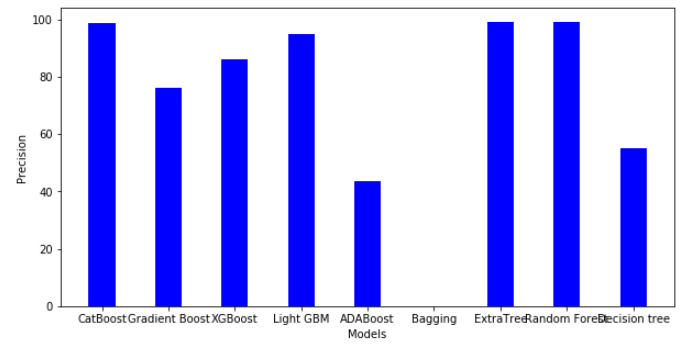
The figure below shows the plot of the F1-Score score by the models. The biggest bar belongs to Extra Tree Classifier and shortest or in this case no bar belongs to Bagging Classifier.



**Figure: F1 Score plot**

(Source: Code Output)

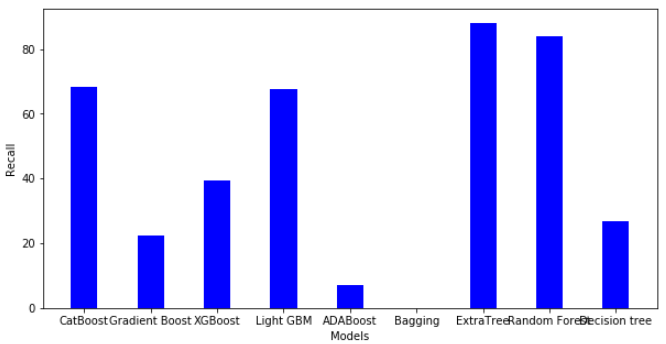
The figure below shows the plot of the precision score by the models. The biggest bar belongs to Extra Tree Classifier, XGBoost and Random Forest Trees and the shortest or in this case no bar belongs to Bagging Classifier.



**Figure: Precision score plot**

(Source: Code Output)

The figure below shows the plot of the recall score by the models. The biggest bar belongs to Extra Tree Classifier and shortest or in this case no bar belongs to Bagging Classifier.



**Figure: Recall rate plot**

(Source: Code Output)

The Extra Tree classifier has performed the best in all the evaluation metrics used.

* 1. **Limitations**

The limitations of the research are

* The data used in the research is considerably less in size. If the data would have been, the results may have been more appropriate.
* If there would have been a user interface, from which entering the details of the employee the result of his attrition probability can be determined.
* The data is from Kaggle.com, from which the data is downloaded by a lot of research for their research. It may be different from the actual data.
* There is a need of reliable and approved source of data can be downloaded.
  1. **Future Work**

For future work, it is recommended to train the model with more data from different sources so the performance of the models can be improved even more. It is also recommended to apply deep learning models like artificial neural networks and deep neural networks for attrition prediction.