LOAN DEFAULT CASE

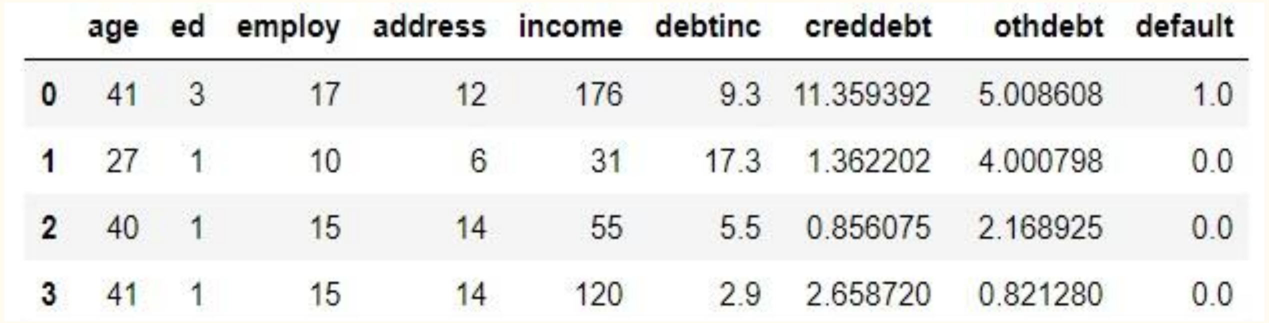
-SAKET KISHORE

PROBLEM STATEMENT:

The loan default dataset has 8 variables and 850 records, each record being loan default status for each customer. Each Applicant was rated as “Defaulted” or “Not-Defaulted”. New applicants for loan application can also be evaluated on these 8 predictor variables and classified as a default or non-default based on predictor variables.

CONTENTS

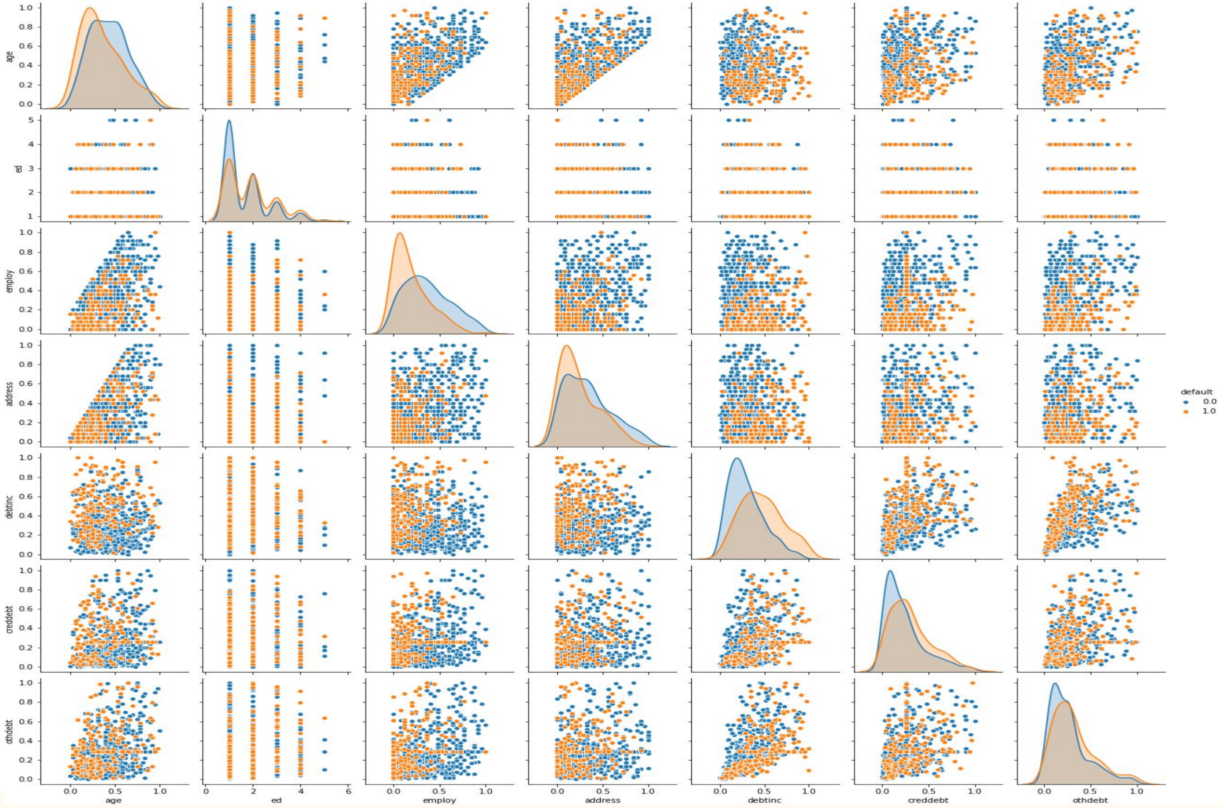
1. Exploratory Data Analysis



	age	ed	employ	address	income	debtinc	creddebt	othdebt	default
0	41	3	17	12	176	9.3	11.359392	5.008608	1.0
1	27	1	10	6	31	17.3	1.362202	4.000798	0.0
2	40	1	15	14	55	5.5	0.856075	2.168925	0.0
3	41	1	15	14	120	2.9	2.658720	0.821280	0.0

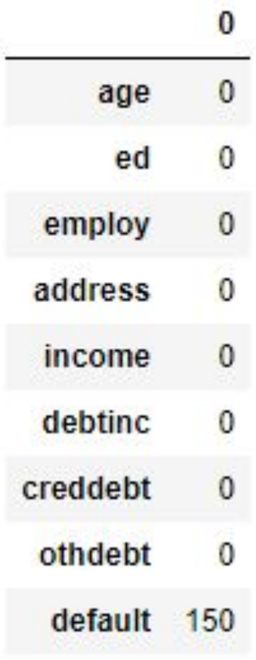
The dataset has 7 continuous dependent variable and only 1 categorical variable.
No extra work seems to be required here as dataset is clean. At a glance, the goal of this project is to create a model which predicts whether the loan will default or not based on 8 given attributes.
The key factor here to notice is that the goal of project is to create **a model for loss prevention**.
Predicting whether the loan will default or not is more important than predicting whether the loan will not default or not.
This means that if the model predicts the loan will not default(0) and it defaults(1) that will cause a lot of harm. Thus, other than increasing the accuracy, **decreasing the false negative rate(FNR) is important**.

This picture shows a pairplot made using the pairplot function in seaborn library.
From the start it is pretty clear that a **logistic regression would not be a very suitable method** for this problem because there is a a **lot of overlap** in the data, a best fit line that could separate the zeros and ones in the target variable will only be suitable for a small number of instances. **KNN method should be very effective here as there is a lot of overlap**. Other non linear classifiers such as **Random Forest and Decision Trees should work well too**, however for a **small dataset** like this one, **Decision Tree will probably lead to overfitting**. A small dataset however is an **advantage for Random Forest as it will take less time to train** the model which otherwise would have been a very time intensive process.



2. Missing Value Analysis

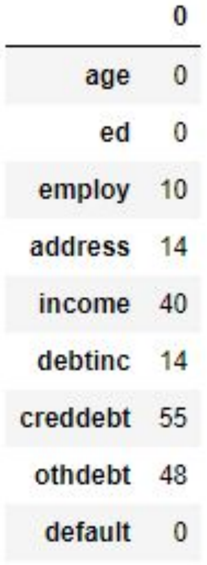
There were **150 missing values in the 'default'** variable.
Since default is the **target variable, these values cannot be imputed** because the model should be trained and tested on data with actual values.
Thus, **all 150 of those columns were dropped**.
The figure shows the number of missing values in each column.



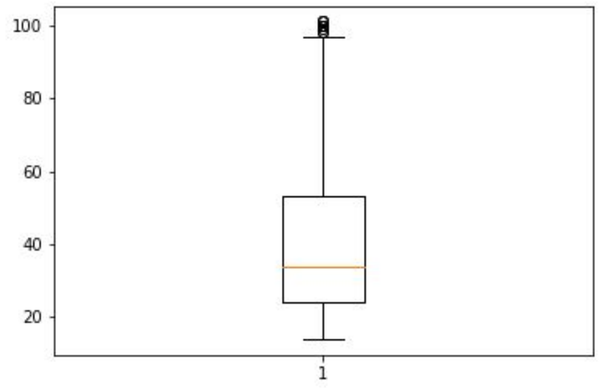
	0
age	0
ed	0
employ	0
address	0
income	0
debtinc	0
creddebt	0
othdebt	0
default	150

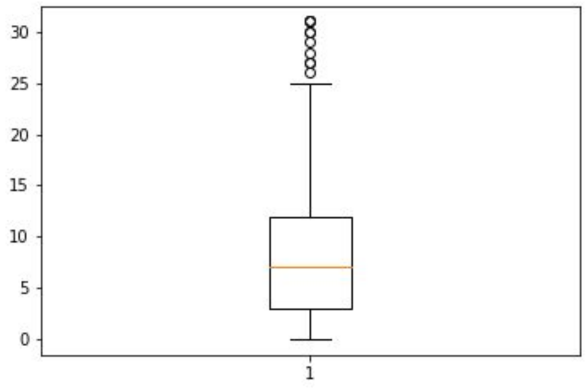
3. Outlier Analysis

Boxplots were plotted to check the outliers in the dataset and a lot of outliers were detected in multiple columns. The columns - age, employment status and address did not have too many outliers so they were not examined separately however, income, individual debt payment, debit to credit ratio and other debts were **examined separately and suitable methods of imputation were selected individually.**
The outliers from all the continuous variables were removed using a for loop and the **required imputation methods were implemented.** The figures show number of out
in each variable and some example boxplots.



	0
age	0
ed	0
employ	10
address	14
income	40
debtinc	14
creddebt	55
othdebt	48
default	0



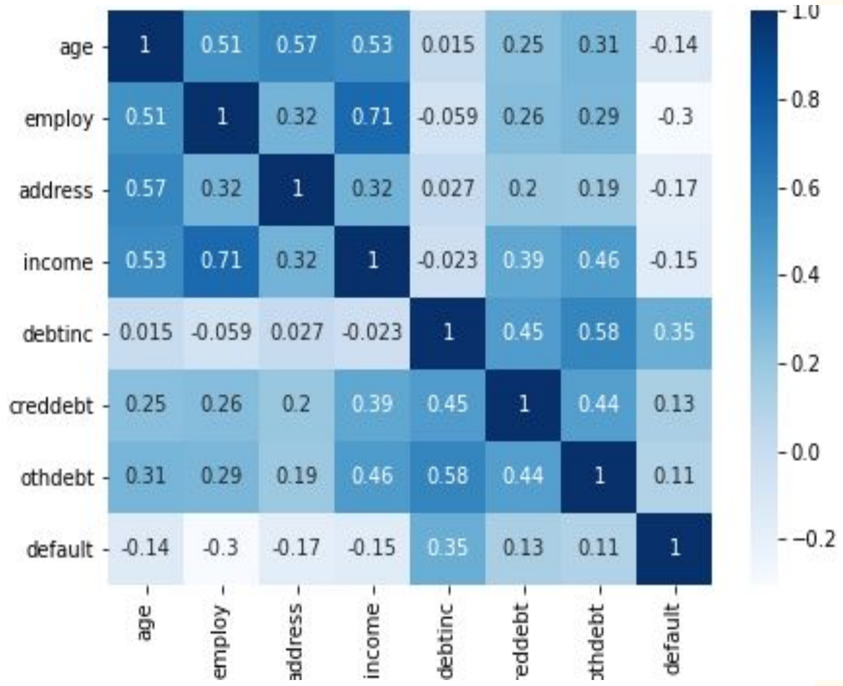


4. Feature Selection

To **avoid multicollinearity**, independent variables which are highly correlated to each other should be dropped. Since there is only one categorical independent variable, the chi-square test need not be performed. A **heatmap was plotted**.

Income and Employment Status had a correlation coefficient which crossed the allowable threshold(0.7)

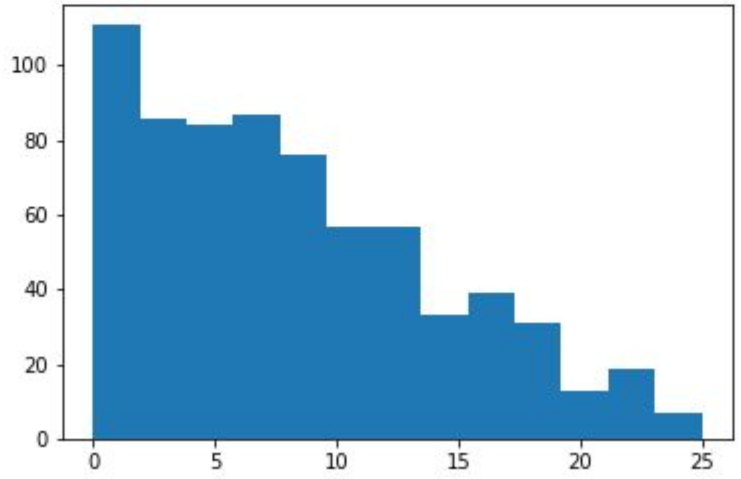
The variable **‘income’** was dropped because it had higher correlations with other variables as compared to employment status.



	age	employ	address	income	debtinc	creddebt	othdebt	default
age	1	0.51	0.57	0.53	0.015	0.25	0.31	-0.14
employ	0.51	1	0.32	0.71	-0.059	0.26	0.29	-0.3
address	0.57	0.32	1	0.32	0.027	0.2	0.19	-0.17
income	0.53	0.71	0.32	1	-0.023	0.39	0.46	-0.15
debtinc	0.015	-0.059	0.027	-0.023	1	0.45	0.58	0.35
creddebt	0.25	0.26	0.2	0.39	0.45	1	0.44	0.13
othdebt	0.31	0.29	0.19	0.46	0.58	0.44	1	0.11
default	-0.14	-0.3	-0.17	-0.15	0.35	0.13	0.11	1

5. Feature Scaling

Histograms were plotted for each continuous variable. Since **no variable had a normal distribution**, **normalization method was adapted** instead of standardization for scaling. The figure shows a sample histogram which was plotted for employment status variable.



6. Model Development

Five models were developed namely:

1. Logistic Regression
2. Decision Tree
3. Random Forest
4. KNN Method
5. Naive's Baye

A histogram was plotted and the number 0s and 1s were checked in the target variable. **Although the dataset was imbalanced, it was not an extreme imbalance, so oversampling was initially avoided.**

The test and train data were split in a standard way, that is **80% of the data was used to train the model and 20% of data was used to test the models**, for all the models.

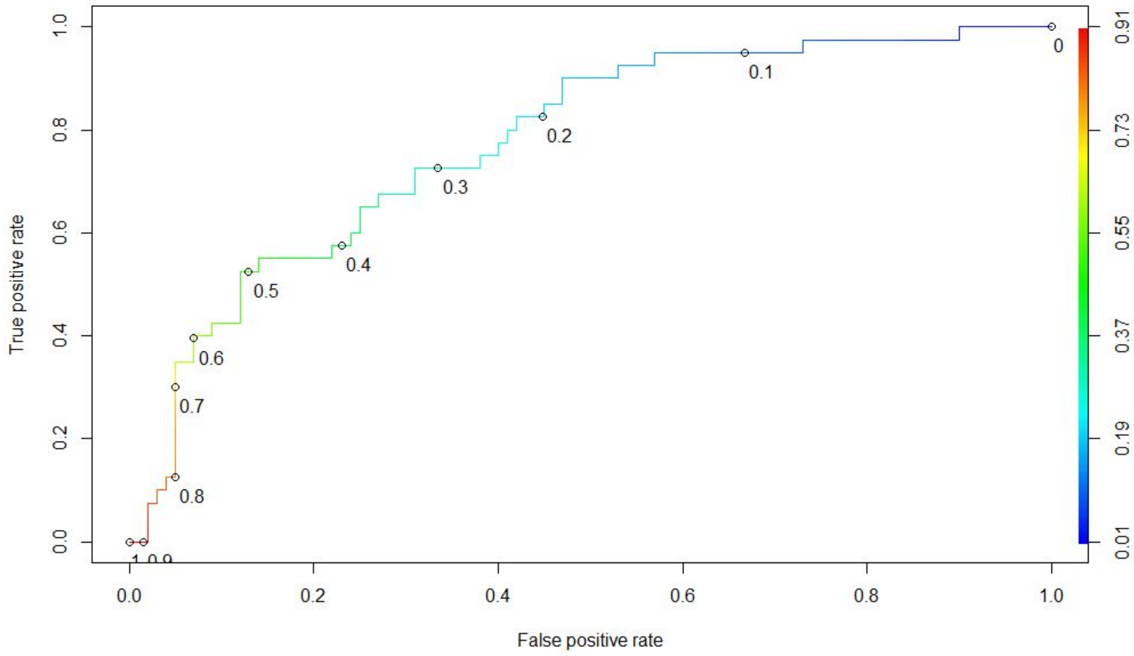
Logistic Regression

In this method logit scores are calculated using a logit link function which are then used to calculate the probabilities using a logistic function. These calculations are done for each continuous variable and each category of a categorical variable separately. Once the probabilities are obtained, a threshold value is used for classification.

The categorical variables were separated, data was split into test and train.
The **model was built on the train data, tested on test data and the probabilities were calculated**.
Once the probabilities were calculated, **an ROC curve was plotted to find the threshold value for classification**. Since for this specific problem, reducing the false negative rate is important, a high true positive rate was. Thus a trade off was made, False Positive Rate was allowed to go a little higher to produced a **low False Negative Rate.**

The following graph shows an **ROC curve plot** which was plotted using the ROCR library present in R.
The graph was used to find a suitable threshold value such that a higher true positive rate could be obtained while trading in for a slightly higher false positive rate.

A **threshold value of 0.4** was finalized.



Decision-Tree

In this method, the algorithm creates multiple trees with different root nodes and impurity is measured for each tree. The impurity is measured in terms of either entropy or gini index which are the two criterias of which one needs to be provide while writing the code.
The tree with the least amount of impurities is selected for the building the model.

Although both the methods, entropy and gini index obtain almost similar values, **in an imbalanced dataset, entropy can give a slight advantage** because in the formula to calculate gini index, probability is used directly however **in the case of entropy, logarithm of probability is used.**

In this case the dataset was imbalanced. Although it was not an extreme imbalance, the **criterion was chosen as entropy to give the model a slight extra edge**.

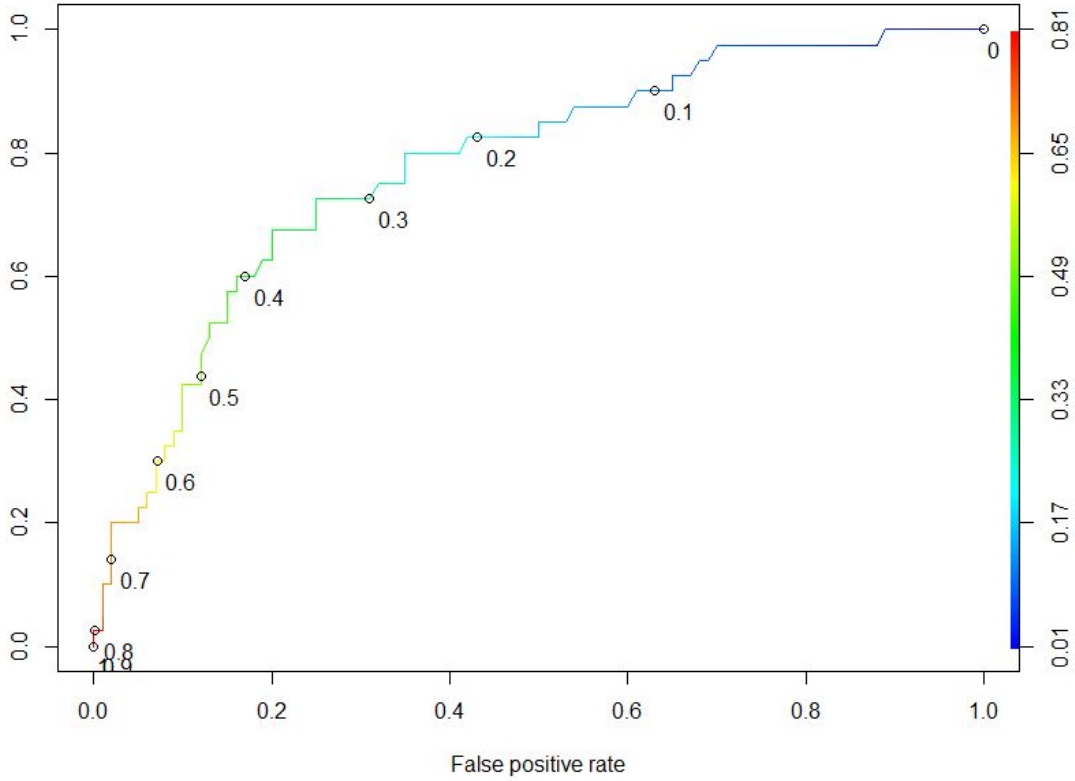
Random Forest

In a Random Forest classifier, **multiple decision trees are created and the predicted majority class from the result of those decision trees is selected**.
A **Decision Tree by itself is prone to overfitting** because a decision tree grown to its full depth learns the training dataset very well however when a new dataset is provided it might give high error rates. This **problem is fixed by using multiple decision trees as done in Random Forest**.

Random forest classifier was used to train a model on the training data and the probabilities were calculated so that the threshold could be adjusted manually. An ROC curve was plotted using ROCR library in R.

Looking at the graph we can see that the **cut-off value can be reduced upto 0.4** approximately without increasing false positive rate too much.

Thus, the **threshold value** for conversio of probabilities was **set to 0.4** for Random Forest.



KNN Method

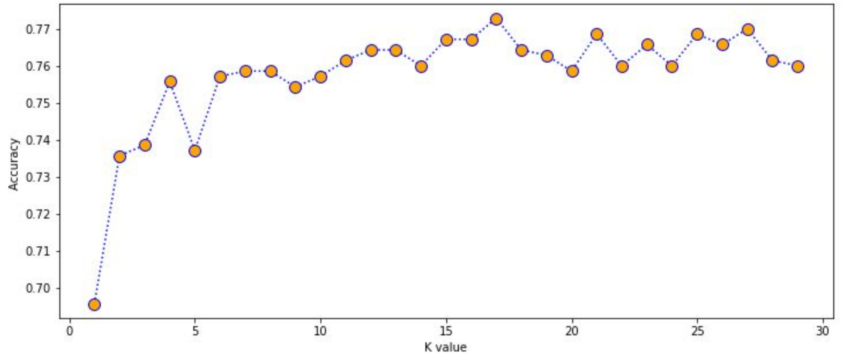
KNN or key nearest neighbour finds the N nearest neighbors and assigns the majority value from the neighbours to the target class.

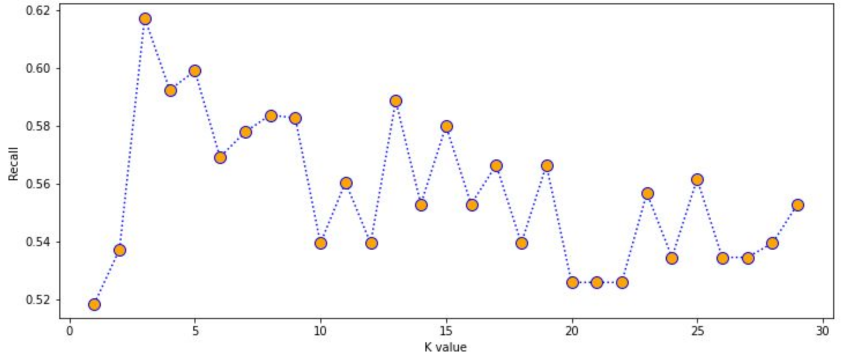
The **value of N provided should be odd** because there should not be a tie in between the two classes.

To **find a good value** of **K** for this project **2 graph was plotted**.
The graph had **K values plotted against** the two most important model characteristics for this specific project which are - **Accuracy and Recall**.

From the graphs as **suitable value for K was extracted**.

From the following graphs a K value of 17 was finalized as both accuracy and recall seem to somewhat stabilize after hitting the K=17 mark approximately .





Naive’s Baye

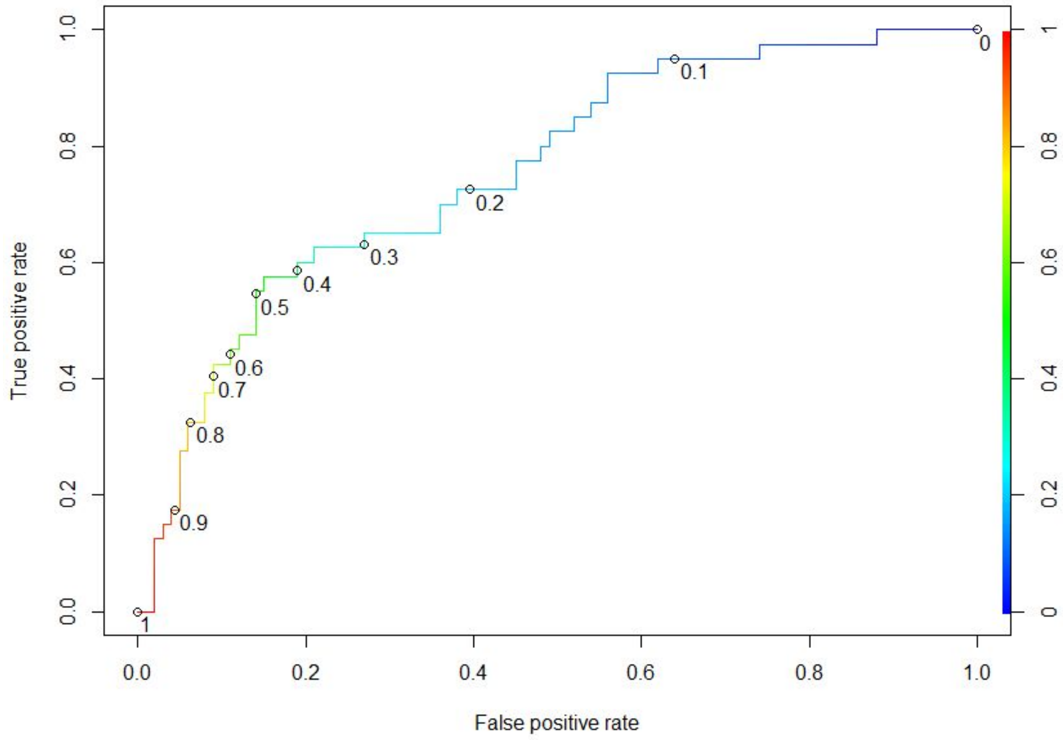
This algorithm is used only for classification problems. It is an **iterative process** which keeps learning based on the new information gain. For eg. - detecting spams in emails.

The algorithm **works on the assumption that each variable is contributing separately**. Thus, multicollinearity is not a problem for Naive’s Baye and correlation analysis can be skipped if this method is decided.

Naive’s baye **calculates the probability of positive and negative class** from the input of dependent variable data provided and assigns the class **with the higher probability to the data**, unless a threshold is set manually.

The figure shows an **ROC curve plotted for Naive's Baye** model which was produced using the ROCR library in R. Just like other models, initially the **probabilities were extracted using the parameter type=raw** while making the predictions using the model.

Later a **threshold value of 0.4** was finalized for classification.



7. Model Analysis

A confusion matrix and the two most important features for this project, accuracy and false negative rate were calculated for each model.

CONFUSION MATRIX	Predicted 0	Predicted 1
Actual 0	True Negative	False Positive
Actual 1	False Negative	True Positive

True Positive - The loan cases which the model predicted will default and they did.
False Positive - The loan cases which the model predicted will default but they didn’t.
True Negative - The loan cases which the model predicted will not default and they didn’t
False Negative - The loan cases which the model predicted will not default and they did.

ERROR METRICS

The accuracy and false negative rate were as follows :

Accuracy=76	FNR=53	for Logistic Regression
Accuracy=72	FNR=62	for Decision Tree
Accuracy=80	FNR=40	for Random Forest
Accuracy=79	FNR=46	for KNN Implementation
Accuracy=83	FNR=46	for Naives Baye

ROC CURVES

ROC curves were plotted for all the models. These were also plotted separately for each model in R to find a suitable threshold value for our specific problem which was to reduce the false negative rate

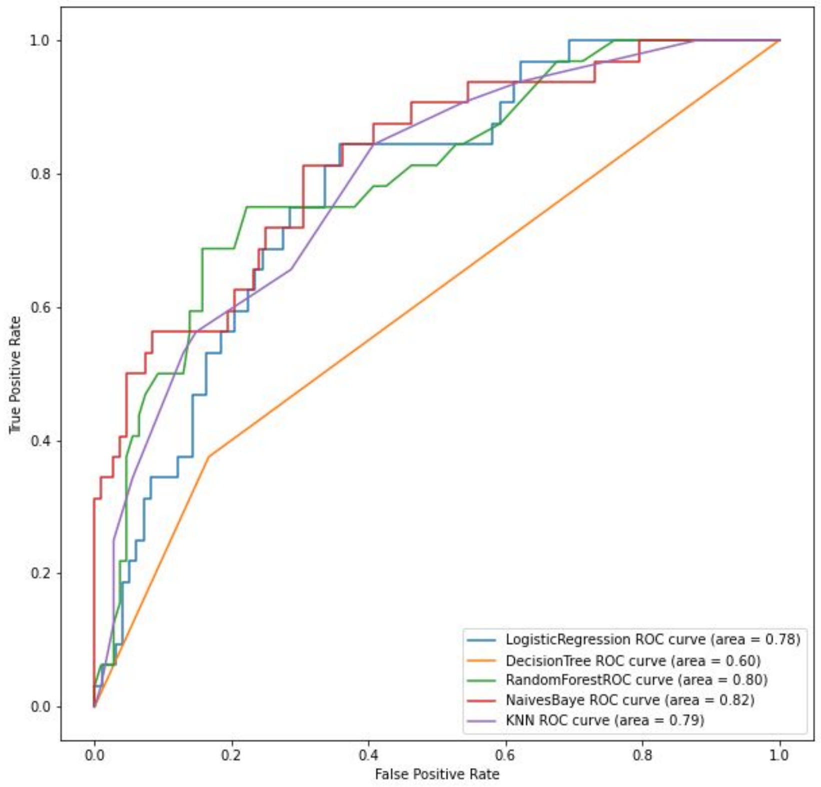
Y-axis : True Positive Rate

X-axis : False Positive Rate

Area under the curve :

0.78	for Logistic Regression
0.60	for Decision Tree
0.80	for Random Forest
0.79	for KNN Implementation
0.82	for Naives Baye

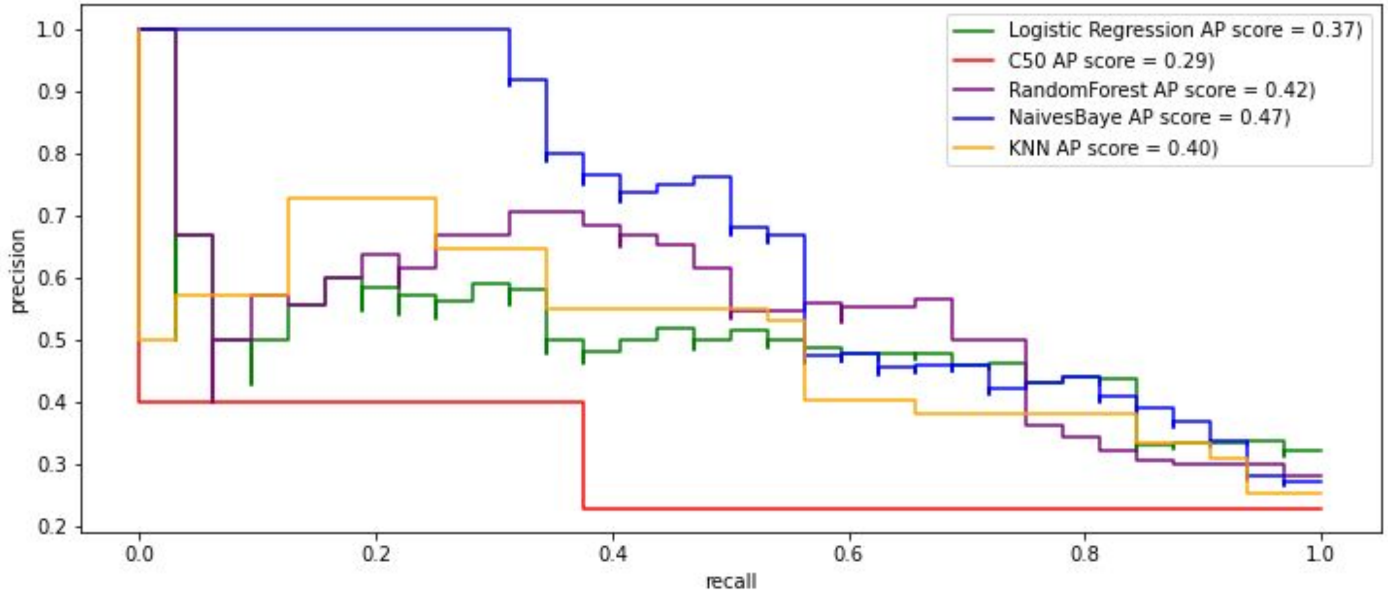
Higher the area under the curve, more accurate the model is considered.



PRECISION RECALL CURVES

A precision recall curve shows the trade off between precision and recall characteristics of the model.
A high precision is desired when the goal of the project is to reduce false positives.
However in our case, the focus is on recall because increasing the recall will result in reduction of false negative rate.

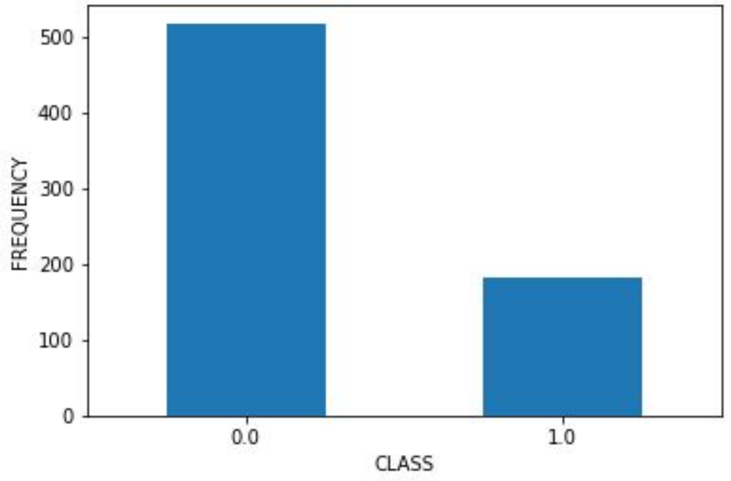
The legend shows averageprecision score score for every model.



OVERSAMPLING

Since using **methods such as increasing the number of trees in random forest, finding the best value of K for KNN method, decreasing the threshold value for all the models did not reduce the False Negative Rate as much as desired**, a library called SMOTETomek was used for oversampling. Thus, the number of 0s and 1s in the target variable was now balanced.
All the models were run on the new balanced set of data

The image shows the imbalance in the data.
There were **517 cases of no default and only 183 cases of loan default** in the dataset which **after over sampling** both the values **increased to 998 each**.



ERROR METRICS AFTER OVERSAMPLING

The accuracy and false negative rate were as follows :

Accuracy=77.5	FNR=16.8	for Logistic Regression
Accuracy=81.5	FNR=18.5	for Decision Tree
Accuracy=85.5	FNR=8.84	for Random Forest if threshold=0.4
Accuracy=77.0	FNR=9.73	for KNN Implementation
Accuracy=73.5	FNR=18.5	for Naives Baye
Accuracy=84.0	FNR=6.19	for Random Forest if threshold=0.3

8. MODEL SELECTION

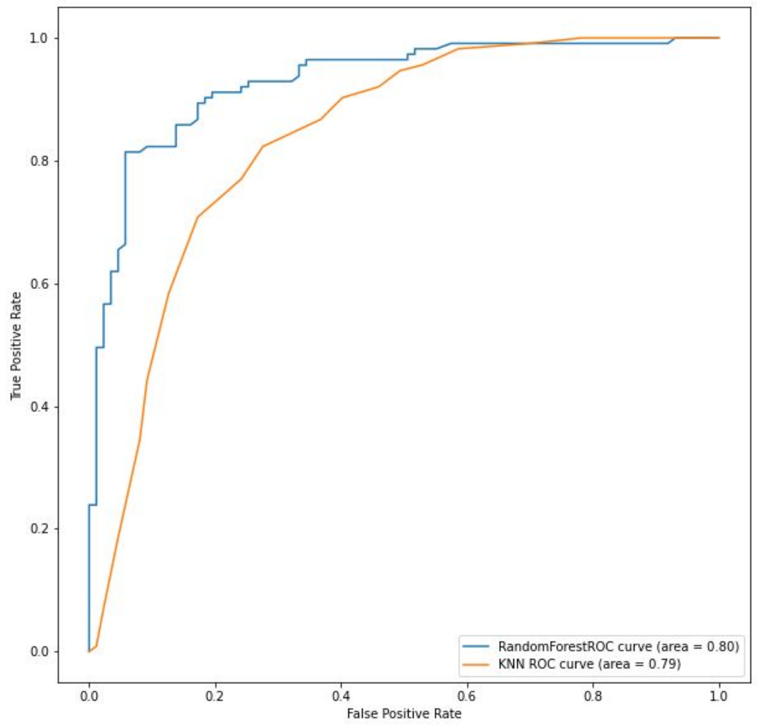
Two models, Random Forest and KNN were both giving **really good and desirable results**.
Both the models were then **tested for stability** using **cross validation**.
Both the models were giving **consistent output**.
The **mean cross validation score for accuracy** was calculated for both the models which turned out as follows :

Accuracy=78%	for Random Forest
Accuracy=77%	for KNN Implementation

ROC COMPARISON

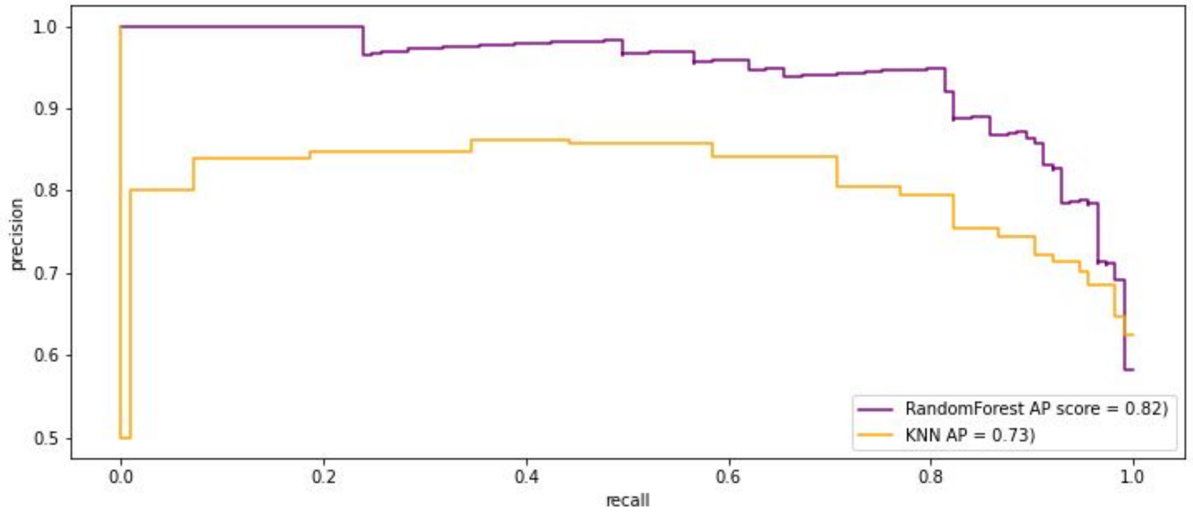
The graph shows an ROC curve plotted for Random forest and KNN only.

Area under the curve for
RandomForest : 0.80
KNN : 0.79



P-R CURVE COMPARISON

The graph shows a precision recall plot for Random Forest and KNN.
The average precision score for
Random Forest : 0.82
KNN : 0.73



FINALIZATION

Random forest with 500 trees and a threshold value of 0.4 was finalized as it proved better than all the other models on 3 performance metrics, namely :

1. Contingency table (Accuracy, FNR)
2. Area under ROC curve
3. Average precision score

The model also proved be stable and produced consistent results when cross validation score for accuracy and recall were calculated.

9. CONCLUSION

The loan default dataset had **850 observations with 8 dependent variables**. Various **methods were used to analyse and clean the data** after which **models were trained and tested**.

The **output was not satisfactory**, so a **new dataset was generated** by using **oversampling** method which had equal number of both positive and negative target classes. After oversampling, the models were again trained and tested. **Two models, KNN and Random Forest showed really good results** so they were **tested for stability** and were **further compared** using techniques such as area under the ROC curve, average precision score etc.

Random Forest with 500 trees and a threshold value of 0.4 proved to be the best for this specific problem. **To further decrease the False Negative Rate** if wanted (as low FNR is very important in this case) **the threshold can be set to 0.3**, this **won't cause a drastic loss in accuracy** while maintaining a very low **False Negative Rate.**

THANK YOU