# Project Report

**By Keyush Shah & Urvi Vaidya**

Finance Elective- Assessment Case Study

**INDEX**

## PROJECT DESCRIPTION

In the modern world determining someone's creditworthiness, or in simple terms, someone's ability to pay back a loan is a very important task. Thanks to rapid increases in data availability and computing power, machine learning now plays a vital role in both technology and business. Machine learning contributes significantly to credit risk modelling applications. Machine learning methods provide a better fit for the nonlinear relationships between the explanatory variables and default risk. We also find that using a broader set of variables to predict defaults greatly improves the accuracy ratio, regardless of the models used.

Machine learning is a method of teaching computers to parse data, learn from it, and then make a determination or prediction regarding new data. Rather than hand-coding a specific set of instructions to accomplish a particular task, the machine is "trained" using large amounts of data and algorithms to learn how to perform the task. Machine learning overlaps with its lower-profile sister field, statistical learning. Both attempt to find and learn from patterns and trends within large datasets to make predictions. The machine learning field has a long tradition of development, but recent improvements in data storage and computing power have made them ubiquitous across many different fields and applications, many of which are very commonplace. Apple's Siri, Facebook feeds, and Netflix movie recommendations all rely upon some form of machine learning. One of the earliest uses of machine learning was within credit risk modelling, whose goal is to use financial data to predict default risk.

When a person/business applies for a loan, the lender must evaluate whether the person/business can reliably repay the loan principal and interest. Lenders commonly use measures of profitability and leverage to assess credit risk. Given two loan applicants – one with high profitability and high leverage, and the other with low profitability and low leverage – which firm has lower credit risk? The complexity of answering this question multiplies when banks incorporate the many other dimensions they examine during credit risk assessment. These additional dimensions typically include other financial information such as liquidity ratio, or behavioural information such as loan/trade credit payment behaviour. Summarizing all of these various dimensions into one score is challenging, but machine learning techniques help achieve this goal.

The common objective behind machine learning and traditional statistical learning tools is to learn from data. Both approaches aim to investigate the underlying relationships by using a training dataset. Typically, statistical learning methods assume formal relationships between variables in the form of mathematical equations, while machine learning methods can learn from data without requiring any rules-based programming. As a result of this flexibility, machine learning methods can better fit the patterns in data.

For this task, we will be using the Assessment dataset provided to us, which is composed of 6000 loan applicants with a range of 15 features. It is up to us to determine whether the applicant is likely to default on his/her loan, based on the attributes provided to us for each applicant. The main aim of this project is to successfully predict the defaulters, according to "default" column which has two values 'yes' and 'no' for default. Although WOE is the requirement, in this assignment we will be using both PCA and WOE as feature selection methods so that we can compare the outcomes using different machine learning models. We will run different ML models on this dataset, using both the feature extraction methods, in order to determine which is the most accurate model. We will do some exploratory data analysis and just see what the dataset looks like. This will give us an overview of the no of datapoints, attributes, datatypes, general description, etc. We will also perform data cleaning in this step, so that we have a workable dataset. This involves checking for null values, replacing them if necessary, etc. At this stage we will also use different plots to visualize the data. - Visuals help us perform the initial high-level analysis. Our dataset has a 30:70 split of defaulters. We will use a Confusion Matrix and ROC – AUC curve to determine the success of our model.

## THE DATASET

To analyse the different Machine Learning Models we will be using the assessment dataset provided to us. Our dataset contains 6000 datapoints and 15 attributes/features. We will ignore the cust_id and acc_no columns as they do not add any value to our determinations.

| | |
|---|---|
| cust_id | age |
| acc_no | other_credit |
| checking_balance | housing |
| months_loan_duration | existing_loans_count |
| credit_history | job |
| purpose | dependants |
| amount (USD) | phone |
| savings_balance | default |
| employment_duration | |
| percent_of_income | |
| years_at_residence | |

As we can see we have a mix of financial ratios and behavioural information. In order to use these attributes in our ML models we need to ensure that the dataset contains no null/infinity values. Our dataset is relatively clean and only the years_at_residence columns has null values. We do not know whether this is an important feature for our ML models, therefore, at present we will retain it and fill all the null values with the mean of the column itself.

We will also check for object types as we cannot perform machine learning with object type values in the dataset. We already know that out default column is object type which we need to change to numerical, 1 - 0 for 'yes' and 'no' respectively.  We will not change the rest of the object types just yet as we need the dataset intact to perform WOE and IV. These are the object type columns along with the no of unique values for each:

| | |
|---|---|
| checking_balance | 4 |
| credit_history | 5 |
| purpose | 6 |
| savings_balance | 5 |
| employment_duration | 5 |
| other_credit | 3 |
| housing | 3 |
| job | 4 |
| phone | 2 |
| default | 2 |

Once we have finished with WOE & IV we will convert the object types to numericals. Although there are various ways to deal with object datatypes we will be using one hot encoding because we cannot rank some types of categories as either better or worse than the other therefore assigning integers to them would not be correct. Although some machine learning algorithms can work directly with categorical data depending on implementation, such as a decision tree, but most require any inputs or outputs variables to be a number, or numeric in value. This means that any categorical data must be mapped to integers.

One hot encoding is one method of converting data to prepare it for an algorithm and get a better prediction. With one-hot, we convert each categorical value into a new categorical column and assign a binary value of 1 or 0 to those columns. Each integer value is represented as a binary vector. All the values are zero, and the index is marked with a 1. One hot encoding is useful for data that has no relationship to each other. Machine learning algorithms treat the order of numbers as an attribute of significance. In other words, they will read a higher number as better or more important than a lower number.

We will perform some high-level analysis using different visuals. This will help us determine the structure of the dataset and check for any outliers in our data. The following observations are made using visual analysis.

- We first see the correlation of all the numeric columns and observe that there is no significant correlation between any of the features in the data.
- Secondly, we encode the default column and check the correlation with the target variable; we observe that the correlation has been insignificant for all the features.
- we can observe that contrary to logical assumption it is in fact applicants with perfect credit history who are more prone to defaulting than vice versa. This can be better observed below when we look at the actual numbers below.
- The data behaves as per norm, where people with lower account balances appear to be more likely to default that those with higher balances.
- We can observe that although the lowest number of applicants have applied for a car0 loan they are the highest defaulters. One would assume that Car0 refers to the first car of the applicant. Additionally even though the number of loans for education and business are low they too have a high default rate. Lastly we can observe that although the above two points are interesting it does not give us much insight into the data patterns as the default rate within the category is similarly distributed.
- It would  appear that people who own their own home are less likely to default that those who do not, however that is not conclusive information at present.

- The majority of the loans were taken for the purpose of buying furniture/appliances followed by car purchase.
- The people who were the most skilled hold the highest proportion in the list of defaulters, though the skilled people held the highest number of loans as per the data.
- Younger people took more loans than the older ones.
- None of the numeric columns has outliers except amount, months_loan_duration and age. The columns age and amount are bound to have some outliers which should not be a matter of concern and the amount of outliers in the months_loan_duration is negligible.
- None of the numeric columns seem to be normally distributed. Mostly they are right skewed.

## ATTRIBUTE SELECTION USING WOE & IV

The weight of evidence tells the predictive power of an independent variable in relation to the dependent variable. Since it evolved from credit scoring world, it is generally described as a measure of the separation of good and bad customers. "Bad Customers" refers to the customers who defaulted on a loan. and "Good Customers" refers to the customers who paid back loan.

Positive WOE means Distribution of Goods > Distribution of Bads whereas Negative WOE means Distribution of Goods < Distribution of Bads. It is calculated by taking the natural logarithm (log to base e) of division of % of non-events and % of events.

Information value is one of the most useful technique to select important variables in a predictive model. It helps to rank variables on the basis of their importance. Information value is not an optimal feature (variable) selection method when you are building a classification model other than binary logistic regression (for e.g. random forest or SVM) as conditional log odds (which we predict in a logistic regression model) is highly related to the calculation of weight of evidence. In other words, it's designed mainly for binary logistic regression model. We will write our own function to calculate the WOE and IV values. For our machine learning models we will use only those attributes that are scored as medium and strong predictors ignoring the rest of the columns.

*Since, random forest can detect non-linear relationship very well so selecting variables via Information Value and using them in random forest model might not produce the most accurate and robust predictive model.*

We will use a loop to get the WOE and IV of each attribute and create a list of features that are determined to be as strong or medium according to their Information Values

| **Attribute** | **IV** | **WOE .** |
|---|---|---|
| checking_balance | 0.6660115033513336 | suspicious |
| months_loan_duration | 0.30953743099843883 | strong |
| credit_history | 0.2932335473908263 | medium |
| purpose | 0.03835478242959107 | weak |
| amount (USD) | 0.04210822801901128 | weak |
| savings_balance | 0.19600955690422672 | medium |
| employment_duration | 0.086433631026641 | weak |

| | | |
|---|---|---|
| percent_of_income | 0.02632209005433453 | weak |
| years_at_residence | 0.008819966530677974 | useless |
| age | 0.25736464632239925 | medium |
| other_credit | 0.057614541955647885 | weak |
| housing | 0.08329343361549926 | weak |
| existing_loans_count | 0.013266524242854377 | useless |
| job | 0.008762765707428294 | useless |
| dependants | 4.339222702973195e-05 | useless |
| phone | 0.0063776050286746735 | useless |

*Our loop gives us the following as the  selected_features based on our conditions:*

months_loan_duration

credit_history

savings_balance

age

## SMOTE

- Using imbalanced data, we may have a model that appears very accurate, while it actually is useless! This is where SMOTE (Synthetic Minority Oversampling Technique) comes in.
- SMOTE is an algorithm that performs data augmentation by creating synthetic data points based on the original data points. SMOTE can be seen as an advanced version of oversampling, or as a specific algorithm for data augmentation. The advantage of SMOTE is that you are not generating duplicates, but rather creating synthetic data points that are slightly different from the original data points.
- Although our dataset is pretty balanced as we can see from our above visuals, we will still use SMOTE and visualize our data to get a better feel.

*We will not be using the resampled data for our model predictions, we are simply viewing it for academic purposes.*

## ATTRIBUTE SELECTION & MODEL TESTING USING PCA

Principal Component Analysis (PCA) is used to explain the variance-covariance structure of a set of variables through linear combinations. It is often used as a dimensionality-reduction technique. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation.

We start with prepping the data. This involves splitting the data into two dataframes wherein the features are stored in one dataframe and the target value (which is the default in the present case) is stored in the second datafarame. At this stage we can choose the columns we wish to retain or not, also known as feature extraction. At present we will retain all the columns in our first set except the customer_id and account_no columns, since these add no value. We have performed this step above when we did SMOTE.

We will use PCA for feature extraction and run a few models so that we can compare it to the models we run with attributes chosen using WOE and IV. Since we have already converted the non-numerical datapoints, taken care of the null values and prepped the data we can straightaway move on to PCA and Model Prediction.

We can see from the output that the PCA is not working very well. Further, PCA does not allow us any control over the choice of features only the number of components. PCA seems is pretty inaccurate for most models, therefore we will not reproduce the results herein. We will next use different machine learning models using the WOE & IV selected_features.

## CONFUSION MATRIX & ROC- AUC CURVE

We create a function to fit our data into the required models, perform the predictions and give us a confusion matrix as output. We will call this function using different models. We will also have the random state set as 42 for our train test split so that we have consistent outputs and are better able to compare the different models. Our function will also give us the AUC score and print the ROC curve. We will first use it to run the different models using all the attributes and then just the selected_features.

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an **error matrix**. It evaluates the performance of the classification models, when they make predictions on test data, and tells how good our classification model is. It not only tells the error made by the classifiers but also the type of errors such as it is either type-I or type-II error. With the help of the confusion matrix, we can calculate the different parameters for the model, such as accuracy, precision, etc.

The AUC-ROC curve helps us visualize how well our machine learning classifier is performing. The **Receiver Operator Characteristic (ROC)** curve is an evaluation metric for binary classification problems. It is a probability curve that plots the **TPR** against **FPR** at various threshold values and essentially **separates the 'signal' from the 'noise'**. The **Area Under the Curve (AUC)** is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.

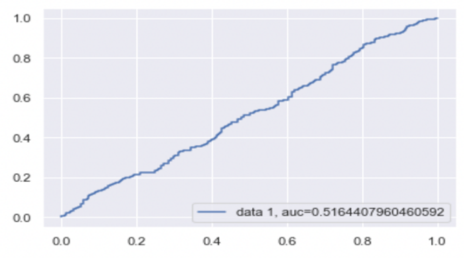
When AUC = 1, then the classifier is able to perfectly distinguish between all the Positive and the Negative class points correctly. If, however, the AUC had been 0, then the classifier would be predicting all Negatives as Positives, and all Positives as Negatives. When 0.5<AUC<1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative class values. This is so because the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives. When AUC=0.5, then the classifier is not able to distinguish between Positive and Negative class points. Meaning either the classifier is predicting random class or constant class for all the data points.

## MODEL TESTING USING ALL ATTRIBUTES

We have run the following Machine Learning Models using all the 15 attributes. We will also view the outputs for better understanding. As we can see our Decision Tree and random Forest Models are performing a bit too well and therefore we must be suspicious of them. We do not want models that overfit, therefore these models are not of much use to us when we use all the attributes.

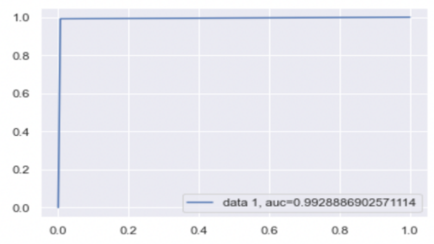
### LOGISTIC REGRESSION

```
Classification report:
              precision    recall  f1-score   support

           0       0.70      1.00      0.82       836
           1       0.00      0.00      0.00       364

    accuracy                           0.70      1200
   macro avg       0.35      0.50      0.41      1200
weighted avg       0.49      0.70      0.57      1200

Confusion matrix:
 [[836   0]
 [364   0]]
Accuracy Score:
0.697
AUC:
0.5164407960460592
ROC:
```



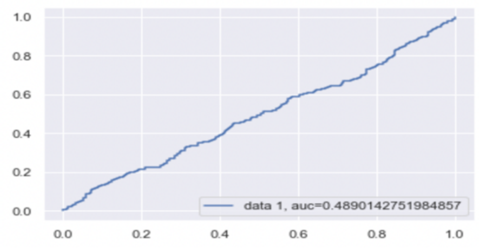
### DECISION TREE

```
Classification report:
              precision    recall  f1-score   support

           0       1.00      0.99      1.00       836
           1       0.99      0.99      0.99       364

    accuracy                           0.99      1200
   macro avg       0.99      0.99      0.99      1200
weighted avg       0.99      0.99      0.99      1200

Confusion matrix:
 [[831   5]
 [  3 361]]
Accuracy Score:
0.993
AUC:
0.9928886902571114
ROC:
```



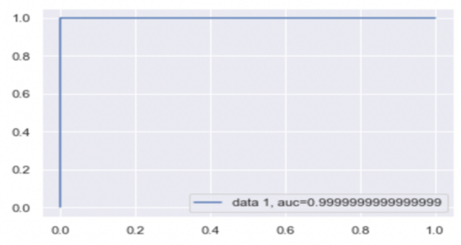
### NAIVE BAYES

```
Classification report:
              precision    recall  f1-score   support

           0       0.70      1.00      0.82       836
           1       0.00      0.00      0.00       364

    accuracy                           0.70      1200
   macro avg       0.35      0.50      0.41      1200
weighted avg       0.49      0.70      0.57      1200

Confusion matrix:
 [[836   0]
 [364   0]]
Accuracy Score:
0.697
AUC:
0.4890142751984857
ROC:
```



### RANDOM FORESTS

```
Classification report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       836
           1       1.00      0.99      1.00       364

    accuracy                           1.00      1200
   macro avg       1.00      1.00      1.00      1200
weighted avg       1.00      1.00      1.00      1200

Confusion matrix:
 [[836   0]
 [  2 362]]
Accuracy Score:
0.998
AUC:
0.9999999999999999
ROC:
```

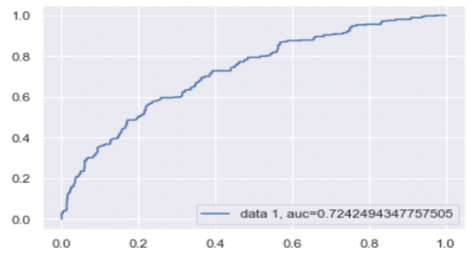


## MODEL TESTING USING SELECTED FEATURES

We will now use our selected_features which was the assignment requirement. Although we have selected only 4 features our model will take in more columns as we have created flag variables. We will now prep our data using just these 4 features. We selected the following features using WOE & IV: months_loan_duration (strong), credit_history (medium), savings_balance (medium) & age (medium).

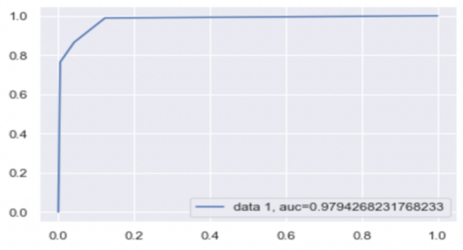
### LOGISTIC REGRESSION

```
Classification report:
              precision    recall  f1-score   support

           0       0.75      0.94      0.83       836
           1       0.66      0.27      0.39       364

    accuracy                           0.74      1200
   macro avg       0.71      0.61      0.61      1200
weighted avg       0.72      0.74      0.70      1200

Confusion matrix:
 [[786  50]
 [265  99]]
Accuracy Score:
0.738
AUC:
0.7242494347757505
ROC:
```



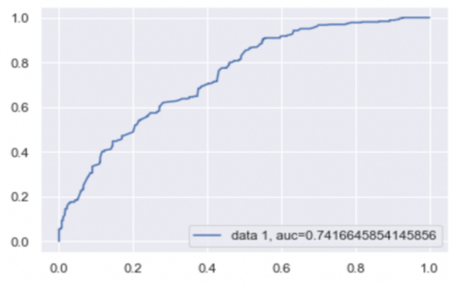
### K NEIGHBORS CLASSIFIERS

```
Classification report:
              precision    recall  f1-score   support

           0       0.94      0.96      0.95       836
           1       0.90      0.87      0.88       364

    accuracy                           0.93      1200
   macro avg       0.92      0.91      0.92      1200
weighted avg       0.93      0.93      0.93      1200

Confusion matrix:
 [[801  35]
 [ 49 315]]
Accuracy Score:
0.930
AUC:
0.9794268231768233
ROC:
```



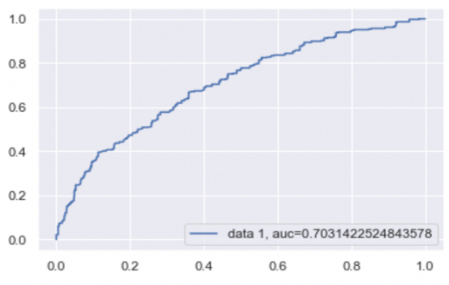
### ADA BOOST CLASSIFIER

```
Classification report:
              precision    recall  f1-score   support

           0       0.74      0.93      0.83       836
           1       0.62      0.25      0.36       364

    accuracy                           0.73      1200
   macro avg       0.68      0.59      0.59      1200
weighted avg       0.71      0.73      0.68      1200

Confusion matrix:
 [[781  55]
 [273  91]]
Accuracy Score:
0.727
AUC:
0.7416645854145856
ROC:
```



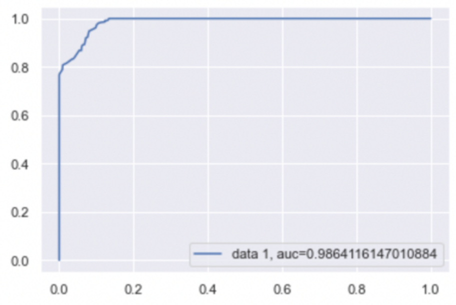
### NAÏVE BAYES

```
Classification report:
              precision    recall  f1-score   support

           0       0.76      0.89      0.82       836
           1       0.60      0.36      0.45       364

    accuracy                           0.73      1200
   macro avg       0.68      0.63      0.63      1200
weighted avg       0.71      0.73      0.71      1200

Confusion matrix:
 [[748  88]
 [234 130]]
Accuracy Score:
0.732
AUC:
0.7031422524843578
ROC:
```



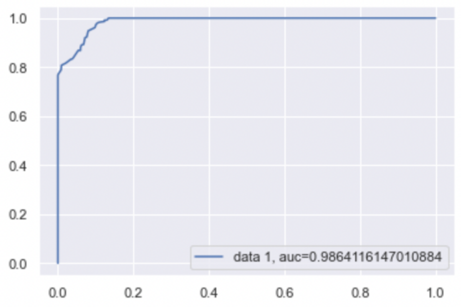
## DESCISION TREE

```
Classification report:
              precision    recall  f1-score   support

           0       0.93      0.96      0.95       836
           1       0.90      0.84      0.87       364

    accuracy                           0.92      1200
   macro avg       0.92      0.90      0.91      1200
weighted avg       0.92      0.92      0.92      1200

Confusion matrix:
 [[802  34]
 [ 59 305]]
Accuracy Score:
0.922
AUC:
0.9864116147010884
ROC:
```



## RANDOM FOREST

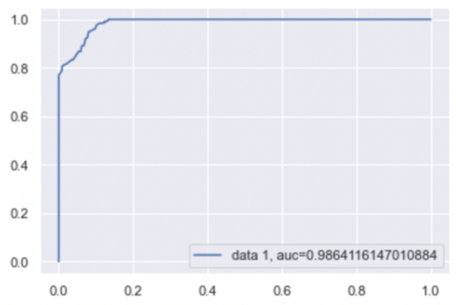
```
Classification report:
              precision    recall  f1-score   support

           0       0.93      0.96      0.95       836
           1       0.90      0.84      0.87       364

    accuracy                           0.92      1200
   macro avg       0.92      0.90      0.91      1200
weighted avg       0.92      0.92      0.92      1200

Confusion matrix:
 [[802  34]
 [ 59 305]]
Accuracy Score:
0.922
AUC:
0.9864116147010884
ROC:
```



## RANDOM FOREST

## WITH N_ESTIMATOR

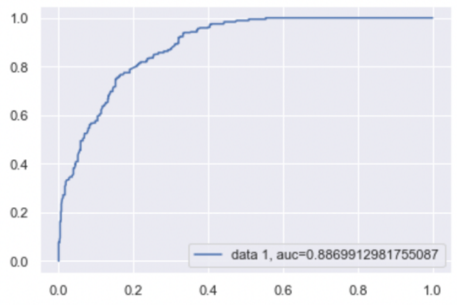
```
Classification report:
              precision    recall  f1-score   support

           0       0.93      0.96      0.95       836
           1       0.90      0.84      0.87       364

    accuracy                           0.92      1200
   macro avg       0.92      0.90      0.91      1200
weighted avg       0.92      0.92      0.92      1200

Confusion matrix:
 [[802  34]
 [ 59 305]]
Accuracy Score:
0.922
AUC:
0.9864116147010884
ROC:
```



## RANDOM FOREST

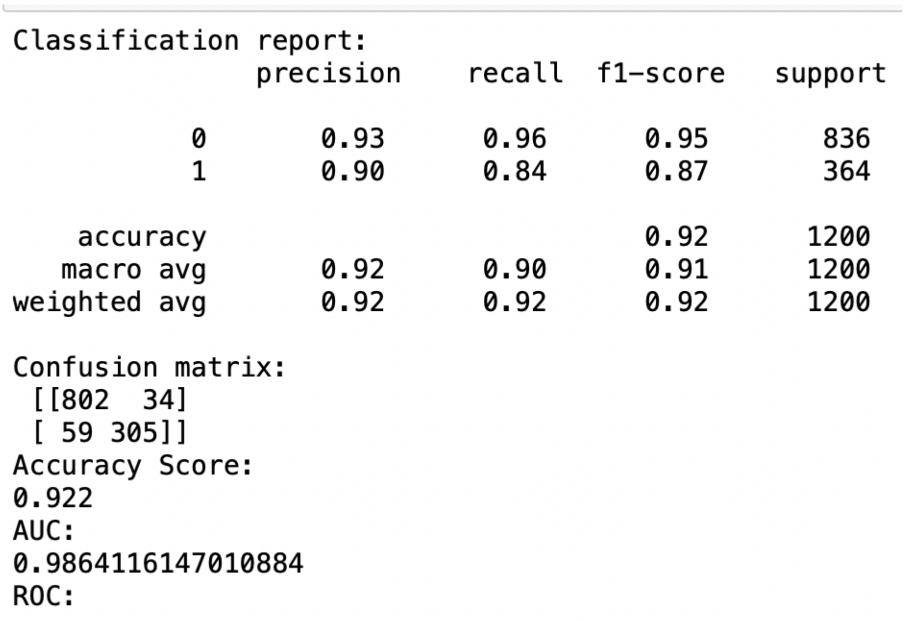
## WITH MODEL OPTIONS

```
Classification report:
              precision    recall  f1-score   support

           0       1.00      0.40      0.57       836
           1       0.42      1.00      0.59       364

    accuracy                           0.58      1200
   macro avg       0.71      0.70      0.58      1200
weighted avg       0.82      0.58      0.58      1200

Confusion matrix:
 [[332 504]
 [  0 364]]
Accuracy Score:
0.580
AUC:
0.8869912981755087
ROC:
```

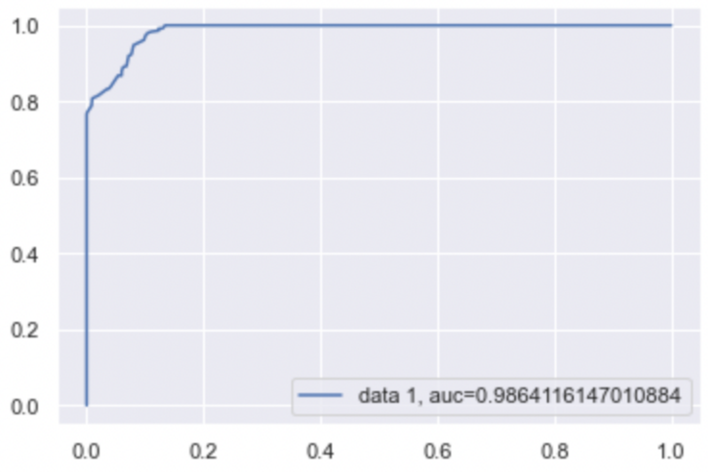


## MODEL FINALISATION & CONCLUSION

- Looking at the above outputs, we can safely choose Decision Tree Classifier using WOE & IV as our final model since it has the highest accuracy score.
- In addition to the overall accuracy the f1- scores of both the Negative and Positive results are very high too.
- These numbers are closely followed by Random Forest as our second best model, however as we have stated above Random forest can detect non-linear relationship very well so selecting variables via Information Value and using them in random forest model might not produce the most accurate and robust predictive model.
- Overall Decision Tree Classifier has given us the best results.
- The Decision Tree code is reproduced hereinbelow for easy reference.



```
Classification report:
              precision    recall  f1-score   support

           0       0.93      0.96      0.95       836
           1       0.90      0.84      0.87       364

    accuracy                           0.92      1200
   macro avg       0.92      0.90      0.91      1200
weighted avg       0.92      0.92      0.92      1200

Confusion matrix:
 [[802  34]
 [ 59 305]]
Accuracy Score:
0.922
AUC:
0.9864116147010884
ROC:
```


data 1, auc=0.9864116147010884