GERMAN CREDIT CARD DATA SET

Varun Jaglan

Contents

Summary

The bank receives requests for loans and must decide about the customer's eligibility for loan based on their applications. The risk to the bank is not approving the customers with good credit, who most likely will repay the loan, or approving the customers with bad credit which are not likely to repay the loan and that can result in financial loss for the Bank.

To help the bank to make a better decision that minimizes the risk that are based on their applications' demographic and economic profile, we applied different predictive models to analyze data (Naive Bayes, Decision Tree and Logistic regression We explored on ways to improve the accuracy of classification models – we introduced SMOTE data and checked if Naïve Bias algorithm's accuracy improved. Ensemble methods such as Bagging were implemented by choosing Decision Tree classifier. Cost Sensitive learning with Logistic Regression as tested. Finally, as part of predictive modelling) we compared all the algorithms to on accuracy and concluded that Naïve Bias yielded the best results.

For the post predictive analysis, we used Clustering and Association techniques. Recommendations to the bank was based on the output of both these algorithms.

We recommend to the bank that creditable borrowers would be categorized as those who do not need guarantors, have no other loans/credits (and no instalment plans), have fewer dependents, own their apartments, have low savings and are thus motivated to borrow, pay off loans on time and want smaller loans. We also recommended to disapprove any loan application for those applicants who pay their existing credit duly, but their account balances are less than 100 DM or even are over drafted. They are living paycheck to paycheck and have no financial means to repay any additional debt.

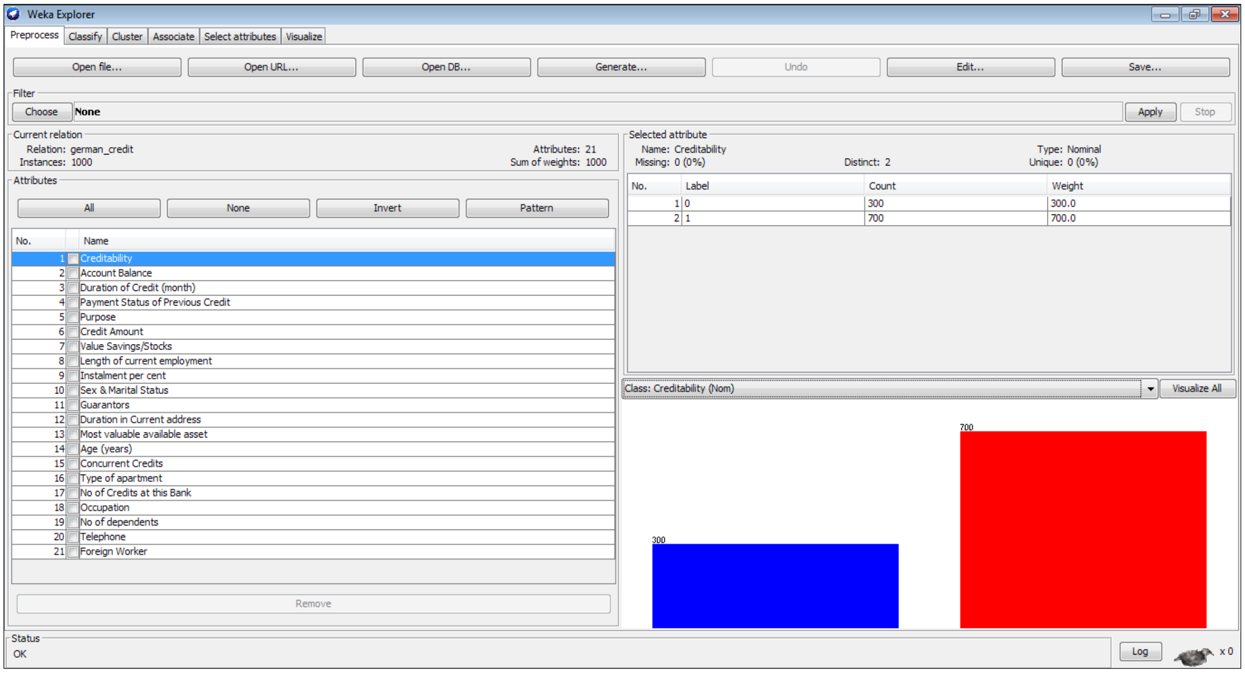
1. DATA PREPARATION

1.1. TYPES OF VARIABLES

The German Credit Card data set consisted of 1000 observations with 21 attributes. The attributes were classified into qualitative and quantitative data types. Qualitative data represents different types of data values such as gender, male and female whereas quantitative data represents numerical values such as test scores, weight etc.

For data preparation, the Credit Card dataset was loaded into Weka (Figure 1). The total number of high-risk individuals (300: shown in blue) is much lower than the amount of low risk individuals (700: shown in red). The data set contained 15 qualitative (10 Nominal and 5 Ordinal) and 6 quantitative (Numerical) data types. The attributes are described in Table 1. The new customers for credit are evaluated based on these 21 attributes.

Figure 1: WEKA explorer with German Credit Card data



Filter: None

Current relation
Relation: german_credit
Instances: 1000
Attributes: 21
Sum of weights: 1000

Attributes

No.	Name
1	Creditability
2	Account Balance
3	Duration of Credit (month)
4	Payment Status of Previous Credit
5	Purpose
6	Credit Amount
7	Value Savings/Stocks
8	Length of current employment
9	Instalment per cent
10	Sex & Marital Status
11	Guarantors
12	Duration in Current address
13	Most valuable available asset
14	Age (years)
15	Concurrent Credits
16	Type of apartment
17	No of Credits at this Bank
18	Occupation
19	No of dependents
20	Telephone
21	Foreign Worker

Selected attribute
Name: Creditability
Missing: 0 (0%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	0	300	300.0
2	1	700	700.0

Status
OK

1.2. ATTRIBUTE DISTRIBUTION

Table 1: Attribute Data Type

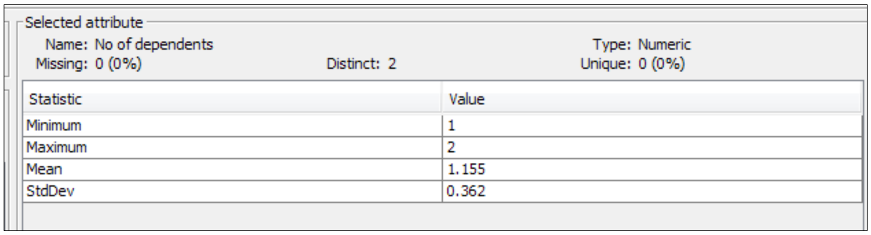
S. No	Attribute	Type	Sub Type
1	Creditability	Qualitative	Nominal
2	Purpose	Qualitative	Nominal
3	Sex & Marital Status	Qualitative	Nominal
4	Telephone	Qualitative	Nominal
5	Foreign Worker	Qualitative	Nominal
6	Occupation	Qualitative	Nominal
7	Type of apartment	Qualitative	Nominal
8	Guarantors	Qualitative	Nominal
9	Most valuable available asset	Qualitative	Nominal
10	Concurrent Credits	Qualitative	Nominal
11	Length of current employment	Qualitative	Ordinal
12	Payment Status of Previous Credit	Qualitative	Ordinal
13	Value Savings/Stocks	Qualitative	Ordinal
14	Duration in Current address	Qualitative	Ordinal
15	Account Balance	Qualitative	Ordinal
16	Duration of credit (month)	Quantitative	Numerical
17	Credit Amount	Quantitative	Numerical
18	Installment per cent	Quantitative	Numerical
19	Age (years)	Quantitative	Numerical
20	No. of Credits at this Bank	Quantitative	Numerical
21	No. of dependents	Quantitative	Numerical

1.3. NUMERICAL DATA SUMMARY

To obtain maximum, minimum, mean, median and standard deviation values, the data set was loaded into Weka and R. In Weka, the "selected attribute" panel lists the name of the selected attribute, data type, number of distinct values, missing values along with its statistics and their values (Figure 2).

In R, "summary ()" function (> summary(data119)) was used to review each attribute in the dataset and the results are shown in Table 2.

Figure 2: Statistics for Numerical attributes in WEKA



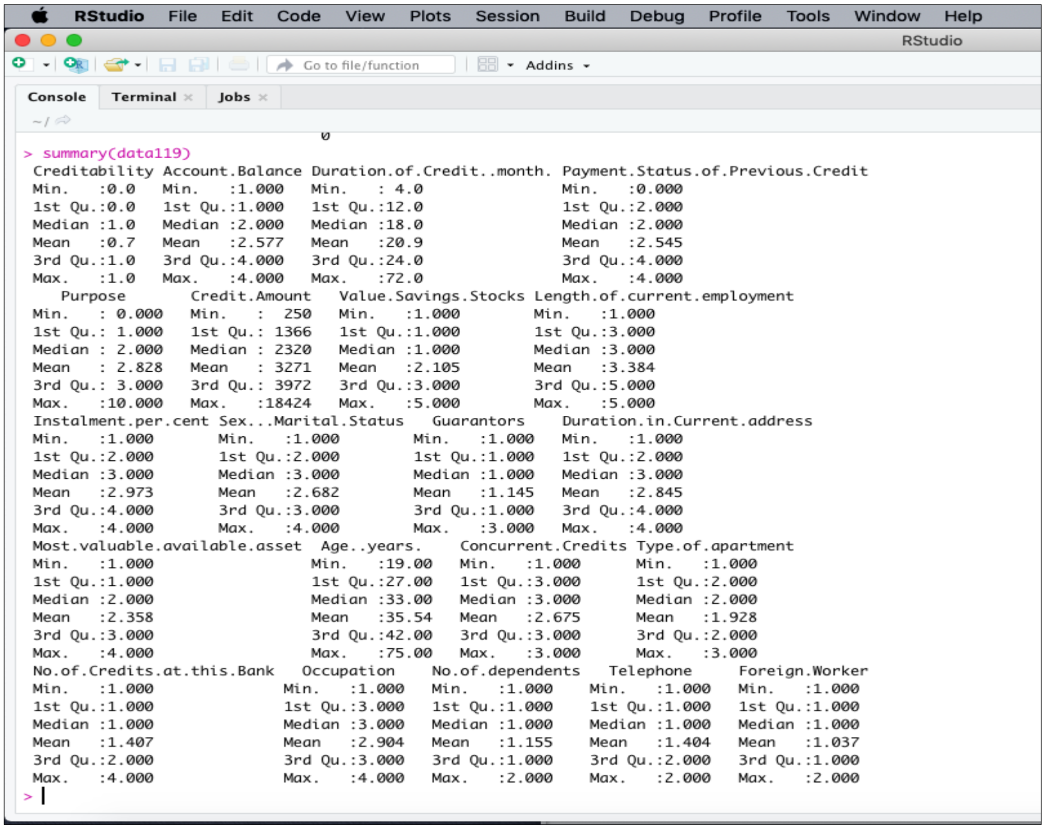
Selected attribute

Name: No of dependents   Type: Numeric

Missing: 0 (0%)   Distinct: 2   Unique: 0 (0%)

Statistic	Value
Minimum	1
Maximum	2
Mean	1.155
StdDev	0.362

Figure 3: Summary Function in R



```
                                0
> summary(data119)
 Creditability Account.Balance Duration.of.Credit..month. Payment.Status.of.Previous.Credit
 Min.   :0.0   Min.   :1.000   Min.   : 4.0               Min.   :0.000
 1st Qu.:0.0   1st Qu.:1.000   1st Qu.:12.0               1st Qu.:2.000
 Median :1.0   Median :2.000   Median :18.0               Median :2.000
 Mean   :0.7   Mean   :2.577   Mean   :20.9               Mean   :2.545
 3rd Qu.:1.0   3rd Qu.:4.000   3rd Qu.:24.0               3rd Qu.:4.000
 Max.   :1.0   Max.   :4.000   Max.   :72.0               Max.   :4.000
    Purpose       Credit.Amount   Value.Savings.Stocks Length.of.current.employment
 Min.   : 0.000   Min.   :  250   Min.   :1.000        Min.   :1.000
 1st Qu.: 1.000   1st Qu.: 1366   1st Qu.:1.000        1st Qu.:3.000
 Median : 2.000   Median : 2320   Median :1.000        Median :3.000
 Mean   : 2.828   Mean   : 3271   Mean   :2.105        Mean   :3.384
 3rd Qu.: 3.000   3rd Qu.: 3972   3rd Qu.:3.000        3rd Qu.:5.000
 Max.   :10.000   Max.   :18424   Max.   :5.000        Max.   :5.000
 Instalment.per.cent Sex...Marital.Status   Guarantors    Duration.in.Current.address
 Min.   :1.000       Min.   :1.000        Min.   :1.000   Min.   :1.000
 1st Qu.:2.000       1st Qu.:2.000        1st Qu.:1.000   1st Qu.:2.000
 Median :3.000       Median :3.000        Median :1.000   Median :3.000
 Mean   :2.973       Mean   :2.682        Mean   :1.145   Mean   :2.845
 3rd Qu.:4.000       3rd Qu.:3.000        3rd Qu.:1.000   3rd Qu.:4.000
 Max.   :4.000       Max.   :4.000        Max.   :3.000   Max.   :4.000
 Most.valuable.available.asset  Age..years.    Concurrent.Credits Type.of.apartment
 Min.   :1.000                 Min.   :19.00   Min.   :1.000      Min.   :1.000
 1st Qu.:1.000                 1st Qu.:27.00   1st Qu.:3.000      1st Qu.:2.000
 Median :2.000                 Median :33.00   Median :3.000      Median :2.000
 Mean   :2.358                 Mean   :35.54   Mean   :2.675      Mean   :1.928
 3rd Qu.:3.000                 3rd Qu.:42.00   3rd Qu.:3.000      3rd Qu.:2.000
 Max.   :4.000                 Max.   :75.00   Max.   :3.000      Max.   :3.000
 No.of.Credits.at.this.Bank   Occupation    No.of.dependents   Telephone     Foreign.Worker
 Min.   :1.000              Min.   :1.000   Min.   :1.000    Min.   :1.000   Min.   :1.000
 1st Qu.:1.000              1st Qu.:3.000   1st Qu.:1.000    1st Qu.:1.000   1st Qu.:1.000
 Median :1.000              Median :3.000   Median :1.000    Median :1.000   Median :1.000
 Mean   :1.407              Mean   :2.904   Mean   :1.155    Mean   :1.404   Mean   :1.037
 3rd Qu.:2.000              3rd Qu.:3.000   3rd Qu.:1.000    3rd Qu.:2.000   3rd Qu.:1.000
 Max.   :4.000              Max.   :4.000   Max.   :2.000    Max.   :2.000   Max.   :2.000
> |
```

Table 2: Numerical Data Summary

Attribute	Maximum	Minimum	Mean	Median	Std. Deviation
Duration of credit (month)	72	4	20.903	18	12.059
Credit Amount	18424	250	3271.25	2320	2822.752
Installment per cent	4	1	2.973	3	1.119
Age (years)	75	19	35.542	33	11.353
No. of Credits at this Bank	4	1	1.407	1	0.578
No. of dependents	2	1	1.155	1	0.362

From the statistics summary in Table 2 and Figure 3, it can be concluded that:

1. **Duration of Credit**: The average duration of credit is 20.9 months which is 21 months approximately.
2. **Credit Amount**: Mean>Median, the data is right skewed which means there are more people with smaller amounts of credit. Maximum amount for the loan is $18,424.
3. **Instalment per cent**: Mean<Median, the data is slightly left skewed which means there are more people who have 3-4% instalment rate.
4. **Age:** Mean >Median, the data is right skewed which means younger people are more likely to apply for the loan or credits (early 30s).

5. **No. of Credits at this Bank**: As Minimum and Median data points are same and data is right skewed (Mean>median), there is more than 50% of the data which has only 1 credit at this bank.
6. **No. of Dependents**: Minimum, 1st Quartile, Median and 3rdQuartile all of them are at 1, so 75% or more people have only 1 dependent in their credit application.

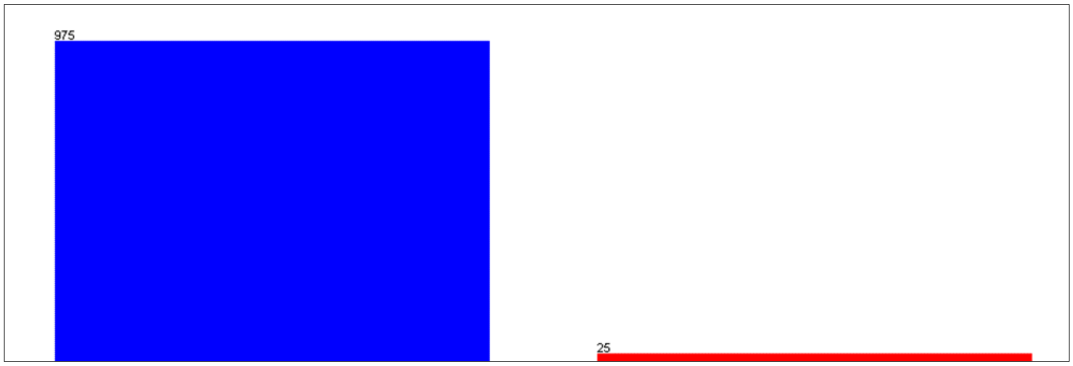
1.4. OUTLIER AND MISSING VALUE

To identify and analyze outliers and missing value, the InterQuartileRange (IQR) feature in Weka was applied on the dataset. This gave an additional attribute called Outlier (Nominal type), which had two distinct values (Yes or No). In the Credit Card dataset, there were 975 No's (if an instance was not an outlier then IQR assigned No as a label) and only 25 Yes's (if it was an outlier). This was good because there were not too many outliers in the data set (Figure 4). The attributes with outliers were:

1. Duration of Credit
2. Age
3. Credit Amount
4. No of Credits at this Bank
5. Instalment Per cent
6. No of dependents

For this project, the outliers were not removed as only 2.5% of the entire data set had outliers verus 97.5% the data which did not had outliers. Since, the data set was small (contains only 1,000 observations) and most of the data was of Nominal type, the small percentage of outliers was not going to affect the data. Also, there were no missing values for any of the 21 attributes (Figure 2).

Figure 4: Outlier analysis in WEKA



1.5. DISTRIBUTION OF ATTRIBUTES

Histogram is a bar chart based on the frequency distribution. The x-axis shows the categories and y-axis shows the frequency of those categories. Below is a summary of distribution of each attribute from Weka (Figure 5) and R (Figure 6,7,8 and 9). Based on these distributions it can be concluded that:

1. **Age:** People who are below the age of 40 tend to apply for loan. The histogram is skewed to the right which means the early part of the histogram has higher frequency.

2. **Credit Amount:** People tend to seek a credit amount less than 8,000. The graph appears to be skewed to the right.
3. **No of dependents:** People with one dependent tend to seek loan as compared to people with more than one dependent.
4. **No of Credits at this Bank:** Population with just one existing loan tend to apply for a new loan.
5. **Sex & Marital Status:** Males who are single, constituted most of the population who had applied for loan.
6. **Occupation:** Skilled employee constituted of most of the population who had applied for loan.

Figure 5: Histogram of all 21 attributes in WEKA

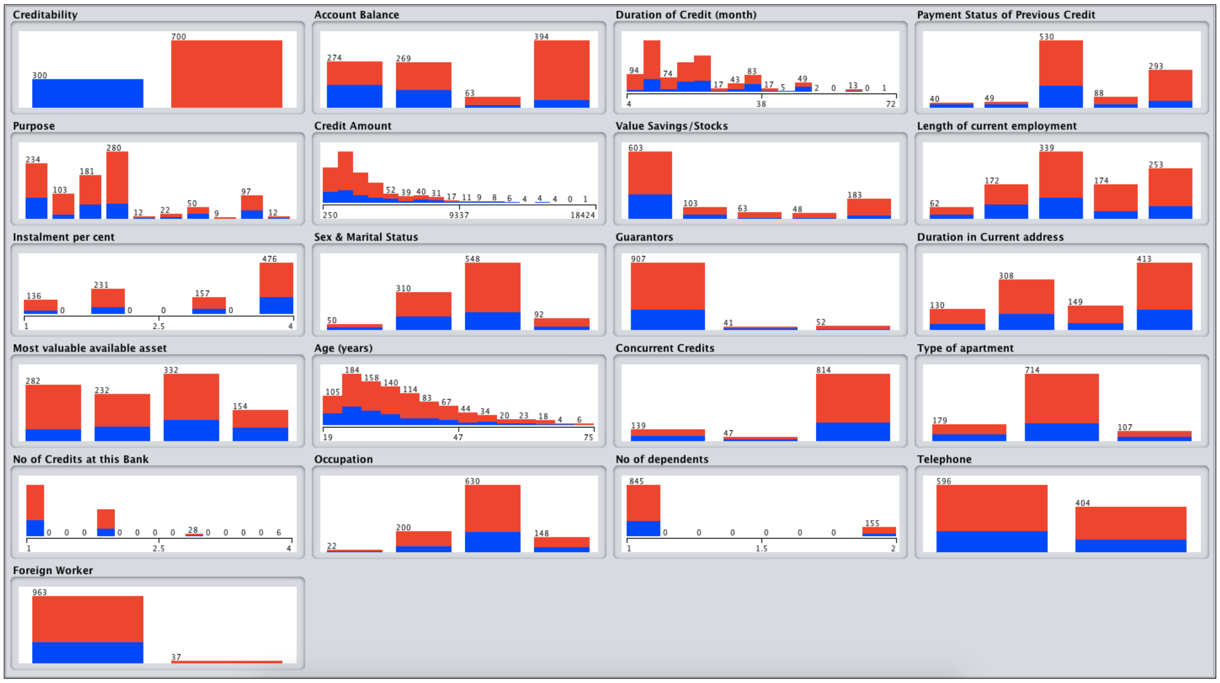


Figure 6: Age Distribution Figure

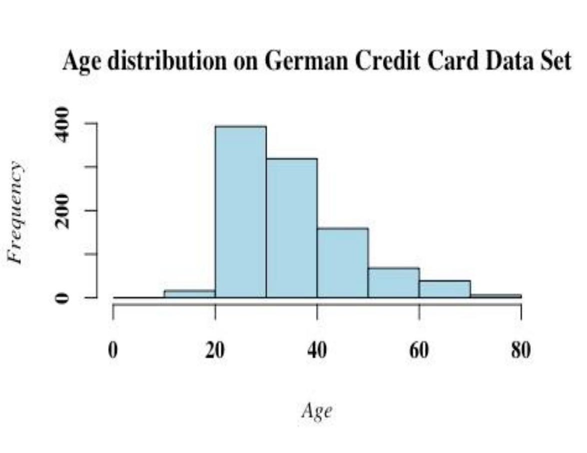


Figure 7: Credit Amount Distribution

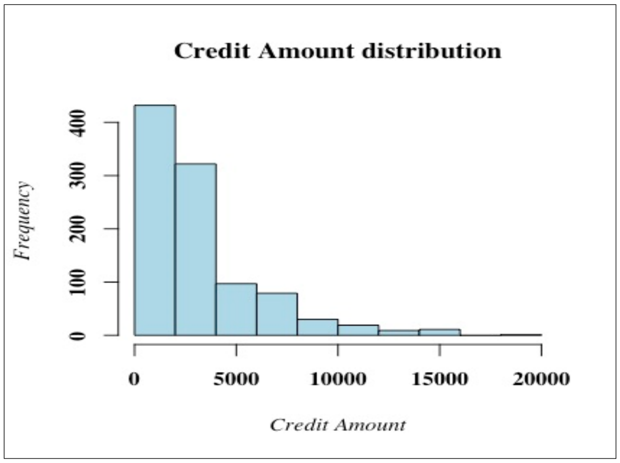


Figure 8: No. of Dependents Distribution

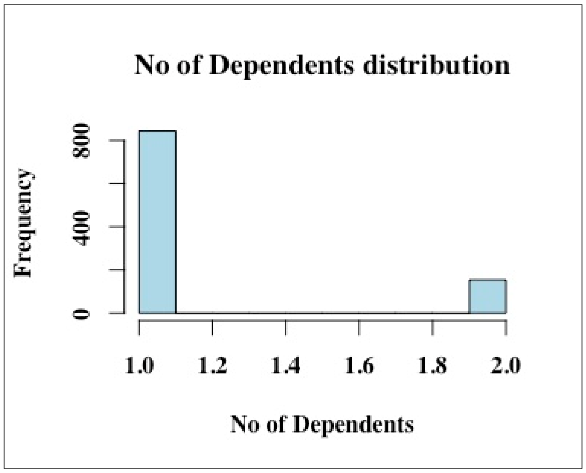
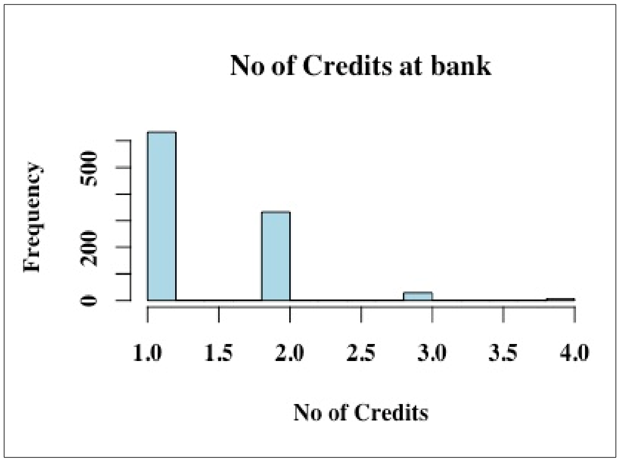


Figure 9: No. of Credits at Bank



1.6. IMBALANCE CLASS DISTRIBUTION

1.6.1. IMBALANCE FEATURE

The total number of high-risk individuals (300) is much lower than the amount of low risk individuals (700). As we apply predictive analytics techniques, we may encounter bias within our model due to the imbalanced dataset. As a result, it is important to balance the dataset in order to build a successful analytical model. Below are multiple methods that can be used to fix this issue. Each strategy has its own variations and will be analyzed in much greater depth as required in the predictive modeling section of the document.

1.6.2. RESAMPLING

Resampling is a widely used technique in data science to balance a dataset by either eliminating instances from the majority class or adding instances to the minority class.

1. Oversampling aims to add random instances to the minority class in order to achieve a more balanced dataset.
   - Random Oversampling
   - SMOTE
2. Under sampling aims to delete records from the majority class when the dataset is large and enough. Through careful analysis, we decided not to pursue this method as our total number of observations are low. Removing any information will cause the model to lose important instances that will hinder the performance of the model.

1.6.3. ALGORITHMIC ENSEMBLE TECHNIQUE

Algorithmic ensemble technique aims to modify existing algorithm classifications to ensure the model incorporates and takes into consideration that the data is imbalanced.

1. Bagging Based techniques for imbalanced data
2. Boosting-Based techniques for imbalanced data
3. Gradient Tree Boosting techniques for imbalanced data
4. Adaptive Boosting- Ada Boost techniques for imbalanced data

1.7. FEATURE SELECTION

Selecting attributes is about narrowing down our potential list to see those attributes that might be most influential in most predictive models. Weka was used to find those patterns that are not always obvious in visualization.
To determine which attributes could be eliminated or included in the analysis, the following list of feature selection methods were performed in Weka (Table 3):

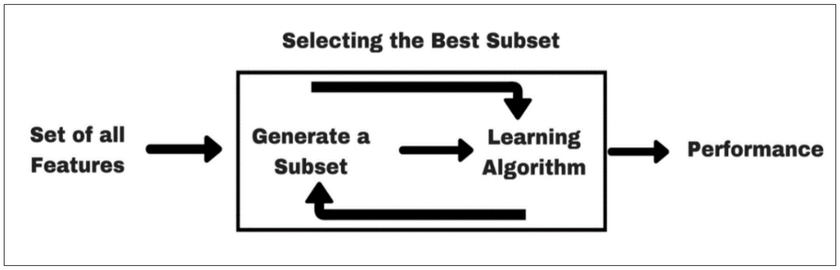
1. **Correlation Based Feature Selection (CorrelationAttributeEval):** This feature evaluates the correlation of all attributes versus Class attribute.

2. **Information Gain Based Feature Selection (InfoGainAttributeEval):** Evaluates the worth of an attribute by measuring the information gain (also known as entropy) with respect to the class attribute. The analysis is performed by using the Ranker InfoGainAttributeEval method. Table 3 shows the ranks for all attributes and the associated normalized values.

3. **Learner Based Feature Selection (Wrapper Method - J48):** This feature evaluates the performance of a subset of features based on the resulting performance of the applied learning algorithm (e.g. what is the gain in accuracy for a classification problem). The WrapperSubsetEval technique was chosen with BestFirst Search Method. As a result, five attributes were selected:

    1. Account Balance
    2. Value Savings/Stocks
    3. Duration of Credit (Month)
    4. Purpose
    5. Payment Status of Previous Credit

Figure 10: Wrapper Method



4. **Chi-Square Evaluator:** A Chi-Square test is used in statistics to test the independence of two events. When two features are independent, the observed count is close to the expected count, therefore we will have a smaller Chi-Square value. So high Chi-Square value indicates that the hypothesis of independence is incorrect. It can be concluded that higher the Chi-Square value, the feature is more dependent on the response and it can be selected for model training.

5. **Symmetrical Uncertainty Evaluation:** Symmetrical uncertainty criterion overcomes the bias of information gain towards the features with the more values by normalizing its value to the range.

Table 3: Feature Selection Numerical Data Summary

Attribute	Correlation	Information Gain	Learner based - J48	Chi-Square	Symmetrical Uncertainty
Account Balance	0.23276	0.094739	1	123.7209	0.070613
Duration of Credit (month)	0.21493	0.0329	2	46.8311	0.030272
Value Savings/Stocks	0.13162	0.028115	4	36.0989	0.021887
Payment Status of Previous Credit	0.08988	0.043618	3	61.6914	0.033641
Credit Amount	0.15474	0.018333		26.3992	0.021448
Type of apartment	0.12283	0.013077		18.674	0.012963
Purpose	0.07494	0.024894		33.3564	0.014033
Length of current employment	0.0527	0.013102		18.3683	0.00863
Instalment per cent	0.0724	0		0	0
Sex & Marital Status	0.07192	0.006811		9.6052	0.005644
Most valuable available asset	0.05838	0.016985		23.7196	0.012008
Age (years)	0.09127	0.011278		16.3681	0.014251
Concurrent Credits	0.108	0.008875		12.8392	0.010284
Occupation	0.01904	0.001337	5	1.8852	0.001166
Foreign Worker	0.08208	0.005823	6	6.737	0.010495
No of dependents	0.00301	0		0	0
No of credit at this Bank	0.04573	0		0	0
Guarantors	0.00612	0.00479		6.6454	0.006758
Duration in current Address	0.01096	0.000543		0.7493	0.000398
Telephone	0.03647	0.000964		1.3298	0.001039

2. PREDICTIVE MODELING / CLASSIFICATION

2.1. INTRODUCTION

The goal was to develop a strategy for the bank managers that can help them in deciding about loan approval for the prospective applicants. Predictive Modeling is a process of using historical data, machine learning and AI to predict what will occur in the future. There are different types of Predictive models such as Classification model, Clustering Model, Forecast model, Outlier model etc. Before applying the Classification Model, the data was randomized using randomization filter in Weka.

2.1.1. CLASSIFICATION

Classification is a two-step process, learning step and prediction step in machine learning. In the learning step, the model is developed based on given training data. In the prediction step, the model is used to predict the response for given data.

1. **Independent variable:** Independent variable is a condition that you change in an experiment. It is the variable that can help predict the output.
2. **Dependent Variable:** Dependent variables represent resulting from changing the inputs/independent variables. In this dataset, Credibility (class attribute) is the dependent variable.

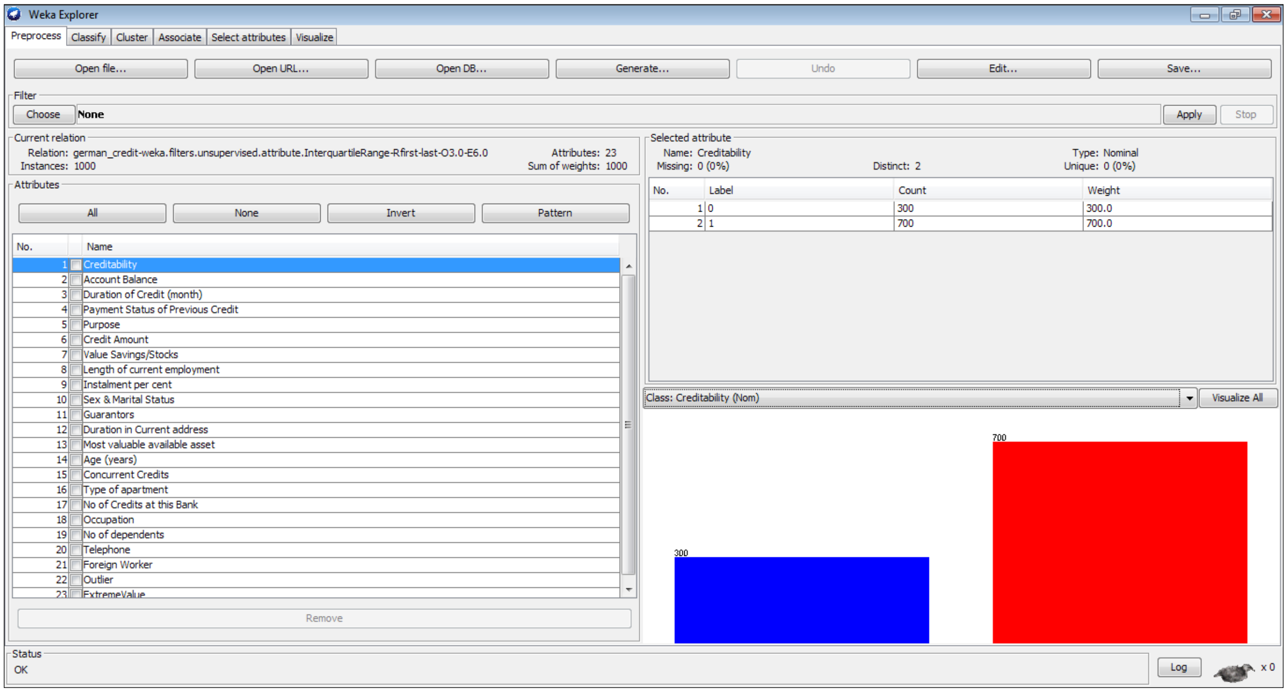
The following three classification algorithms were used:

1. Naïve Bayes
2. Decision Tree
3. Logistic Regression

2.1.2. SMOTE OVERSAMPLING

SMOTE function handles unbalanced classification problems and it generates synthetic dataset that addresses the unbalanced class problem. It artificially generates observations of minority classes using the nearest neighbours of this class of elements to balance the training dataset. With SMOTE, 300 additional records (synthetic data) were added to '0' class attribute. This increased the total number of '0' to 600 and total records to 1300.

Figure 11: Dataset before oversampling (Class 0 - 300 records vs Class 1 - 700 records)



Filter

None

Current relation

Relation: german_credit-weka.filters.unsupervised.attribute.InterquartileRange-Rfirst-last-O3.0-E6.0   Attributes: 23
Instances: 1000   Sum of weights: 1000

Selected attribute

Name: Creditability   Type: Nominal
Missing: 0 (0%)   Distinct: 2   Unique: 0 (0%)

No.	Label	Count	Weight
1	0	300	300.0
2	1	700	700.0

Attributes

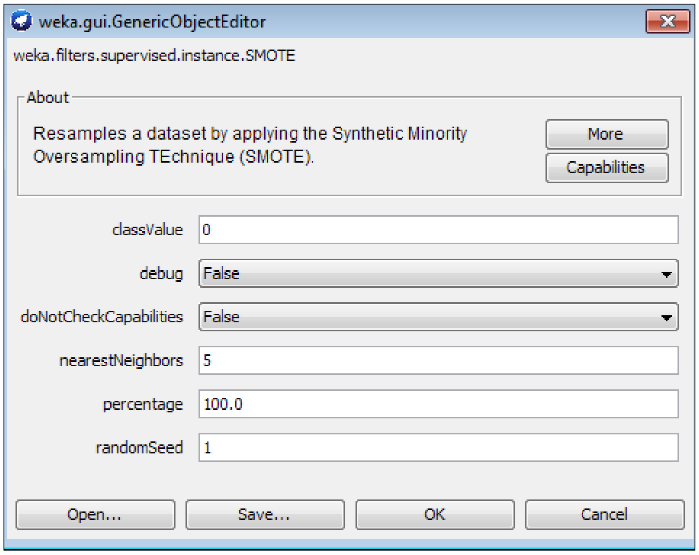
No.		Name
1	□	Creditability
2	□	Account Balance
3	□	Duration of Credit (month)
4	□	Payment Status of Previous Credit
5	□	Purpose
6	□	Credit Amount
7	□	Value Savings/Stocks
8	□	Length of current employment
9	□	Instalment per cent
10	□	Sex & Marital Status
11	□	Guarantors
12	□	Duration in Current address
13	□	Most valuable available asset
14	□	Age (years)
15	□	Concurrent Credits
16	□	Type of apartment
17	□	No of Credits at this Bank
18	□	Occupation
19	□	No of dependents
20	□	Telephone
21	□	Foreign Worker
22	□	Outlier
23	□	ExtremeValue

Class: Creditability (Nom)

Status

OK

Figure 12: SMOTE Filter



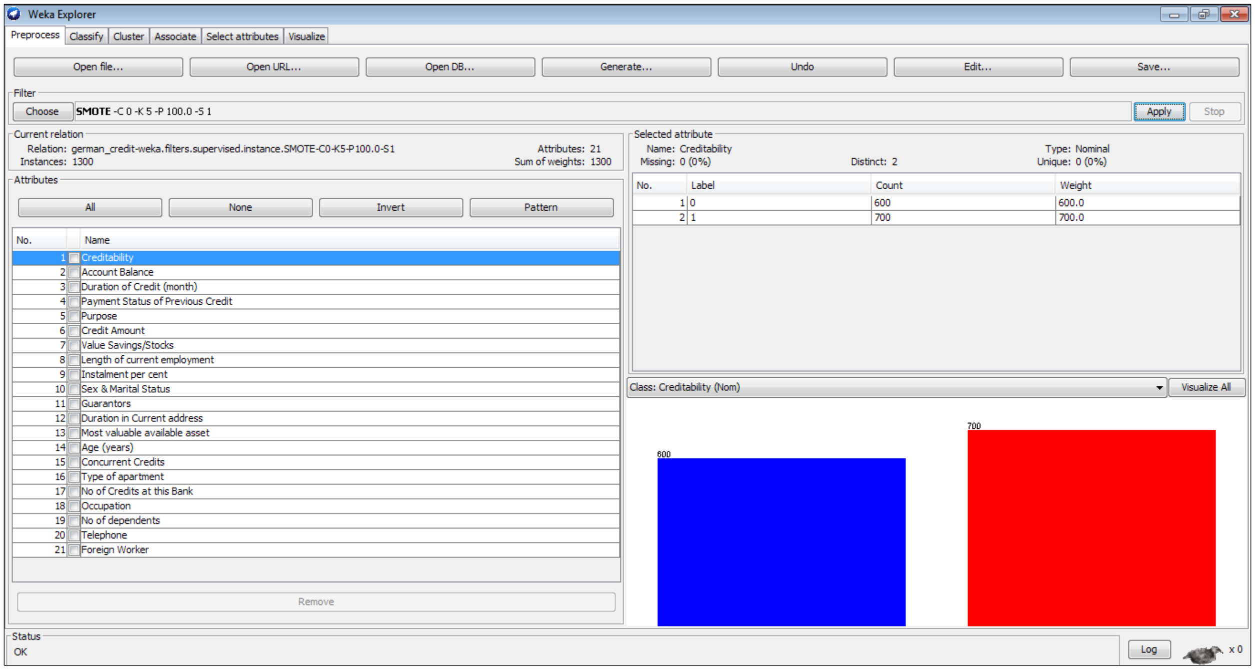
weka.filters.supervised.instance.SMOTE

About

Resamples a dataset by applying the Synthetic Minority Oversampling TEchnique (SMOTE).

classValue	0
debug	False
doNotCheckCapabilities	False
nearestNeighbors	5
percentage	100.0
randomSeed	1

Figure 13: Oversampling using SMOTE filter (Class 0 - 600 records vs Class 1 - 700 records)



Filter

SMOTE -C 0 -K 5 -P 100.0 -S 1

Current relation

Relation: german_credit-weka.filters.supervised.instance.SMOTE-C0-K5-P100.0-S1
Instances: 1300
Attributes: 21
Sum of weights: 1300

Attributes

No.	Name
1	Creditability
2	Account Balance
3	Duration of Credit (month)
4	Payment Status of Previous Credit
5	Purpose
6	Credit Amount
7	Value Savings/Stocks
8	Length of current employment
9	Instalment per cent
10	Sex & Marital Status
11	Guarantors
12	Duration in Current address
13	Most valuable available asset
14	Age (years)
15	Concurrent Credits
16	Type of apartment
17	No of Credits at this Bank
18	Occupation
19	No of dependents
20	Telephone
21	Foreign Worker

Selected attribute

Name: Creditability
Missing: 0 (0%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	0	600	600.0
2	1	700	700.0

Class: Creditability (Nom)

Status

OK

2.1.3. TRAINING AND VALIDATION OF DATASET

Cross-validation is a statistical method used to estimate the skill of the model on new data. The general procedure is as follows:

1. Shuffle the dataset randomly
2. Split the dataset into k groups
3. For each unique group:
    - 3.1 Take the group as a hold out or test data set
    - 3.2 Take the remaining groups as a training data set
    - 3.3 Fit a model on the training set and evaluate it on the test set
    - 3.4 Retain the evaluation score and discard the model
4. Summarize the skill of the model using the sample of model evaluation scores

Percentage split splits the data and separates x% of the data for learning and the rest of it for testing. It is useful when the algorithm is slow. Each set was parameterised using a random selection of 900 or 660 observations and then tested on the remaining 100 or 340.

2.2. NAÏVE BAYES CLASSIFIER

2.2.1. INTRODUCTION

A Naïve Bayes classifier is a probability-based machine learning model which is used for classification task. The classifier is based on the Bayes theorem.

Equation 1: Bayes Theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Using Bayes theorem, we can find the probability of A happening, given that B has occurred. Here, B is the evidence and A is the hypothesis. The assumption made here is that the predictors or features are independent. That is, the presence of one particular feature does not affect the other. Hence it is called naive.

The fundamental Naïve Bayes assumption is that each feature makes an:

1. Independent
2. Equal

contribution to the outcome.

2.2.2. ADVANTAGES OF USING NAÏVE BAYES CLASSIFIER

1. Naïve Bayes is a fast and easy classifier to predict a class of test data set
2. Naïve Bayes Classifier performs better compared to other models assuming independence
3. It performs well in case of categorical input variables compared to numerical variables

2.2.3. RESULTS

Table 4: Unbalanced Data with 20 Attributes (without SMOTE)

Algorithm (Naive Bayes)	**Cross Validation, K= 10,**	**Percentage Split, 60%**	**Percentage Split, 90%**
Correctly Classified instances (%)	75.4%	74.5%	75%
Incorrectly Classified instances (%)	24.6%	25.5%	25%
F-Score	0.746	0.726	0.733
Precision rate	0.743	0.737	0.756
Recall rate	0.754	0.745	0.750

Table 5: Unbalanced Data with 6 Attributes (without SMOTE)

Algorithm (Naive Bayes)	**Cross Validation, K= 10,**	**Percentage Split, 60%**	**Percentage Split, 90%**
Correctly Classified instances (%)	75.4%	74.75%	73%
Incorrectly Classified instances (%)	24.6%	25.25%	27%
F-Score	0.738	0.719	0.698
Precision rate	0.739	0.750	0.754
Recall rate	0.754	0.748	0.730

Table 6: Balanced Data with 20 Attributes (with SMOTE)

Algorithm (Naive Bayes)	Cross Validation, K= 10,	Percentage Split, 60%	Percentage Split, 90%
Correctly Classified instances (%)	80%	80.9%	83.8%
Incorrectly Classified instances (%)	20%	19.8%	16.1%
F-Score	0.800	0.803	0.839
Precision rate	0.800	0.804	0.840
Recall rate	0.800	0.802	0.838

Table 7: Balanced Data with 6 Attributes (with SMOTE)

Algorithm (Naive Bayes)	Cross Validation, K= 10,	Percentage Split, 60%	Percentage Split, 90%
Correctly Classified instances (%)	78.5%	80.9%	83.8%
Incorrectly Classified instances (%)	21.4%	19.8%	16.1%
F-Score	0.786	0.803	0.838
Precision rate	0.786	0.804	0.838
Recall rate	0.785	0.802	0.838

2.2.4. CONCLUSION FOR NAÏVE BAYES CLASSIFICATION MODEL

We implemented Naïve Bayes classifier to evaluate the credit risk in the German credit dataset. A total of 12 tests were performed. 6 tests were performed on unbalanced dataset and 6 tests were performed on balanced dataset. Data was balanced by using SMOTE function. Also 6 tests which were performed on balanced and unbalanced dataset can further classified in 2 sub-groups each; 3 tests with all 20 attributes and 3 tests with top 6 attributes.

Two types of testing techniques were used, the 10-fold test set and percentage split, where data was split into the ratio of 60:40 and 90:10 i.e. 90% of the data was used to train the model and 10% of the data is used to validate the model.

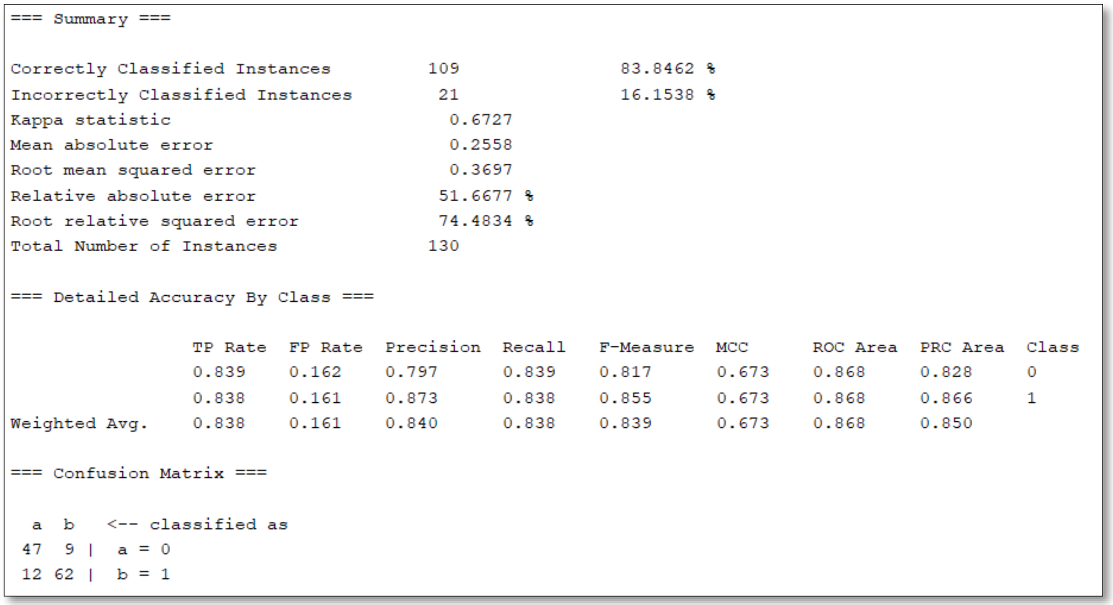
In our first test, we used all 20 attributes and unbalanced dataset. The maximum accuracy of 75.4% was achieved with 10-fold test, 74.5% and 75% with both 60% and 90% percentage split respectively. In second test, we used unbalanced dataset but with top 6 attributes only. The maximum accuracy was again 75.4% with 10-fold test and 74.75% with 60% percentage split. Overall there was not much difference observed in terms of accuracy in unbalanced dataset for 20 attributes and top 6 attributes.

In the next test, we used balanced dataset with 20 attributes and accuracy observed was 83.8% with 90% percentage split. This was the maximum accuracy we observed so far. Similar results were observed when we used balanced dataset with 6 attributes. 83.8% accuracy was observed with 90% percentage split and 78.5% accuracy observed in 10-fold test set. Higher accuracy in 90% split can be attributed to overfitting of test model on relatively small dataset.

In conclusion, balanced dataset (SMOTE) with 20 attributes, in 60% percentage split test yields the statistically significant accuracy compared to 90% split. Test performed with 6 attributes did not yield statistically significant accuracy.

2.2.5. SAMPLE OUTCOME OF NAÏVE BAYES

Figure 14: Balanced Data with 20 Attributes (Percentage Spilt 90%)



```
=== Summary ===

Correctly Classified Instances         109               83.8462 %
Incorrectly Classified Instances        21               16.1538 %
Kappa statistic                          0.6727
Mean absolute error                      0.2558
Root mean squared error                  0.3697
Relative absolute error                 51.6677 %
Root relative squared error             74.4834 %
Total Number of Instances              130

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.839    0.162    0.797      0.839    0.817      0.673    0.868     0.828     0
                 0.838    0.161    0.873      0.838    0.855      0.673    0.868     0.866     1
Weighted Avg.    0.838    0.161    0.840      0.838    0.839      0.673    0.868     0.850

=== Confusion Matrix ===

  a  b   <-- classified as
 47  9 |  a = 0
 12 62 |  b = 1
```

2.3. DECISION TREE

Decision Tree is a supervised learning algorithm. It is used for solving Regression and Classification problems. In this case, we used Decision Tree algorithm to solve a classification problem. **J48 classifier was used in Weka** for Decision Tree. Data is split into Nodes and Leaves, where Leaves can be used to predict the output. In all our decision trees, Account Balance was the root node and we can conclude that this parameter had the highest gain among all the parameters.

2.3.1. ATTRIBUTE SELECTION

As explained in the Feature selection section, various tests were performed to select the attributes that could yield the best results. We conducted tests using J48 classifier by choosing all 20 independent variable and also used the below attributes to predict the output.

The top 6 attributes to predict the class variable:

1. Account Balance
2. Credit Amount
3. Duration of Credit (month)
4. Value Savings/Stocks
5. Payment Status of Previous Credit
6. Most valuable available asset

2.3.2. PRUNING

Decision Trees tend to perform well with training data but perform poorly on training data. This can be corrected using Pruning. **Unpruned** parameter in Weka is set to False. This means that pruning is enabled by default. **ConfidenceFactor** is used to configure. A small confidence value corresponds to heavy pruning, a large one to little pruning. The values of Confidence Factor can be between 0 and 1. The table below shows the impact of changing confidence factor on accuracy and number of leaves of a decision tree.

Table 8: Accuracy and Number of leaves

Confidence factor	Accuracy	Comments
0.25	72.8	Number of Leaves: 99 Size of the tree: 136
0.5	72.1	Number of Leaves: 180 Size of the tree: 245
0.9	70.2	Number of Leaves: 336 Size of the tree: 439

2.3.3. SMOTE: Synthetic Minority Over-sampling Technique (SMOTe)

As the data set has only 1000 records, we decide to ignore the case where the decision tree would under-fit. In order to explore the condition of the decision tree overfitting, we decided to use SMOTE. With SMOTE we introduced 300 additional records (synthetic data) to '0' class attribute. This bought the total number of '0' to 600 and total records to 1300. Our observation that the parameters of accuracy, precision and recall improve, after synthetic data was introduced.

2.3.4. ENSEMBLE METHODS

Ensemble methods are meta-algorithms that combine several machine learning techniques into one predictive model in order to decrease variance (bagging), bias (boosting), or improve predictions (stacking). An ensemble method combines the predictions from multiple machine learning algorithms together to make more accurate predictions than any individual model. Decision Tree suffer from variance and bias. In order to compensate for variance, we used one of the ensemble techniques - boosting technique to check if this improved accuracy.

To reduce the variance and amplify the robustness of the model, Bagging and Boosting can be used. Bagging was achieved by using **Bagging Classifier in Weka** and selecting J48 as classifier.

2.3.5. RESULTS & O/P PARAMETERS CONSIDERED TO DETERMINE PERFORMANCE OF DECISION TREE

Below is the list of parameters that were the output of decision tree.

1. Accuracy
2. Weighted Precision
3. Weighted Recall
4. Weighted F measures

Below are the results of all the tests that were conducted. We chose Accuracy as the primary factor to determine the performance of Decision Tree.

Table 10: Decision Tree Tests and Results

No	Parameters Used	No of Attributes	Test Data Method	Test Data Size	Accuracy (%)	Weighted Precision	Weighted Recall	Weighted F Measure	Confusion Matrix
1	J48 M2 C0.25	20	Cross Validation - 10	1000	72.8	0.712	0.728	0.716	a b <-- classified as 128 172 | a = 0 100 600 | b = 1
2	J48 M2 C0.25	20	Percentage Split - 60%	400	71.75	0.706	0.718	0.687	a b <-- classified as 45 90 | a = 0 23 242 | b = 1
3	weka.classifiers.meta.**Bagging** -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2	20	Percentage Split - 60%	400	70.5	0.691	0.705	0.666	a b <-- classified as 38 97 | a = 0 21 244 | b = 1
4	weka.classifiers.meta.**Bagging** -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2	20	Cross Validation - 10	1000	74.3	0.729	0.743	0.731	a b <-- classified as 134 166 | a = 0 91 609 | b = 1
5	weka.classifiers.trees.J48 -C 0.25 -M 2 (with **SMOTE**)	20	Cross Validation - 10	1300	75.2308	0.752	0.752	0.752	a b <-- classified as 430 170 | a = 0 152 548 | b = 1
6	weka.classifiers.trees.J48 -C 0.25 -M 2 (with **SMOTE**)	20	Percentage Split - 60%	520	78.4615	0.792	0.785	0.786	a b <-- classified as 177 42 | a = 0 70 231 | b = 1
7	weka.classifiers.meta.**Bagging** -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2 (With **SMOTE**)	20	Percentage Split - 60%	520	**79.4231**	0.798	0.794	0.795	a b <-- classified as 175 44 | a = 0 63 238 | b = 1
8	weka.classifiers.meta.**Bagging** -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2 (With **SMOTE**)	20	Cross Validation - 10	1300	78.8462	0.778	0.758	0.768	a b <-- classified as 455 145 | a = 0 130 570 | b = 1
9	weka.classifiers.trees.J48 -C 0.25 -M 2	6	Cross Validation - 10	1000	72.8	0.712	0.728	0.716	a b <-- classified as 128 172 | a = 0 100 600 | b = 1
10	weka.classifiers.trees.J48 -C 0.25 -M 2	6	Percentage Split - 60%	400	70.5	0.691	0.705	0.666	a b <-- classified as 38 97 | a = 0 21 244 | b = 1
11	weka.classifiers.meta.**Bagging** -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2	6	Cross Validation - 10	1000	74.3	0.729	0.743	0.731	a b <-- classified as 134 166 | a = 0 91 609 | b = 1
12	weka.classifiers.meta.**Bagging** -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2	6	Percentage Split - 60%	400	70.5	0.691	0.705	0.666	a b <-- classified as 38 97 | a = 0 21 244 | b = 1

13	weka.classifiers.trees.J48 -C 0.25 -M 2 (with **SMOTE**)	**6**	Cross Validation - 10	1300	77.6154	0.777	0.776	0.776	a b <-- classified as 464 136 | a = 0 155 545 | b = 1
14	weka.classifiers.trees.J48 -C 0.25 -M 2 (with **SMOTE**)	6	Percentage Split - 60%	520	77.1154	0.781	0.771	0.773	a b <-- classified as 177 42 | a = 0 77 224 | b = 1
15	weka.classifiers.meta.**Bagging** -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2 (With **SMOTE**)	6	Percentage Split - 60%	520	**78.4615**	0.791	0.785	0.786	a b <-- classified as 176 43 | a = 0 69 232 | b = 1
16	weka.classifiers.meta.**Bagging** -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2 (With **SMOTE**)	6	Cross Validation - 10	1300	**77.6154**	0.776	0.776	0.776	a b <-- classified as 448 152 | a = 0 139 561 | b = 1

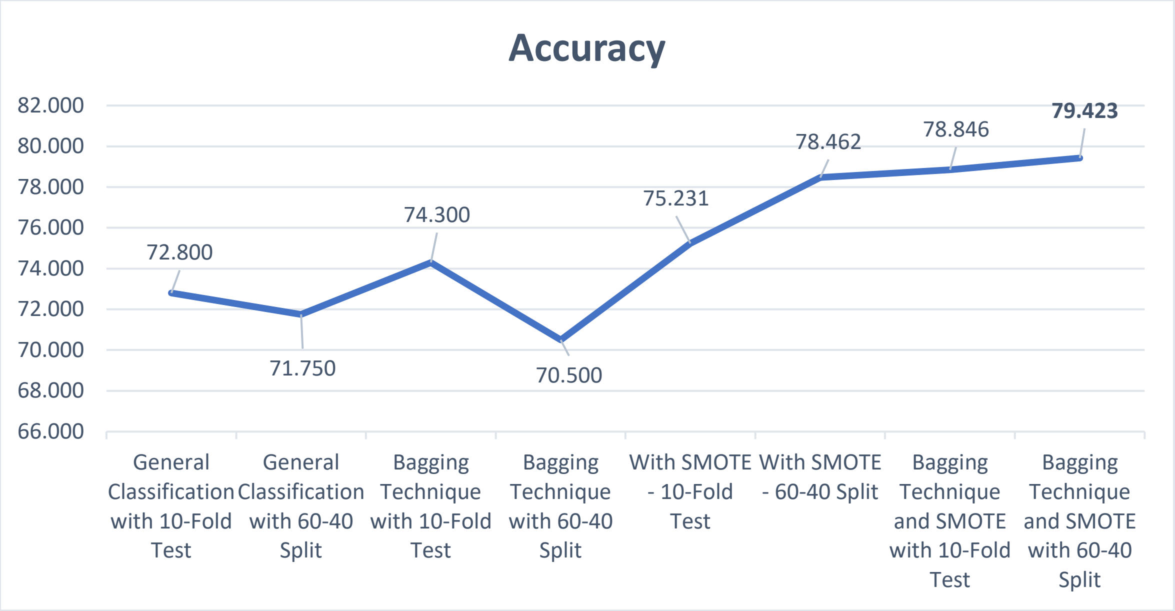
Table 10: Results of tests conducted on 20 Attributes

Parameters Used	No of Attributes	Accuracy
General Classification with 10-Fold Test	20	72.80%
General Classification with 60-40 Split	20	71.75
Bagging Technique with 10-Fold Test	20	74.3
Bagging Technique with 60-40 Split	20	70.5
With SMOTE - 10-Fold Test	20	75.2308
With SMOTE - 60-40 Split	20	78.4615
Bagging Technique and SMOTE with 10-Fold Test	20	78.8462
Bagging Technique and SMOTE with 60-40 Split	20	**79.4231**

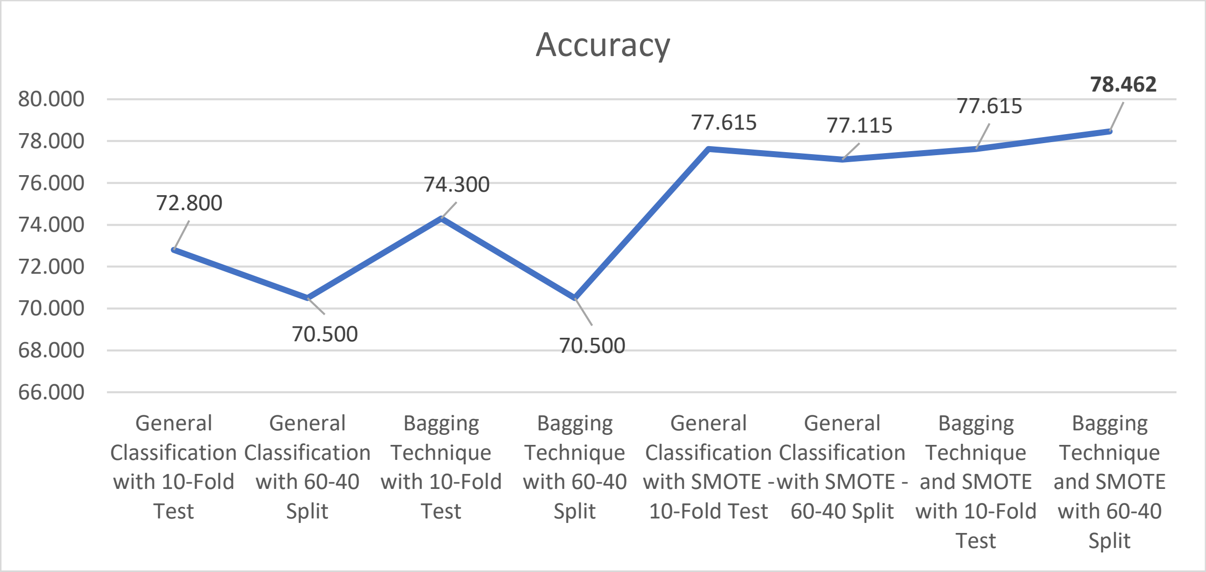
Table 11: Results of tests conducted on 6 Attributes

Parameters Used	No of Attributes	Accuracy
General Classification with 10-Fold Test	6	77.6154
General Classification with 60-40 Split	6	77.1154
Bagging Technique with 10-Fold Test	6	77.6154
Bagging Technique with 60-40 Split	6	**78.4615**
With SMOTE - 10-Fold Test	6	74.2
With SMOTE - 60-40 Split	6	74.5
Bagging Technique and SMOTE with 10-Fold Test	6	74.2
Bagging Technique and SMOTE with 60-40 Split	6	74.75

Graph 1: Changes in Accuracy with 20 parameters



Graph 2: Changes in Accuracy with 6 parameters



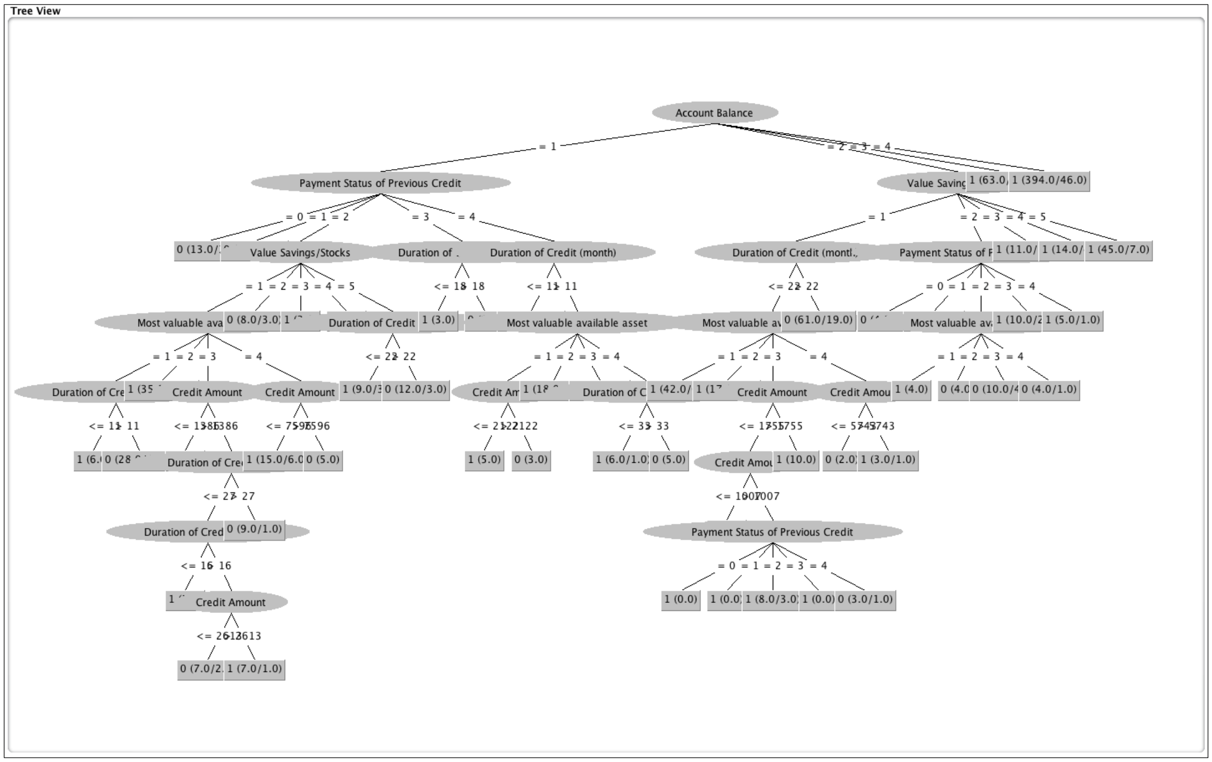
2.3.6. CONCLUSIONS FOR DECISION TREE CLASSIFICATION MODEL

A total of 16 tests were performed. Firstly, we used all 20 independent variables to construct a decision tree. Two types of testing techniques were used, the 10-fold test and Percentage Split, where data was split into the ratio of 60:40 i.e. 60% of the data was used to training the model and 40% of the data was used test the model. We achieved an accuracy of 72.8% with 10-fold test and 71.8% when percentage split was used. Next we carried the same tests, by introducing synthetic data via SMOTE technique. Introduction of SMOTE improved the accuracy of the model. We were able to achieve much better results. 78.4 % accuracy was achieved by using J48 algorithm on a total of 530 test records. Introduction of SMOTE might have caused the model to overfit. We wanted to check this, hence we used Bagging technique. Highest accuracy of 79.4% was the output of Bagging + SMOTE + Percentage Split Test.

The same tests were then carried out by reducing the attributes to 6. The top 6 attributes were chosen by performing feature selection tests as part of Data Preparation. Very similar results as that of 20 attributes were observed. Highest accuracy of 78.4% was achieved by using combination of Bagging + SMOTE + Percentage Split Test. We can conclude that reducing the number of parameters to 6, did not cause any improvement in the classification of the model in this case.

2.3.7. DECISION TREE SAMPLE

Figure 15: Decision Tree in Weka

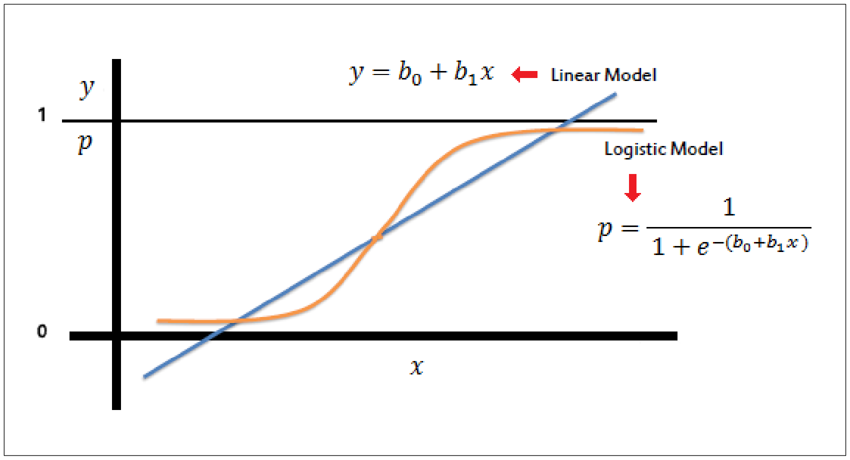


2.4. LOGISTIC REGRESSION

As a third algorithm, we decided to use logistic regression. Much like linear regression, logistic regression is a linear method, but the predictions are transformed using the logistic function in order to get a probability between 0 and 1. A threshold is then decided upon to classify each instance into either class (good credit/bad credit). We decided to keep the threshold at 0.5.

Consequently, any output of the logistic function that returns a value greater than 0.5 will be classified as good credit individual

Figure 16: Logistic Model



2.4.1. ATTRIBUTE SELECTION AND TRAINING STRATEGY

We decided to run our model multiple times, each time by altering a different parameter. Firstly, the experiment included all 20 attributes followed by the main 6 attributes chosen. Each time, the data was split using two strategies. Finally, cost sensitive learning will be applied as a measure to reduce the imbalance feature of our dataset.

Table 12: Training Strategy and top 6 Attributes

Top 6 attributes	Training Strategy
Account Balance	10-fold cross validation
Duration of Credit (month)	60-40 percentage split
Payment Status of Previous Credit	
Credit Amount	
Value Savings/Stocks	
Most valuable available asset	

2.4.2. COST SENSITIVE LEARNING

In machine learning, most learning algorithms assume that miss classification costs made by a classifier are equal. However, in our situation, this is not the case. As a bank, the cost of lending out a loan to a bad customer (false positive) can result in much greater losses than denying a loan to a good customer (false negative). Cost sensitive learning is a subfield of machine learning that takes into consideration the cost of prediction errors when training the model. When training our model, we will be changing our cost matrix by altering the false positive error weight from 1 to 2. With a goal to **minimize the bank's error costs**, our training algorithm will be much more careful in identifying bad credit individuals. Below are the results split into two tables:

Table 13: Results of Logistic Regression on 20 Attributes

S. No	Parameters Used	No of Attributes	Cost Matrix	Test Data Method	Test Data Size	Accuracy (Correctly Classified Instances)	Weighted Precision	Weighted Recall	Weighted F Measure	Confusion Matrix
1	Logistic regression	20	NA	10-fold	1000	75.20%	0.742	0.752	0.745	a b <-- classified as 150 150 | a = 0 98 602 | b = 1
2	Logistic Regression	20	NA	60-40%	400	73.50%	0.725	0.735	0.725	a b <-- classified as 66 69 | a = 0 37 228 | b = 1
3	Logistic regression with cost sensitive learning	20	Cost Matrix 0 2 1 0	10-fold	1000	72.40%	0.745	0.724	0.731	a b <-- classified as 196 104 | a = 0 172 528 | b = 1
4	Logistic regression with cost sensitive learning	20	Cost Matrix 0 2 1 0	60-40%	400	70.50%	0.713	0.705	0.708	a b <-- classified as 83 52 | a = 0 66 199 | b = 1

Table 14: Results of Logistic Regression on Top 6 Attributes

S. No	Parameters Used	No of Attributes	Cost Matrix	Test Data Method	Test Data Size	Accuracy	Weighted Precision	Weighted Recall	Weighted F Measure	Confusion Matrix
1	Logistic Regression	6	NA	10-fold	1000	74.20%	0.725	0.742	0.725	a b <-- classified as 122 178 | a = 0 80 620 | b = 1
2	Logistic Regression	6	NA	60-40%	400	72.67%	0.725	0.738	0.718	a b <-- classified as 43 63 | a = 0 13 181 | b = 1
3	Logistic regression with cost sensitive learning	6	Cost Matrix 0 2 1 0	10-fold	1000	71.70%	0.747	0.717	0.726	a b <-- classified as 205 95 | a = 0 188 512 | b = 1
4	Logistic regression with cost sensitive learning	6	Cost Matrix 0 2 1 0	60-40%	400	71.67%	0.721	0.708	0.721	a b <-- classified as 72 47 | a = 0 42 152 | b = 1

2.4.3. MODEL EVALUATION CRITERIA

There are a variety of different methods to evaluate a machine learning model. We decided to look at several different criteria's in combination rather than using one of them exclusively.

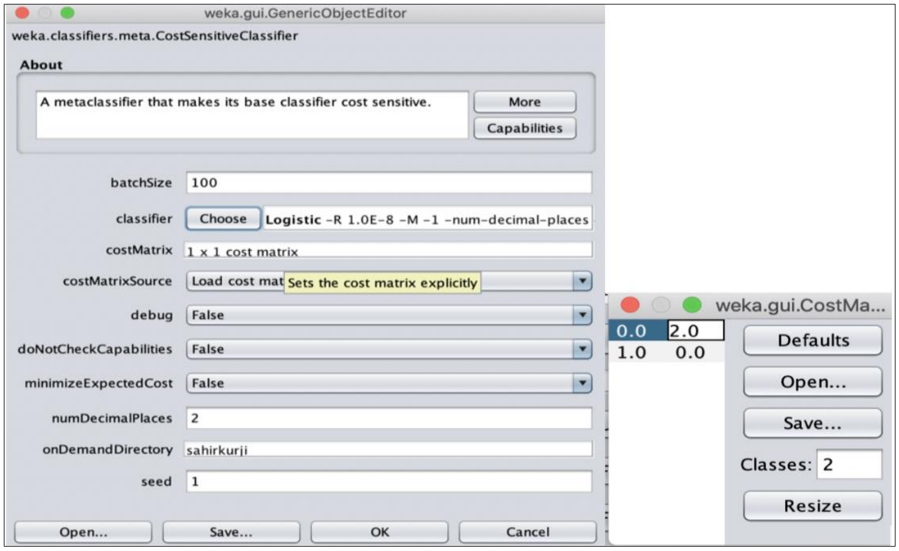
1. Accuracy (Correctly Classified Instances)
2. Weighted Recall
3. Weighted Precision
4. Weighted F Measure

2.4.4. WEKA PARAMETERS

In order to perform logistic regression, the following steps were performed to achieve our results. Firstly, the dataset was uploaded onto Weka. In the classifier section, under the function's dropdown, logistic was chosen. All other parameters remained the same. The test was then run with both training strategies and sets of attributes.

In order to make use of the cost sensitive function, the metaclassifier CostSensitiveClassifier was chosen. In the parameters section, the logistic algorithm was chosen, and the cost matrix was updated.

Figure 17: CostSensitiveClassifier in Weka



weka.gui.GenericObjectEditor

weka.classifiers.meta.CostSensitiveClassifier

About

A metaclassifier that makes its base classifier cost sensitive.

batchSize	100
classifier	Logistic -R 1.0E-8 -M -1 -num-decimal-places
costMatrix	1 x 1 cost matrix
costMatrixSource	Load cost mat Sets the cost matrix explicitly
debug	False
doNotCheckCapabilities	False
minimizeExpectedCost	False
numDecimalPlaces	2
onDemandDirectory	sahirkurji
seed	1

weka.gui.CostMa...

0.0	2.0
1.0	0.0

Classes: 2

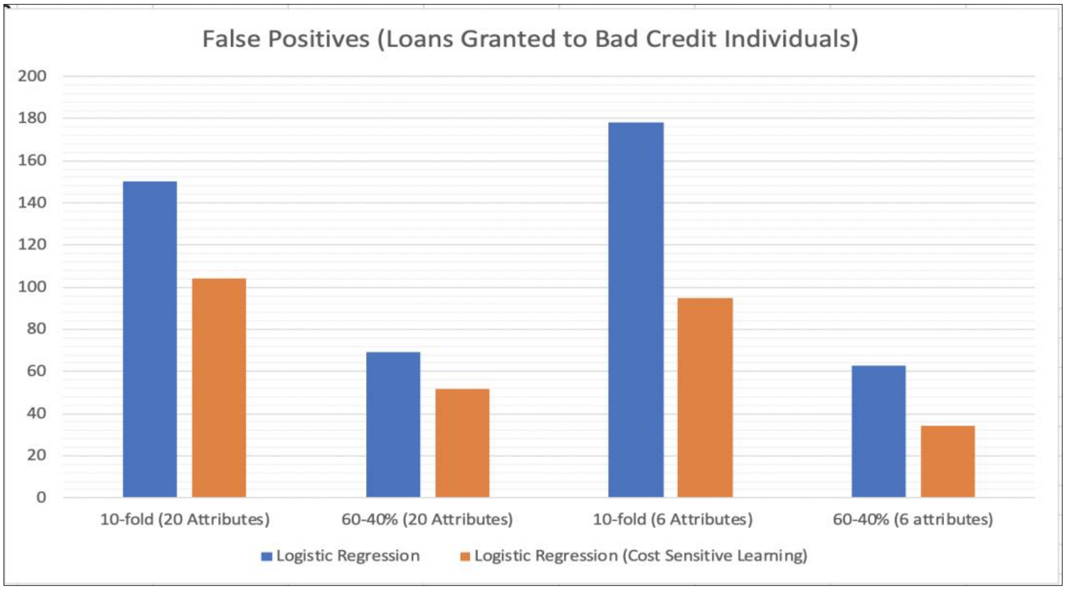
2.4.5. MODEL EVALUATION

From Table 13 and 14, we notice that all instances and measuring criteria's range in a small window between 70 and 75%. The yellow boxes illustrate the highest performance for each decision criteria. As we can see, there is no model that clearly outperforms the other. However, when analyzing the results more carefully, there are multiple observations worth noting. Firstly, using our evaluation criteria, we can see that the best strategy to split the dataset is 10-fold cross validation rather than a simple percentage split. This is shown in the table where all instances have a higher accuracy using the cross-validation training strategy. The f-score also confirms the same.

Another important observation lies within the confusion matrices. Although we don't see an improvement in the 5 criteria's using a cost sensitive strategy, the number of false positives does reduce (instances classified as b=1 but are actually a=0; loans given out to bad credit customers).

The aim of using a cost sensitive model is to minimize the overall **error cost** and not optimize the correctness of classifications. This is a function that includes the probability of default as well as the cost of default to the bank. When analyzing the ROC chart and Area Under Curve, there is relatively no difference when applying a cost sensitive filter. As a result, we can come to the conclusion that applying a cost sensitive strategy will not improve the **weighted** accuracy, precision and recall of the model but does improve the false positive rate of the model. In simple terms, this means that out of all the bad credit individuals, the number of misclassifications is lower. For example, comparing the first experiment to the 3rd, the false positive rate goes down from 50% to 34%. However, with this improvement, we also see a reduction in the true positive rate from 86% to 75%. This highlights the inverse relationship between optimizing the false positive rate and true positive rate. When reducing the false positive rate, we tend to reduce the true positive rate as well. Ultimately, by using the cost sensitive classifier, we are not necessarily improving the performance of the model. We are shifting the risk away from giving loans to bad customers at the expense of missing out on potential good clients. This is done because the cost of default is much greater to the bank than the cost of denying a good client. Ultimately, the cost sensitive model should prove to be more profitable.

Graph 3: Logistic Regression Results Comparison



2.5. CLASSIFICATION ALGORITHM COMPARISION

Table 15: Comparison of All Classification Algorithms

No	Algorithm	Testing Method	Number of Instances	Accuracy	Precision	Recall	F-Measure
1	Naive Bayes	10-Fold Test	1000	**75.4**	0.743	0.754	0.746
2	Naive Bayes	60-40 Split	400	74.5	0.737	0.745	0.726
3	Decision Tree	10-Fold Test	1000	72.8	0.712	0.728	0.716
4	Decision Tree	60-40 Split	400	71.75	0.706	0.718	0.687
5	Logistic Regression	10-Fold Test	1000	75.2	0.742	0.752	0.745
6	Logistic Regression	60-40 Split	400	73.5	0.725	0.735	0.725

Although we tested the dataset on various techniques and classification algorithms, we decided to draw a general conclusion on performance by comparing results of Naïve Bayes, Logistic Regression and Decision Tree by comparing the results of test performed on all 20 attributes. Table above summarize the performance on two different testing techniques. For simplicity reasons, we used accuracy as our main measure of comparison, and It can be observed that Naïve Bayes classification method used in 10-fold test setting provides higher accuracy of credit risk evaluation.

It should be also noted that 10-Fold Test yielded better results as compared to testing using percentage split.

3. POST PREDICTIVE ANALYSIS

Post Predictive Analysis was performed to identify the characteristics of customer whose loan was approved as well as the characteristics of those who got disapproved. Below are the different strategies that were used:

1. K-Means
2. Apriori

3.1. K-MEANS CLUSTERING

Clustering technique is used to find homogeneous subgroups within the data set such that data points in each cluster are as similar as possible. Clustering is an unsupervised technique. K-Means in one of the algorithms that is used for cluttering data.

This algorithm tries to partition the dataset into k pre-defined clusters. It tries to make the points within a cluster as similar as possible at the same time, try to keep the inter-cluster distance as far as possible. It uses the sum of least squares technique – the sum of distances between data points and the clusters centroid is at the minimum.

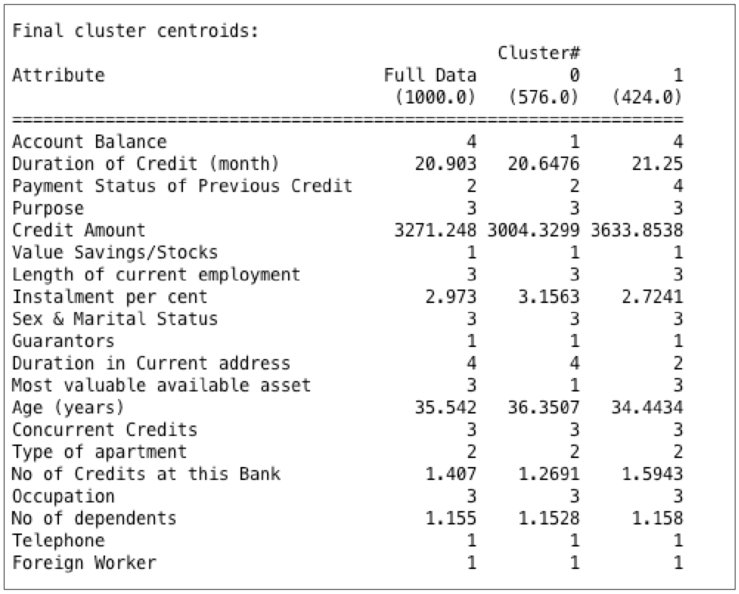
SimpleKmeans is available under the Cluster category in Weka. There are three different tests that we performed using this algorithm.

1. Class to Cluster evaluation using all 1000 instances
2. Test on only those who were credit approved (700)
3. Test on only those who were disapproved (300)

3.1.1. CLASS TO CLUSTER EVALUATION

This test was performed by selecting the ‘Class to cluster evaluation’ in Weka to verify how K-Means would segregate the data based on the class attribute (Credibility). The *distanceFunction* was defaulted to EuclideanDistance and the *numClusters* parameters was set to 2.

Figure 18: Results of K Means segregation on class attribute

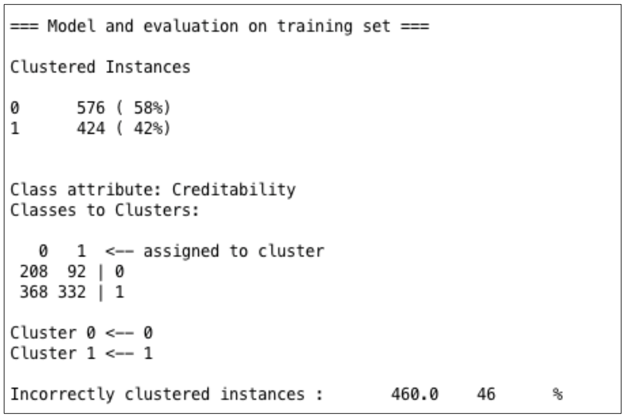


```
Final cluster centroids:
                                                  Cluster#
Attribute                              Full Data         0          1
                                        (1000.0)   (576.0)    (424.0)
======================================================================
Account Balance                                4         1          4
Duration of Credit (month)                20.903   20.6476      21.25
Payment Status of Previous Credit              2         2          4
Purpose                                        3         3          3
Credit Amount                           3271.248 3004.3299  3633.8538
Value Savings/Stocks                           1         1          1
Length of current employment                   3         3          3
Instalment per cent                        2.973    3.1563     2.7241
Sex & Marital Status                           3         3          3
Guarantors                                     1         1          1
Duration in Current address                    4         4          2
Most valuable available asset                  3         1          3
Age (years)                               35.542   36.3507    34.4434
Concurrent Credits                             3         3          3
Type of apartment                              2         2          2
No of Credits at this Bank                 1.407    1.2691     1.5943
Occupation                                     3         3          3
No of dependents                           1.155    1.1528      1.158
Telephone                                      1         1          1
Foreign Worker                                 1         1          1
```

The algorithm was able to split the data into two clusters based on the credibility class, but the accuracy of this classification was very low as shown in Figure 20, 46% of the instances were classified incorrectly.

Based on classification that was performed, we observed that the following characteristics are different between the two clusters; Account Balance, Payment Status of Previous Credit, Credit Amount, Instalment per cent, Duration in Current address and most valuable available asset. We can therefore infer that these characteristics are the differentiating factors within our experiment. It is important to note that the accuracy of this experiment was low at 55%. As a result, we decided to run multiple tests that will be explained below in order to get a better holistic view.

Figure 19: Classification Metrics of K means

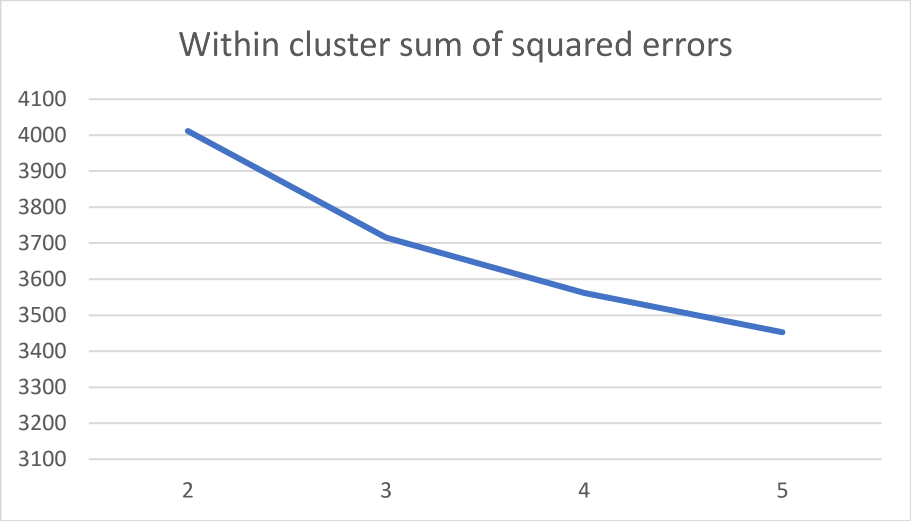


```
=== Model and evaluation on training set ===

Clustered Instances

0       576 ( 58%)
1       424 ( 42%)


Class attribute: Creditability
Classes to Clusters:

   0   1  <-- assigned to cluster
 208  92 | 0
 368 332 | 1

Cluster 0 <-- 0
Cluster 1 <-- 1

Incorrectly clustered instances :       460.0    46      %
```

3.1.2. TEST TO DETERMINE THE CHARACTERISTICS OF GOOD CREDIT RATING

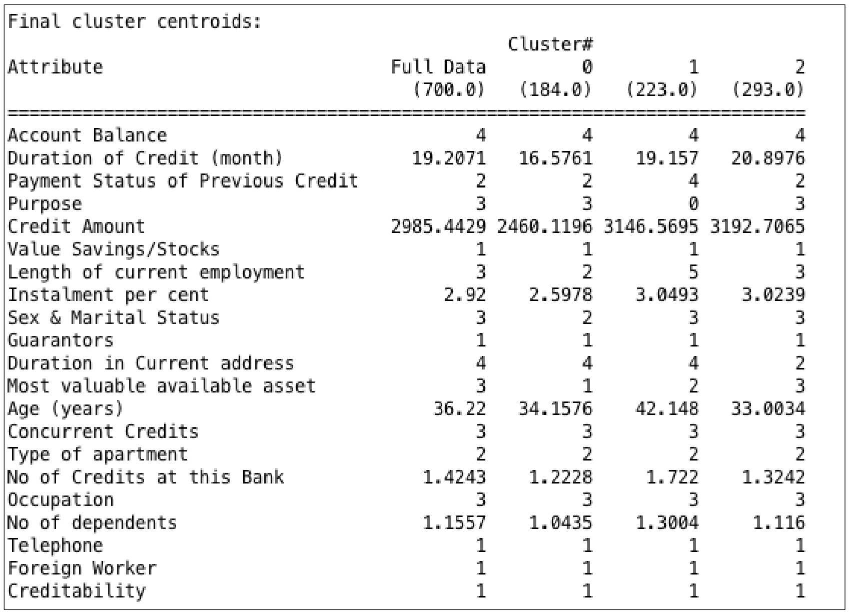
We leveraged the RemoveWithValues feature in Weka to retain only the 700 records (credit approved). The number of clusters were determined by using “Elbow method”.

Graph 4: Elbow method to determine number of clusters



For simplicity measures, we have decided to segregate the good credit individuals into 3 different clusters.

Figure 20: K Means Cluster output for Good Credit data



```
Final cluster centroids:
                                                  Cluster#
Attribute                             Full Data          0          1          2
                                        (700.0)    (184.0)    (223.0)    (293.0)
=================================================================================
Account Balance                               4          4          4          4
Duration of Credit (month)              19.2071    16.5761     19.157    20.8976
Payment Status of Previous Credit             2          2          4          2
Purpose                                       3          3          0          3
Credit Amount                         2985.4429  2460.1196  3146.5695  3192.7065
Value Savings/Stocks                          1          1          1          1
Length of current employment                  3          2          5          3
Instalment per cent                        2.92     2.5978     3.0493     3.0239
Sex & Marital Status                          3          2          3          3
Guarantors                                    1          1          1          1
Duration in Current address                   4          4          4          2
Most valuable available asset                 3          1          2          3
Age (years)                               36.22    34.1576     42.148    33.0034
Concurrent Credits                            3          3          3          3
Type of apartment                             2          2          2          2
No of Credits at this Bank               1.4243     1.2228      1.722     1.3242
Occupation                                    3          3          3          3
No of dependents                         1.1557     1.0435     1.3004      1.116
Telephone                                     1          1          1          1
Foreign Worker                                1          1          1          1
Creditability                                 1          1          1          1
```

There are several observations that can be made when analyzing the results. Firstly, we see that all individuals who were granted a loan do not have a checking at the banks. Perhaps the bank is trying to attract new customers by approving their loans. Next, across most instances, credit approved individuals had existing credits paid back duly. The bank therefore considers these individuals to have the capacity to remain in good credit over the duration of credit.

In addition, we see that individuals who don't need guarantors are less risky and therefore more likely to get approved. Finally, we see that the bank considers the occupation of an individual as important. Across all clusters, credit approved individuals are highly skilled.

When looking within each cluster, we can we can classify the clusters, based on the following factors: Age, Duration of Credit, Most Valuable Asset, Credit Amount and purpose. The mean value of all other attributes remained the same in the clusters, so we ignored these attributes.

Cluster 0 consists of middle-aged people who seek Lower credit amount and seek loans for a short duration of time. Another characteristic of this population is that they had one valuable asset.

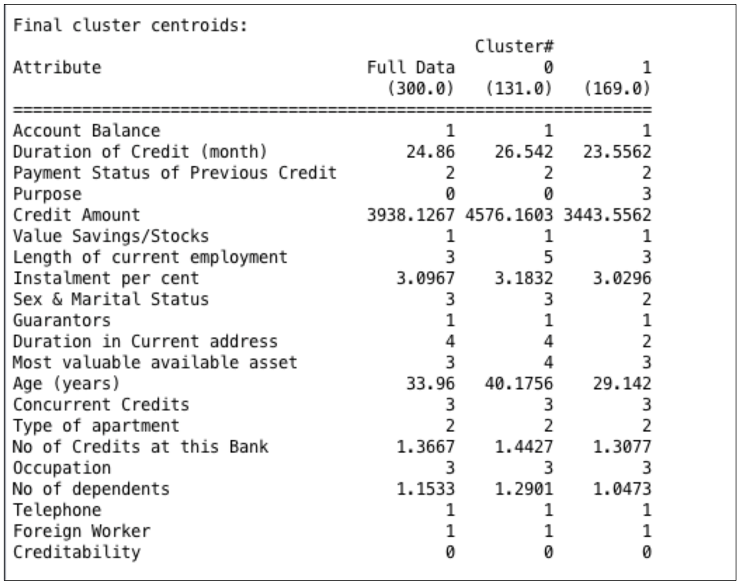
Cluster 1 consists of older population who applied for higher credit amount. People in this cluster chose a longer duration of credit. The mean of Cluster 1 tends towards 0, it could be possible that these people purchased a new car with the loan amount.

The final cluster - **Cluster 2** consisted of middle-aged population who had applied for higher credit. The people in this cluster went for longer duration credits. An important trait of this cluster is, that this cluster consisted of people who had an average of 3 valuable assets.

3.1.3. CHARACTERISTICS OF DISAPPROVED CLIENTS (300 INSTANCES)

The third and final test was conducted to determine the characteristics of bad credit customers. While analyzing the output of both clusters, we extract and infer several significant observations. Firstly, we see that both clusters who have a checking account balance under zero were considered high risk individuals and were disapproved. The bank also considered a longer duration of credit as being risky. This is evident in the difference between the individuals who received a loan (short duration of credit between 16-20 months) and those who were rejected the loan (long duration of credit >20 months). In addition, we observe that the bank was more reluctant in granting loans with higher credit amount. Those who were approved requested a mean amount of around $3,000 while those who were denied requested a larger sum of around $4,000. Lastly, individuals who had assets that are worth less were also less likely to be given a loan. These individuals either had a car or no assets at all.

Figure 21: K Means Cluster output for Bad Credit data



```
Final cluster centroids:
                                            Cluster#
Attribute                           Full Data          0          1
                                      (300.0)    (131.0)    (169.0)
====================================================================
Account Balance                             1          1          1
Duration of Credit (month)              24.86     26.542    23.5562
Payment Status of Previous Credit           2          2          2
Purpose                                     0          0          3
Credit Amount                       3938.1267  4576.1603  3443.5562
Value Savings/Stocks                        1          1          1
Length of current employment                3          5          3
Instalment per cent                    3.0967     3.1832     3.0296
Sex & Marital Status                        3          3          2
Guarantors                                  1          1          1
Duration in Current address                 4          4          2
Most valuable available asset               3          4          3
Age (years)                             33.96    40.1756     29.142
Concurrent Credits                          3          3          3
Type of apartment                           2          2          2
No of Credits at this Bank             1.3667     1.4427     1.3077
Occupation                                  3          3          3
No of dependents                       1.1533     1.2901     1.0473
Telephone                                   1          1          1
Foreign Worker                              1          1          1
Creditability                               0          0          0
```

Within each cluster, we can build a profile of each type of consumer as follows:

We can observe that the loan application of people in Cluster 0 was denied because they were old people who applied for very high credit amount. The members of this cluster had been in the same location and job for relatively long time. The mean values of attribute Purpose tends to zero i.e. they wanted to buy a new case, this could be one probable reason for denial of loan.

In contrast to Cluster 0, Cluster 1 consisted of young individuals who had relatively less job experience but wanted long duration of credit. It is possible that their loan application was declined due to the high amount they asked for despite having less experience.

3.2. APRIORI

Association rules analysis is a technique to uncover how items are associated to each other. As part of post predictive analysis, we used Apriori – an association algorithm, to determine the association between attributes provided in the German Credit Card Data set and verify our results from earlier. Apriori algorithm is applicable only to nominal, binary and unary data. All numerical attributes in the data set were converted to Nominal by using the NumericToNominal feature.

Below were the parameters that we considered when using the algorithm in Weka:

1. **Support**: Measures the popularity of an attribute in the dataset. The lowerBoundMinSupport (lower interval) and upperBoundMinSupport (upper interval) parameters can be adjusted in Weka. The Apriori algorithm works between these two intervals and increments by a delta value (increment level).

2. **Confidence**: Measures the probability of attribute X being chosen, given attribute Y. Confidence can be changed by choosing metricType as Confidence and minMetric parameter.

3.2.1. NUMERIC TO NOMIAL PARAMETER CONVERSION

Apriori Algorithm works only on Nominal Data. There are two ways to convert Numeric values to Nominal values in Weka

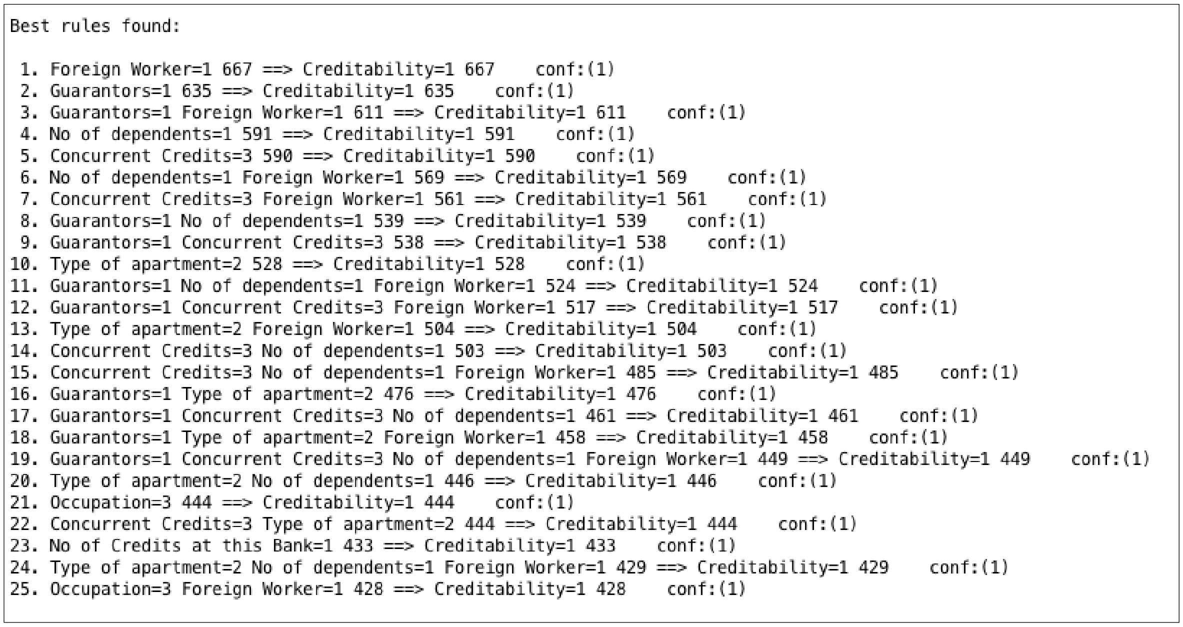
1. **Remove with values feature (RemoveWithValues):** This converts all the numeric attributes to Nominal in the dataset.
2. **Discretize:** In this method, we get to choose the attribute and the numbers of bins to be used when converting numeric data.

- Duration of Credit – 8 bins
- Credit Amount – 6 bins
- Installment percent – 4 bins
- Age – 8 bins
- Number of Credits at this Bank – 3 bins
- No of Dependents – 2 bins

3.2.2. RESULTS OF APRIORI ALGORITHM

To begin with, Apriori algorithm was then applied on observations for which credit was approved (700 instances). We set the metricType to Confidence, *car* parameter was set to True, so that associations would be made on class attribute. Support value was set to 0.6. Higher the confidence value higher will be the ranking.

Figure 22: Output of Apriori for Good Credit Data



```
Best rules found:

 1. Foreign Worker=1 667 ==> Creditability=1 667    conf:(1)
 2. Guarantors=1 635 ==> Creditability=1 635    conf:(1)
 3. Guarantors=1 Foreign Worker=1 611 ==> Creditability=1 611    conf:(1)
 4. No of dependents=1 591 ==> Creditability=1 591    conf:(1)
 5. Concurrent Credits=3 590 ==> Creditability=1 590    conf:(1)
 6. No of dependents=1 Foreign Worker=1 569 ==> Creditability=1 569    conf:(1)
 7. Concurrent Credits=3 Foreign Worker=1 561 ==> Creditability=1 561    conf:(1)
 8. Guarantors=1 No of dependents=1 539 ==> Creditability=1 539    conf:(1)
 9. Guarantors=1 Concurrent Credits=3 538 ==> Creditability=1 538    conf:(1)
10. Type of apartment=2 528 ==> Creditability=1 528    conf:(1)
11. Guarantors=1 No of dependents=1 Foreign Worker=1 524 ==> Creditability=1 524    conf:(1)
12. Guarantors=1 Concurrent Credits=3 Foreign Worker=1 517 ==> Creditability=1 517    conf:(1)
13. Type of apartment=2 Foreign Worker=1 504 ==> Creditability=1 504    conf:(1)
14. Concurrent Credits=3 No of dependents=1 503 ==> Creditability=1 503    conf:(1)
15. Concurrent Credits=3 No of dependents=1 Foreign Worker=1 485 ==> Creditability=1 485    conf:(1)
16. Guarantors=1 Type of apartment=2 476 ==> Creditability=1 476    conf:(1)
17. Guarantors=1 Concurrent Credits=3 No of dependents=1 461 ==> Creditability=1 461    conf:(1)
18. Guarantors=1 Type of apartment=2 Foreign Worker=1 458 ==> Creditability=1 458    conf:(1)
19. Guarantors=1 Concurrent Credits=3 No of dependents=1 Foreign Worker=1 449 ==> Creditability=1 449    conf:(1)
20. Type of apartment=2 No of dependents=1 446 ==> Creditability=1 446    conf:(1)
21. Occupation=3 444 ==> Creditability=1 444    conf:(1)
22. Concurrent Credits=3 Type of apartment=2 444 ==> Creditability=1 444    conf:(1)
23. No of Credits at this Bank=1 433 ==> Creditability=1 433    conf:(1)
24. Type of apartment=2 No of dependents=1 Foreign Worker=1 429 ==> Creditability=1 429    conf:(1)
25. Occupation=3 Foreign Worker=1 428 ==> Creditability=1 428    conf:(1)
```

Some of the rules and their interpretation is as follows:

1.Foreign Worker=1 667 ==> Creditability=1 667

Loan seeking population mostly constituted of Foreign works

2. Guarantors=1 635 ==> Creditability=1 635

Most of the loan applications which were approved, did not require a guarantor

3. Guarantors=1 Foreign Worker=1 611 ==> Creditability=1 611

Debtor who were foreign workers required no guarantors in 611 cases

4. No of dependents=1 591 ==> Creditability=1 591

In most cases, debtor has only 1 dependent

10. Type of apartment=2 528 ==> Creditability=1 528

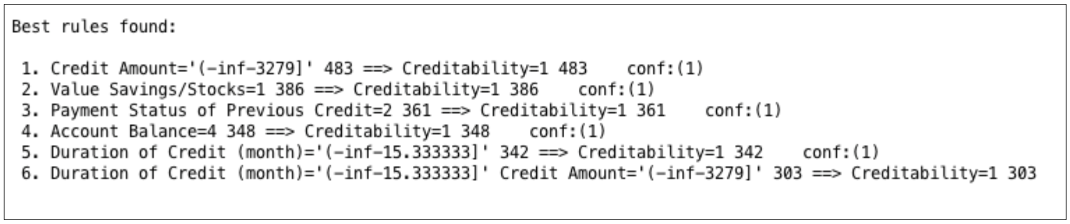
Debtor owns his/her apartment in 528 instances

21. Occupation=3 444 ==> Creditability=1 444

Skilled Employee were debtors

Next, we tested the algorithm on the top 6 attributes. Initially, we tried running this test with high values of support, but association rules were generated only when support value was 0.4 or lower.

Figure 23: Output of Apriori for Good Credit and top 6 attributes



```
Best rules found:

 1. Credit Amount='(-inf-3279]' 483 ==> Creditability=1 483    conf:(1)
 2. Value Savings/Stocks=1 386 ==> Creditability=1 386    conf:(1)
 3. Payment Status of Previous Credit=2 361 ==> Creditability=1 361    conf:(1)
 4. Account Balance=4 348 ==> Creditability=1 348    conf:(1)
 5. Duration of Credit (month)='(-inf-15.333333]' 342 ==> Creditability=1 342    conf:(1)
 6. Duration of Credit (month)='(-inf-15.333333]' Credit Amount='(-inf-3279]' 303 ==> Creditability=1 303
```

Some association rules and their interpretation were as follows:

1. Credit Amount='(-inf-3279]' 483 ==> Creditability=1 483

   Debtors seek smaller credit amount

2. Value Savings/Stocks=1 386 ==> Creditability=1 386;

   Good debtors have low saving

3. Payment Status of Previous Credit=2 361 ==> Creditability=1 361

   Good debtors paid back their old loans

Lastly, we tried to implement the algorithm on all 20 attributes all 1000 records in the dataset. For this particular test, we started with default support value of 0.5, there were no associate rules in this setting. Results were observed only when the support value was 0.3. We concluded that results obtained using this test could not be used to figure out associations as the support value was too low.

3.2.3. APRIORI CONCLUSION

We recommend the bank that creditable debtors be characterized as people:

1. Do not require guarantors
2. Have no additional loans
3. Have less dependents
4. Own their own apartments
5. Have low savings and are interested to borrow
6. Pay off loans in a timely manner

Also, the bank manager should not approve the loans of the debtors whose checking account balance is < 0 DM (Deutsche Mark) and average balance in savings and stocks is <100 DM as they would have no means to repay the loan.

3.3. RECOMMENDATIONS FOR BANK

We ran K Means algorithm after dividing the given dataset into clients whose loan application was approved and clients whose loan application was rejected. By observing patterns in various clusters, we were able to find traits of good and bad debtors.

Factors such as Age, Most valuable available asset and Purpose: were some of the common traits we observed in all the clusters, which could help decide if loan amount could be issued. For instance, we could conclude that a middle-aged person who applies for lower credit amount for a short duration of time is more likely to be issued a loan than a middle-aged person who applied for loan for high amount and for longer duration of time.

It could also be noted that **not** all top attributes which were identified as part of Feature selection contributed to effectively predict loan eligibility. Factors like Value Savings/Stocks had a common value of 1 in both in good credit clusters and bad credit clusters.

Next, we used Apriori algorithm, to mine association rules by only considering the credit worthy population and were able to come up with recommendation - Creditable borrowers don’t need guarantors, have no other loans/credits with no existing instalment plans, have fewer dependents, own their own apartments, have low savings, pay off loans on time, and want smaller loans. Based on these recommendations, prospective borrowers not profiled as creditable should have markers that deviate from the profile above.

4. REFERENCES

Brownlee, Jason. “How to Perform Feature Selection With Machine Learning Data in Weka.” *Machine Learning Mastery*, 13 Dec. 2019, machinelearningmastery.com/perform-feature-selection-machine-learning-data-weka/.

Brownlee, Jason. “How to Transform Your Machine Learning Data in Weka.” *Machine Learning Mastery*, 13 Dec. 2019, machinelearningmastery.com/transform-machine-learning-data-weka/.

Chakrabarty, Navoneel. “Application of Synthetic Minority Over-Sampling Technique (SMOTe) for Imbalanced Datasets.” *Medium*, Towards AI - Best Artificial Intelligence Publication, 16 July 2019, medium.com/towards-artificial-intelligence/application-of-synthetic-minority-over-sampling-technique-smote-for-imbalanced-data-sets-509ab55cfdaf.

Chakure, Afroz. “Decision Tree Classification.” *Medium*, Towards Data Science, 7 July 2019, towardsdatascience.com/decision-tree-classification-de64fc4d5aac.

Ciortan, Madalina. “Overview of Feature Selection Methods.” *Medium*, Towards Data Science, 26 July 2019, towardsdatascience.com/overview-of-feature-selection-methods-a2d115c7a8f7.

Dabbura, Imad. “K-Means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks.” *Medium*, Towards Data Science, 3 Sept. 2019, towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a.

“Decision Tree Learning.” *Wikipedia*, Wikimedia Foundation, en.wikipedia.org/wiki/Decision_tree_learning.

“Feature Selection.” *Wikipedia*, Wikimedia Foundation, en.wikipedia.org/wiki/Feature_selection.

Gajawada, Sampath Kumar. “Chi-Square Test for Feature Selection in Machine Learning.” *Medium*, Towards Data Science, 20 Oct. 2019, towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223.

Janabi, Dr. Kadhim B. S. Al. “Data Reduction Techniques: A Comparative Study for Attribute Selection Methods.” (2018).

Joshi, Renuka. “Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures.” *Exsilio Blog*, 9 Sept. 2016, blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/.

Lutins, Evan. “Ensemble Methods in Machine Learning: What Are They and Why Use Them?” *Medium*, Towards Data Science, 2 Aug. 2017, towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f.

Ng, Annalyn. “Association Rules and the Apriori Algorithm: A Tutorial.” *KDnuggets*, Apr. 2016, www.kdnuggets.com/2016/04/association-rules-apriori-algorithm-tutorial.html.

Paul, Sayak. “Diving Deep with Imbalanced Data.” *DataCamp Community*, 4 Oct. 2018, www.datacamp.com/community/tutorials/diving-deep-imbalanced-data.

Shubham, Jinde. “Ensemble Learning - Bagging and Boosting.” *Medium*, Becoming Human: Artificial Intelligence Magazine, 6 July 2018, becominghuman.ai/ensemble-learning-bagging-and-boosting-d20f38be9b1e.

Smolyakov, Vadim. “Ensemble Learning to Improve Machine Learning Results.” *Medium*, Stats and Bots, 7 Mar. 2019, blog.statsbot.co/ensemble-learning-d1dcd548e936.