Final Project
Credit Risk Analysis
Leveraging Classification Algorithms for Informed Loan Approval Decisions

---

Course: CIND 119 - Introduction to Big Data
Toronto Metropolitan University
Instructor: Zekiye Erdem

Project Members

Rozani Jeganathan
Eden Wondimu
Gbemisola Ayejuni

Table of Contents

1. Abstract

This project aims to develop a robust credit risk assessment strategy for bank managers using the German Credit Dataset, comprising 1000 entries and 21 columns. The primary objective is to employ classification algorithms to recommend effective loan approval strategies. The dataset features a mix of qualitative and quantitative attributes, including socioeconomic factors and credit history indicators. The target attribute, "Creditability," denotes creditworthiness and will be utilized for classification purposes.

The initial methodology involves a simple train-test set split, allocating 70% of the dataset for training and 30% for testing. Prior to splitting, data randomization is performed to prevent bias. The target variable, "Creditability" denotes creditworthiness, where a value of 1 indicates good credit and 0 indicates bad credit. Two classification algorithms, Decision Tree and Naive Bayes, are applied.

Results indicate that Decision Tree with Label Encoding yields the highest accuracy in credit risk prediction, with Naive Bayes also performing reasonably well. The implications of these findings underscore the importance of utilizing advanced classification techniques for improved loan approval decisions.

In the advanced analysis, feature selection methods including the Correlation Matrix, BestFirst algorithm, and Chi-Square Test were employed to identify the most informative attributes for credit risk assessment. These techniques were utilized to filter the dataset and prioritize features based on their relevance to loan approval decisions.

In conclusion, this project provides valuable insights into credit risk assessment strategies, demonstrating the potential of machine learning algorithms in enhancing lending practices.

Keywords: Credit risk assessment, Classification algorithms, German Credit Dataset, Loan approval, Machine learning.

Tools: Python, Weka, Sklearn and SAS for data analysis
Data visualization Library: Matplotlib, Seaborn, Tabulate, Graphviz
Tableau for data visualization

2. Workload Distribution

Member Name	List of Tasks
Eden Wondimu	○ Dataset Summary ○ Data Preparation - Attribute analysis and Summary Statistics - Outlier Detection - Distribution Analysis - Correlation Analysis and Attribute Elimination
Rozani Jeganathan	○ Abstract ○ Predictive Modeling (Classification) - Decision Tree ○ Performance Metrics Comparison and Conclusion
Gbemisola Ayejuni	○ Predictive Modeling (Classification) - Naive Bayes ○ Comparison of Baseline and Selected Features in Predictive Modeling ○ Post-Predictive Analysis ○ Conclusion and Recommendation

3. Dataset Summary

The German Credit Card dataset has 1000 entries and 21 features - 20 attributes and the class attribute ‘Creditability’. Of these features - 15 of them, including the class attribute, are qualitative while 6 of the features have quantitative values. The dataset was first loaded into a suitable data structure, in this case a pandas DataFrame in Python, and explored to understand its structure, including the number of rows and columns, column (attribute) names, and the types of data it contains. The dataset consists of both qualitative (nominal and ordinal) and quantitative attributes, with the target attribute "Creditability" indicating good or bad credit ratings of customers.

Our team will analyze the data and use classification algorithms in order to recommend a strategy with which bank managers can decide whether to grant loans to prospective applicants.

4. Data Preparation

4.1 Data Integrity Checks

We first checked for inconsistencies such as missing values (Figure 1) and duplicated entries (rows) using the df.info() and df.duplicated() functions, respectively, to help identify whether any data cleaning or handling is required prior to data analysis. The dataset had no missing values and did not have any duplicated entries, indicating it is clean and ready for further processing.

4.2 Attribute Analysis and Summary Statistics

All 15 categorical attributes are pre-encoded numerically, eliminating the need for additional one-hot encoding steps. We can gain more insight into the dataset by summarizing measures like mean, median, standard deviation, min, and max values, particularly for the dataset's numeric attributes. Using the pandas df.describe() function, returns the key summary statistics of each numerical attribute in the dataset and helps us better understand central tendency, dispersion, and range of values. Using Tableau, the attribute data types, and summary statistics were summarized in Table 1 below.

Table 1: Attribute data types and summary statistics

Data Type	Attributes	Mean	Std	Min	25%	50%	75%	Max	Missing Values
Quantitative	Age (years)	36	11	19	27	33	42	75	0
	Credit Amount	3,271	2,823	250	1,366	2,320	3,972	18,424	0
	Duration of Credit (month)	21	12	4	12	18	24	72	0
	Instalment per cent	3	1	1	2	3	4	4	0
	No of Credits at this Bank	1	1	1	1	1	2	4	0
	No of dependents	1	0	1	1	1	1	2	0
Nominal	Concurrent Credits								0
	Creditability								0
	Foreign Worker								0
	Guarantors								0
	Most valuable available asset								0
	Occupation								0
	Purpose								0
	Sex & Marital Status								0
	Telephone								0
	Type of apartment								0
Ordinal	Account Balance								0
	Duration in Current address								0
	Length of current employment								0
	Payment Status of Previous Credit								0
	Value Savings/Stocks								0

4.3 Outlier Detection

The summary statistics of numerical attributes in Table 1 above is also useful in identifying potential outliers by comparing the max and min values with the quartile ranges. For example, credit amounts vary from very small to significantly large loans. Although 50% of all credits being less than 2319.5 Deutsche Mark (DM), there are credits that are much higher (max = 18424). Therefore, we can surmise from the presence of high values and significant spread that the dataset might contain outliers or at the very least a heavy tail on the higher end. Such outliers may require handling or removal if found to have an impact on the performance of models during data analysis steps.

Using Python, outliers were detected in several numeric attributes, including ‘Duration of Credit (month)’ and ‘Credit Amount’. Box plots of the 3 continuous-qualitative attributes are shown in Figure 1 below.

Figure 1: Boxplots of Numeric Attributes

To better visualize and assess outliers and the distribution of Duration of credit, Credit amount and Age between groups with good and bad credit ratings, we used Tableau. Since the original dataset had already undergone one-hot encoding, we first used functions in Tableau to create a calculated field for Creditability to transform indicators 1 and 0 into the corresponding categorical values "Good" and "Bad", respectively. All 3 continuous quantitative attributes were then plotted against our target class variable (Figure 2).

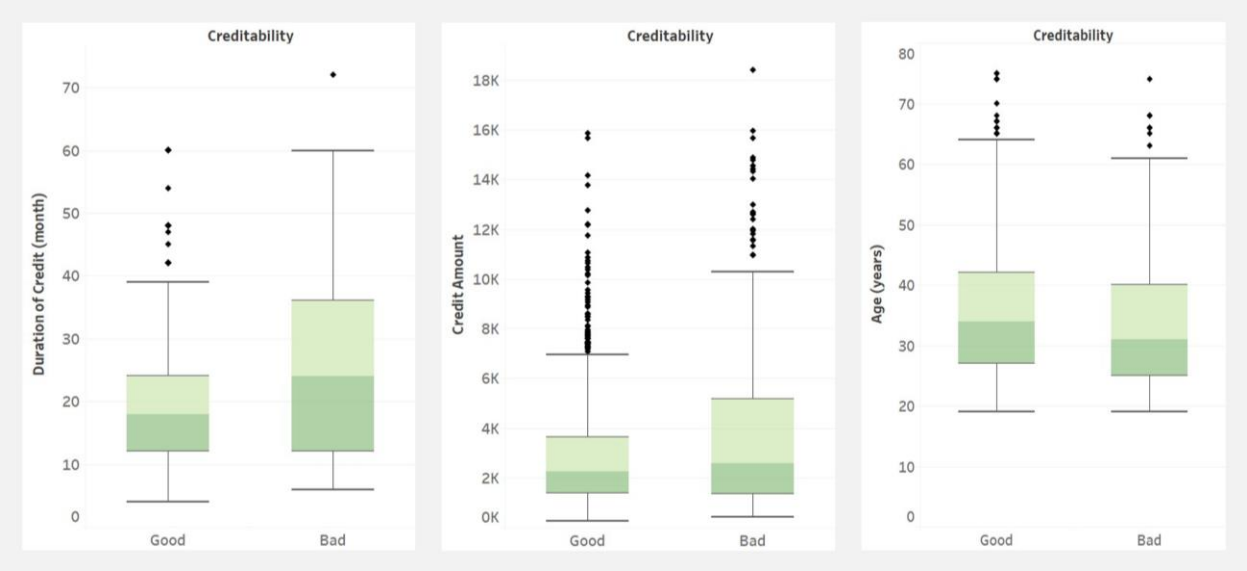


Figure 2: Distribution of Credit duration, Credit Amount, and Age by Credit Rating categories

Figure 2 reveals that (1) customers with bad credit ratings tend to have a slightly higher median duration of credit compared to those with good credit ratings, and that (2) there are customers with remarkably long credit durations in both good and bad credit groups. This shows that although credit duration on its own may not be a definitive predictor of credit quality, as there is some overlap in the interquartile ranges, there appears to be a pattern suggesting that customers with bad credit ratings tend to have marginally longer credit periods.

The median credit amount for customers with bad credit is slightly higher than for those with good credit. Similar to duration of credit, there are many outliers on the higher end for both good and bad credit ratings, indicating that some customers have taken significantly larger loans than the typical customer. Overall, the amount of credit taken on by customers with bad credit ratings trends higher, suggesting that larger loan amounts might be associated with increased risk of bad credit ratings.

Unlike the other two predictors, the age distributions for both good and bad credit ratings are similar, with nearly overlapping interquartile ranges and medians and in both good and bad

credit groups, there are some older customers as indicated by the outliers. Figure 2 suggests that age may not be a strong differentiator between good and bad credit ratings in this dataset.

4.4 Distribution Analysis

Examining the distribution of numeric attributes, particularly those that are continuous, through histograms (Figure 3) will give us important insights on spread and skewness, which are crucial to determining data transformation needs and in understanding underlying patterns.

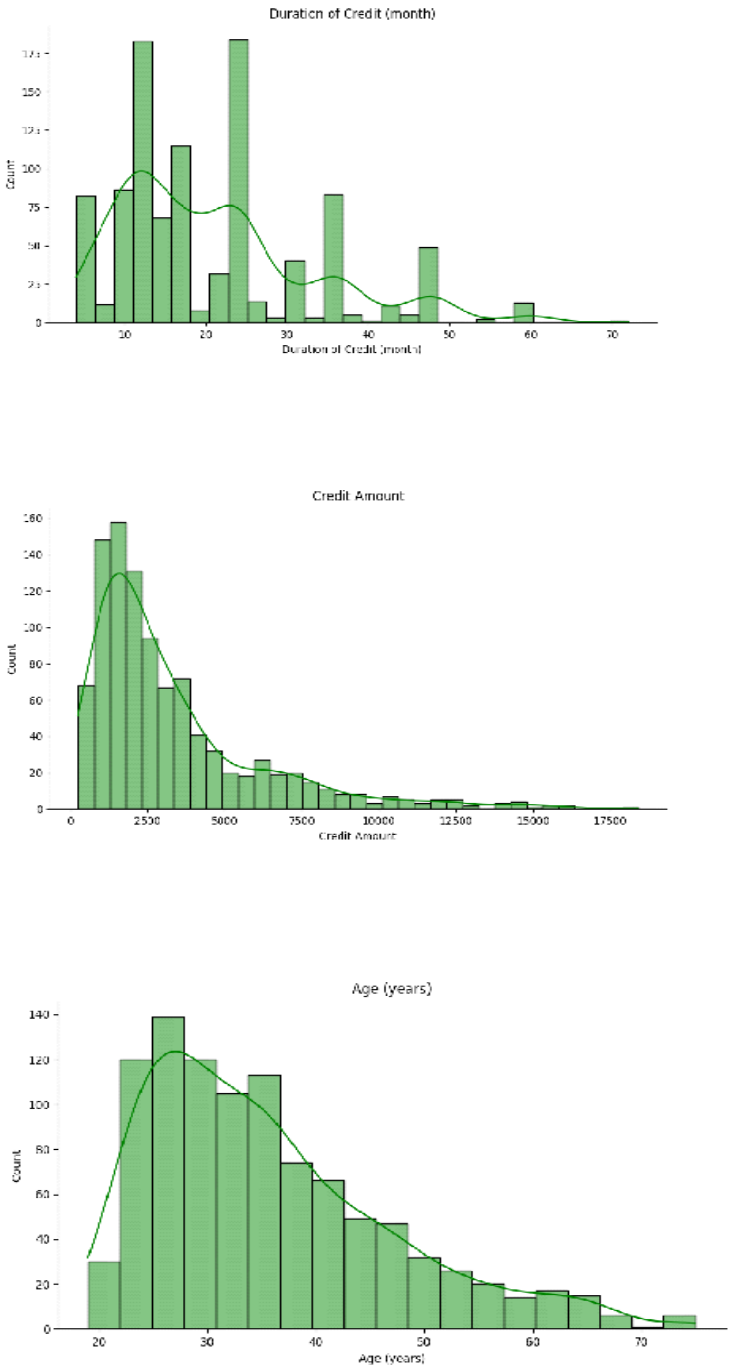


Figure 3: Histograms Depicting the Distribution of Credit Duration, Credit Amount, and Age

The Duration of Credit and Credit Amount histograms above, coupled with the kernel density estimation (KDE) overlay (Figure 3), showed right-skewed distributions, indicating that most loans/credits are of shorter duration and lower amounts, respectively. The histogram for the Duration of Credit shows a multi-modal distribution with prominent peaks, suggesting that certain loan durations are more common than others – the bars where the number of loans is concentrated signify that these may be preferred term lengths. It's also important to note that the

long tail to the right indicates that while most loans have shorter durations, a smaller number have significantly longer durations.

The distribution of Credit Amount shows that most loans are of smaller amounts while fewer loans are of higher amounts. The steep drop-off in frequency as the credit amounts increase suggests that higher loan amounts are less common. The distribution of Age is right-skewed as well, indicating that a larger proportion of borrowers are younger.

Imbalanced Class Distribution Check

To check if the target classes are balanced in size, we plotted a bar chart for the Creditability attribute (Figure 4).

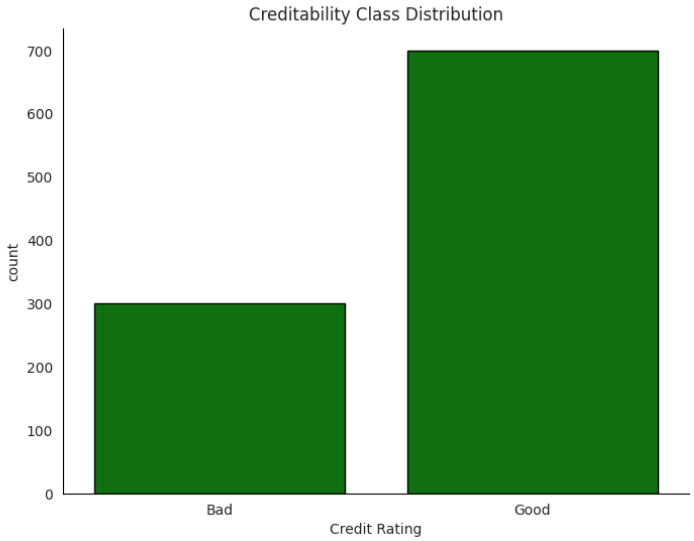


Figure 4: Creditability Class Distribution

The target variable "Creditability" is somewhat imbalanced, with a higher proportion of good credit ratings (70%) compared to bad (30%). This imbalance will need to be addressed in succeeding algorithmic and modelling steps to ensure model fairness. Imbalanced classes can bias the models towards the majority class, impacting model performance.

4.5 Attribute Correlation and Selection

Correlation analysis determines the strength and direction of the relationship between variables. This helps in understanding which attributes are most related to the target variable and each other. The correlation between all features and the target attribute can be seen in the correlation matrix heatmap below (Figure 5).

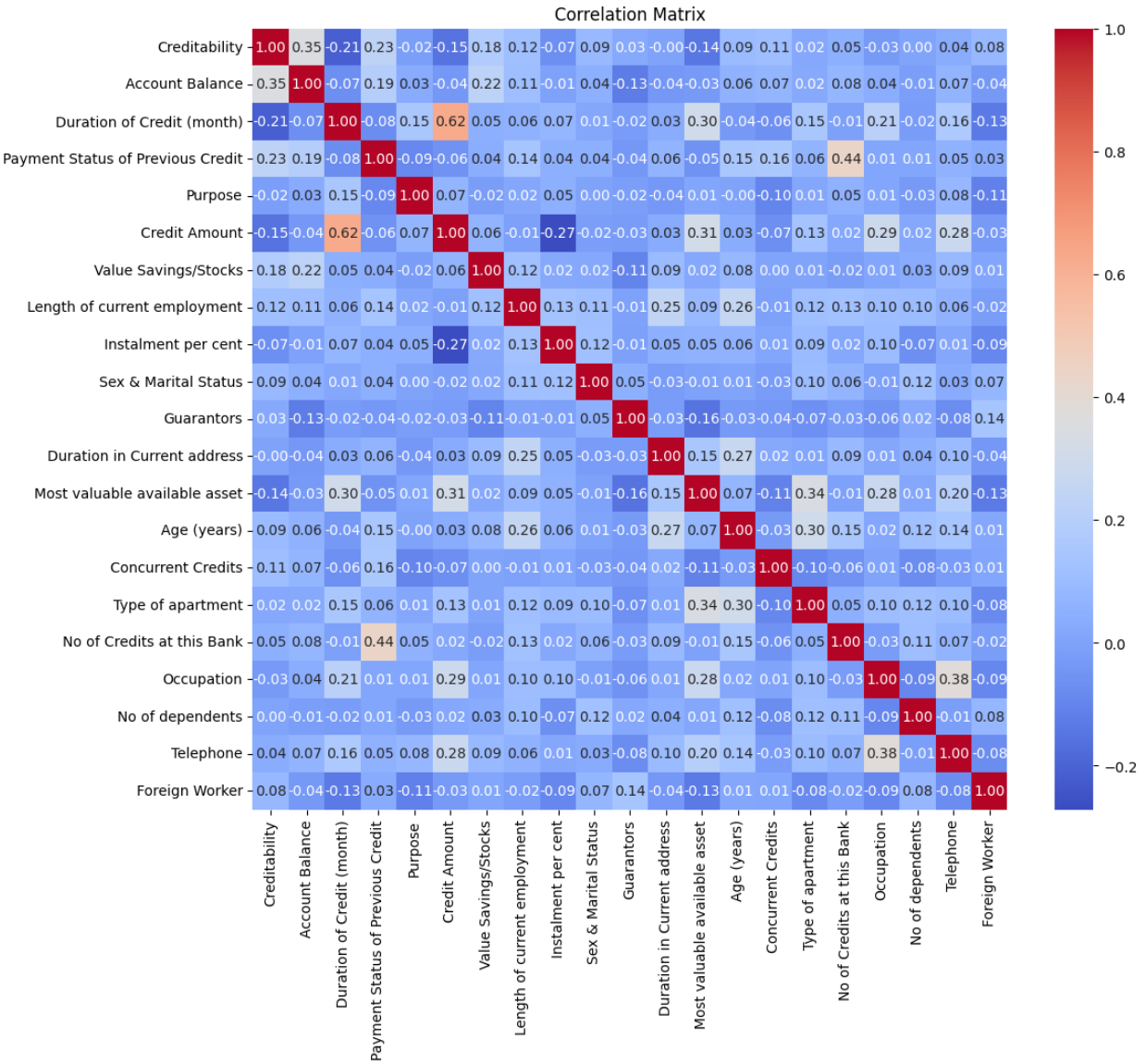


Figure 5: Correlation Matrix Heatmap

Most attributes showed low to moderate correlation with the target variable Creditability. Below, the DataFrame generated from the correlation analysis, with attributes listed in order of descending correlation values, better summarizes the heatmap (Table 2).

Table 2: Correlation of all attributes with Creditability

	Attribute	Correlation with Creditability
0	Creditability	1.000000
1	Account Balance	0.350847
2	Payment Status of Previous Credit	0.228785
3	Value Savings/Stocks	0.178943
4	Length of current employment	0.116002
5	Concurrent Credits	0.109844
6	Age (years)	0.091272
7	Sex & Marital Status	0.088184
8	Foreign Worker	0.082079
9	No of Credits at this Bank	0.045732
10	Telephone	0.036466
11	Guarantors	0.025137
12	Type of apartment	0.018119
13	No of dependents	0.003015
14	Duration in Current address	-0.002967
15	Purpose	-0.017979
16	Occupation	-0.032735
17	Instalment per cent	-0.072404
18	Most valuable available asset	-0.142612
19	Credit Amount	-0.154740
20	Duration of Credit (month)	-0.214927

The attributes with the most positive correlation with Creditability include Account Balance, Payment Status of Previous Credit, Value Savings/Stocks, Length of current employment and Concurrent Credits.

The attributes with the weakest correlation with Creditability include:

- Duration in Current address
- Telephone, and
- Type of apartment

Based on the correlation matrix heatmap and our understanding of the dataset, we can surmise that Duration in Current address, Telephone and Type of apartment have less significance to the target attribute and are therefore best suited for elimination. No of dependents, Sex & Marital Status, and Foreign worker also have low correlation with Creditability, however, these attributes might hold some relevance and operational significance, and provide information about

demographic factors that might influence the bank's credit risk assessment processes. Therefore, Duration in Current address, Telephone and Type of apartment were eliminated and a filtered dataset without these features was generated.

4.6 Summary of Determinations about the Dataset:

- The dataset is comprehensive, with 1,000 entries and 21 attributes (both qualitative and quantitative) and has been verified for data integrity with no missing or duplicated entries, ensuring a solid foundation for analysis.
- Outlier detection in key financial attributes like Duration of Credit and Credit Amount suggests a heavy-tailed distribution, necessitating further examination or potential outlier treatment.
- Creditability analysis indicates a modest imbalance, with a greater proportion of 'Good' credit ratings, which must be addressed in modeling to avoid bias.
- Numerical attributes show right-skewed distributions, suggesting a concentration of loans with shorter durations and smaller amounts, as well as a predominance of younger borrowers.
- The distribution patterns indicate that while age is not a definitive differentiator, credit amount and duration show a trend where higher values are slightly more associated with 'Bad' credit ratings.
- When looking at correlations, attributes like Account Balance and Payment Status of Previous Credit show a connection to Creditability. On the other hand, factors like Duration in Current Address, Telephone and Type of Apartment have little correlation and were removed from the model for simplicity.
- Low to moderate correlations with the target variable suggest that no single attribute dominates the prediction of creditworthiness, highlighting the potential need for a multifaceted approach in modeling.

5. Predictive Modeling (Classification)

5.1 Data split strategy

The data split Strategy we used for this project is Simple train-test set split. Empirical studies have demonstrated that optimal results are achieved when utilizing 20-30% of the dataset for testing purposes, while allocating the remaining 70-80% for training. Therefore, we divided 70% data for training and 30% for testing. We train the model using the training set and then apply the model to the test set. In this way, we can evaluate the performance of our model. We shuffled the data before splitting, thus preventing any bias that could have arisen from the order of the data.

The train_test_split function from scikit-learn performs random shuffling of the data before splitting it into training and test sets.

Creditability is designated as the target variable, while all other columns in the dataset are considered as features. Good credit is represented by 1 and bad credit rating is represented by 0.

Applying Classification Algorithm

Two classification models, Decision Tree, utilizing two encoding techniques: Label Encoding and One-hot Encoding and Naive Bayes, were employed.

5.2 Decision Tree

CART (Classification and Regression Trees) algorithm that can be used for both classification and regression problems in machine learning. This builds a binary tree by recursively splitting the dataset based on feature thresholds to minimize impurity. The max depth parameter controls the maximum depth of the tree to prevent the tree from becoming overly complex and helps prevent overfitting. We used max depth of the tree to 5 for our analysis.

We used a “Label encoding” algorithm that can handle both categorical and numerical input features. Label Encoding carries a drawback where higher numerical labels might erroneously imply a higher significance or weight. We used an alternative method “one hot encoding” to address this issue. Below is the comparison of both methods.

	precision	recall	f1-score	support
0	0.78	0.32	0.45	91
1	0.76	0.96	0.85	209
accuracy			0.77	300
macro avg	0.77	0.64	0.65	300
weighted avg	0.77	0.77	0.73	300

	precision	recall	f1-score	support
0	0.783784	0.318681	0.453125	91.000000
1	0.764259	0.961722	0.851695	209.000000
accuracy	0.766667	0.766667	0.766667	0.766667
macro avg	0.774021	0.640202	0.652410	300.000000
weighted avg	0.770181	0.766667	0.730795	300.000000

Figure 5.2.1. Performance Results of Label Encoding

	precision	recall	f1-score	support
0	0.48	0.49	0.49	95
1	0.76	0.75	0.76	205
accuracy			0.67	300
macro avg	0.62	0.62	0.62	300
weighted avg	0.67	0.67	0.67	300

	precision	recall	f1-score	support
0	0.479592	0.494737	0.487047	95.00
1	0.762376	0.751220	0.756757	205.00
accuracy	0.670000	0.670000	0.670000	0.67
macro avg	0.620984	0.622978	0.621902	300.00
weighted avg	0.672828	0.670000	0.671349	300.00

Figure 5.2.2. Performance Results of One-hot Encoding

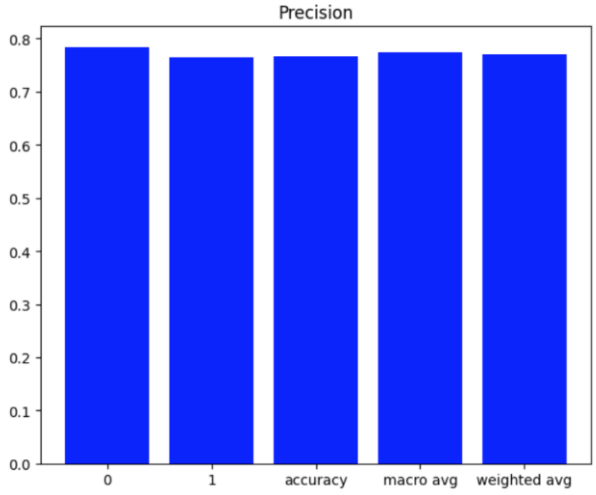


Figure 5.2.3. Precision of Label Encoding method

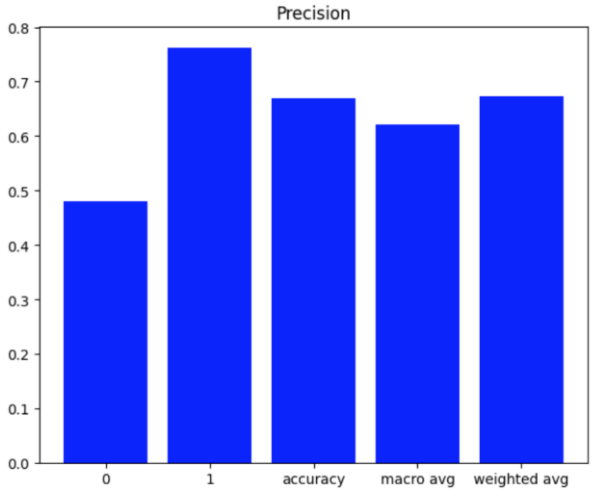


Figure 5.2.4. Precision of One-hot method

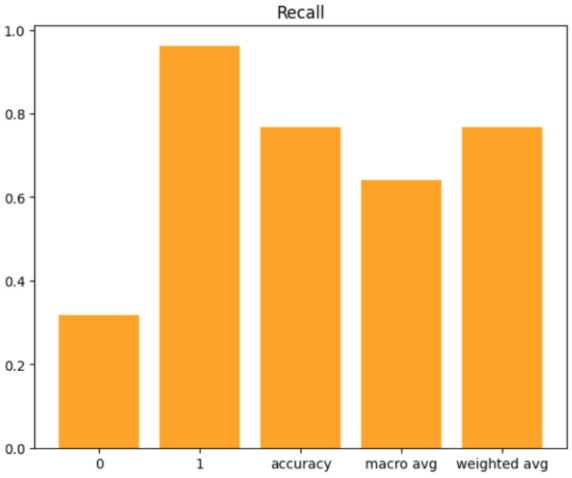


Figure 5.2.5. Recall of Label Encoding method

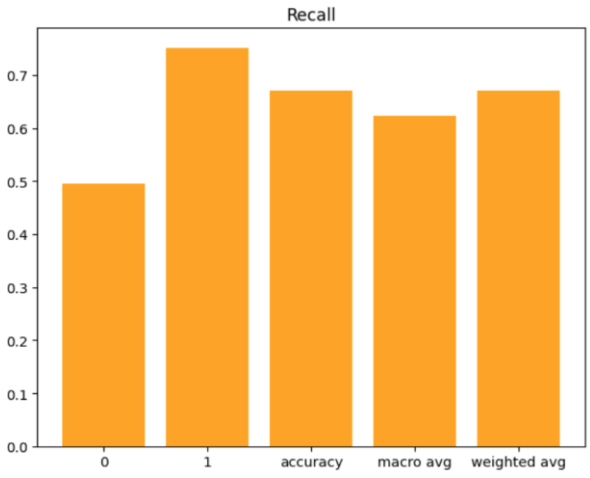


Figure 5.2.6. Recall of One-hot method

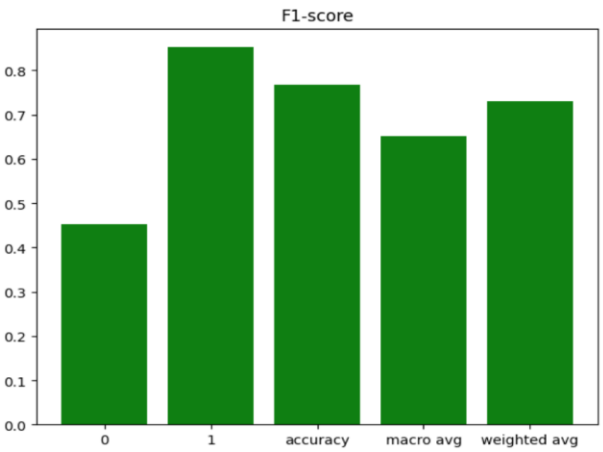


Figure 5.2.7. F1-Score of Label Encoding method

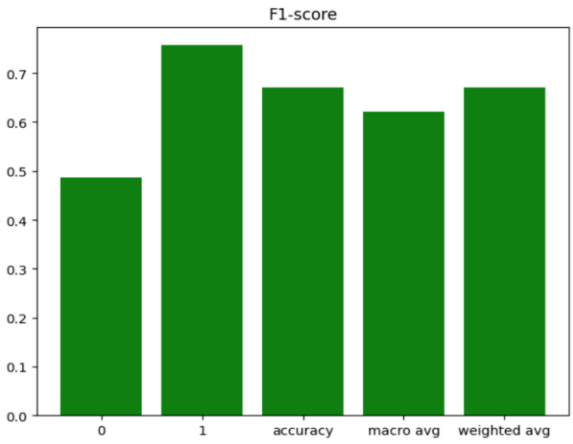


Figure 5.2.8. F1-Score of One-hot method

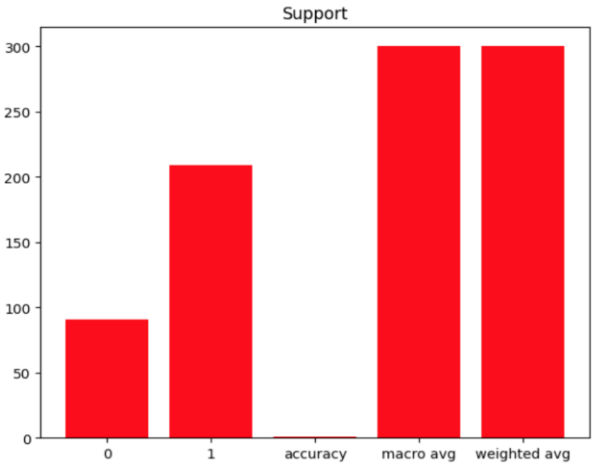


Figure 5.2.9. Support of Label Encoding method

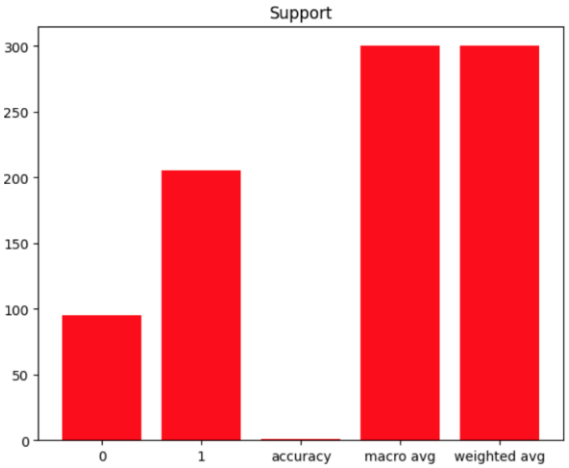


Figure 5.2.10. Support of One-hot method

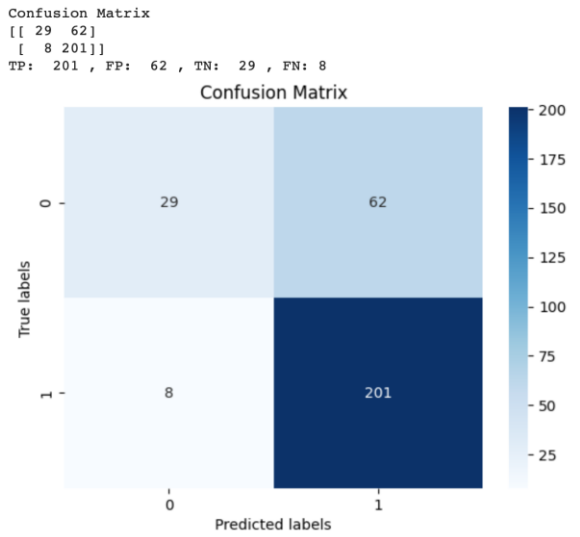


Figure 5.2.11. Confusion Matrix of Label Encoding

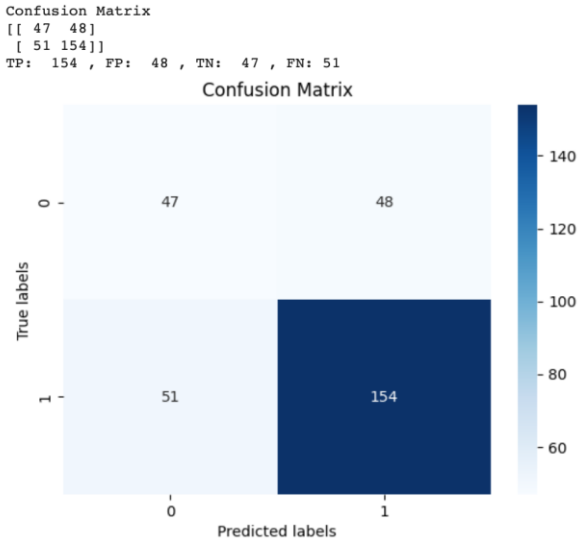


Figure 5.2.12. Confusion Matrix of One-hot Encoding

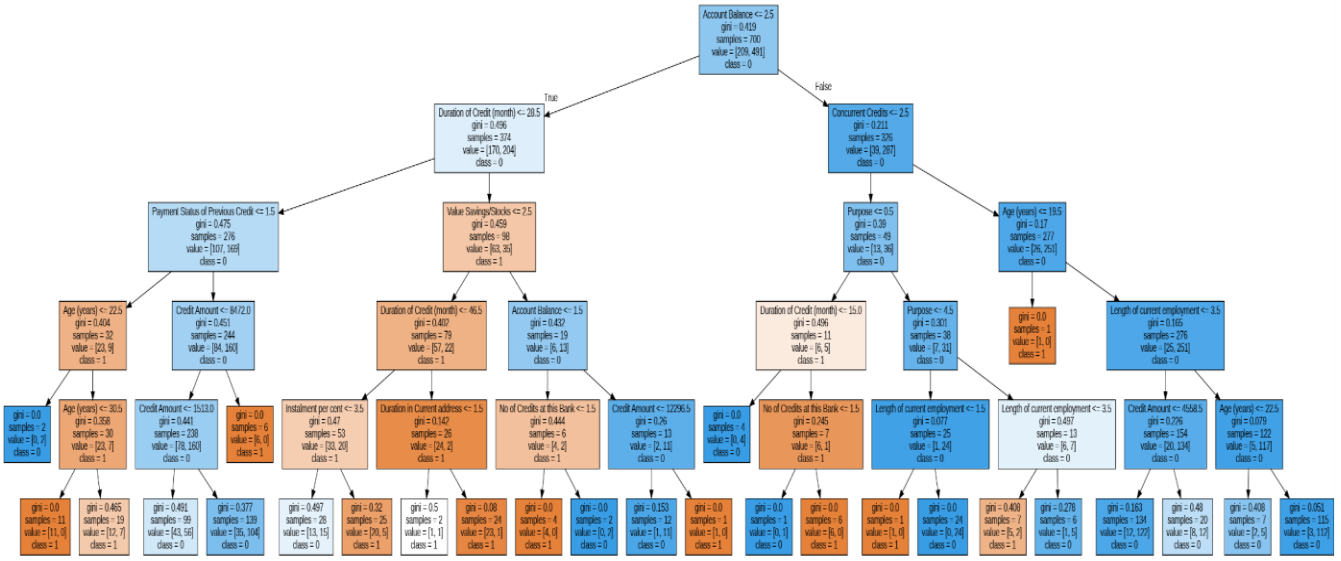


Figure 5.2.13. Decision Tree of Label Encoding

Root node: Account balance

Close to Root node: Duration of Credit (month), Credit amount

Frequently used for splitting: Credit amount, Purpose, Duration of Credit (month)

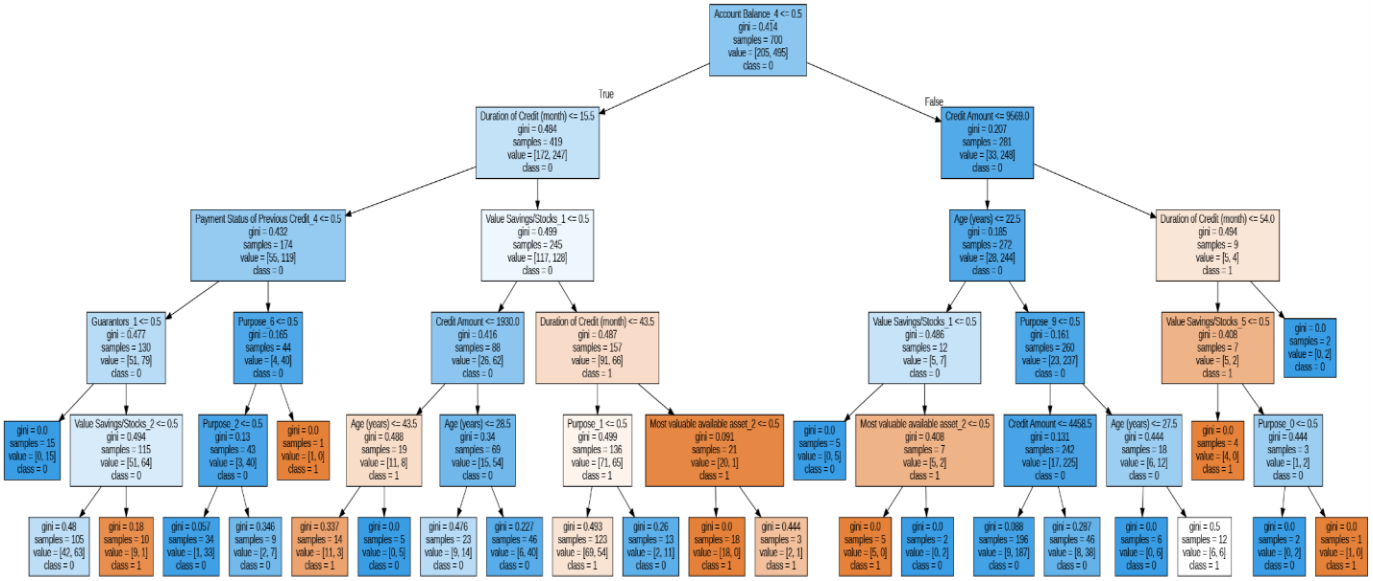


Figure 5.2.14. Decision Tree of One-hot Encoding

Root node: Account balance

Close to Root node: Duration of Credit (month), Credit amount

Frequently used for splitting: Credit amount, Purpose, Duration of Credit (month)

The decision tree model, trained with both label encoding and one-hot encoding, highlights “Account Balance” as the root node, suggesting its importance in the classification process. Also, the Gini index of 0.4 suggests that there is some degree of mixing of classes within the root node. Additionally, attributes closely linked to the root node, such as “Duration of Credit (month)” and “Credit Amount”", hold significance in decision-making. Furthermore, “Credit Amount”, “Purpose”, and “Duration of Credit (month)” emerge as frequently utilized attributes for further data partitioning. These variables collectively contribute to predicting the target variable, Creditability.

5.2.1 Best attribute selected using the BestFIRst algorithm in Weka

- Attribute 1 (Index 2): Account Balance
- Attribute 2 (Index 3): Duration of Credit (month)
- Attribute 3 (Index 4): Payment Status of Previous Credit

```
# attributes: 3
attributes: [1 2 3 0]
result string:


=== Attribute Selection on all input data ===

Search Method:
        Best first.
        Start set: no attributes
        Search direction: forward
        Stale search after 5 node expansions
        Total number of subsets evaluated: 133
        Merit of best subset found:    0.076

Attribute Subset Evaluator (supervised, Class (nominal): 1 Creditability):
        CFS Subset Evaluator
        Including locally predictive attributes

Selected attributes: 2,3,4 : 3
                     Account Balance
                     Duration of Credit (month)
                     Payment Status of Previous Credit
```

Figure 5.2.15 BestFIRst algorithm in Weka

The BestFirst algorithm in Weka, a popular data mining tool, determined the most influential attributes for constructing a decision tree model. After thorough analysis, it identified three attributes as pivotal for predicting the target variable.

Firstly, "Account Balance" emerged as the primary attribute, indicating its significant impact on the decision making process. Following closely, "Duration of Credit (month)" was recognized as another crucial factor influencing the outcome. Lastly, "Payment Status of Previous Credit" was highlighted as an essential predictor, contributing substantially to the decision tree's accuracy.

5.2.2 Best attribute selection using SAS

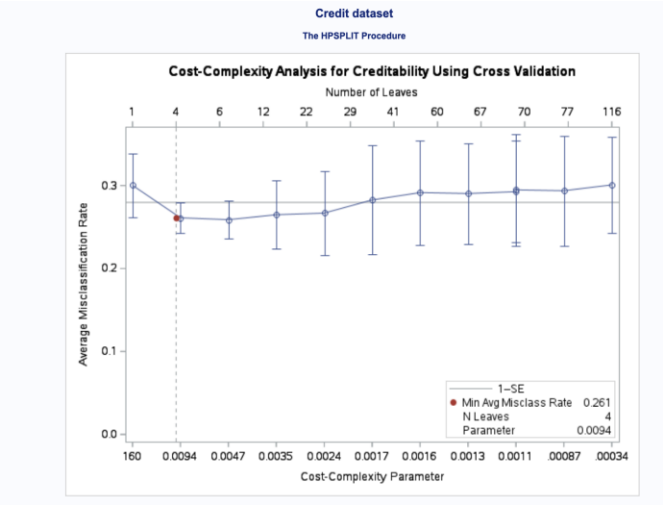


Figure 4.2.16 Cost-Complexity Analysis

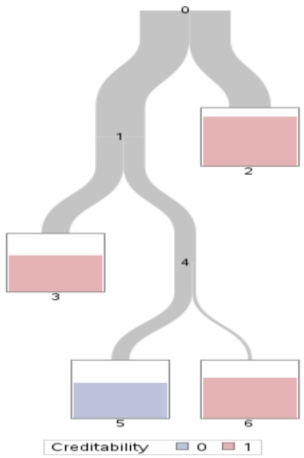


Figure 4.2.17 Decision Tree SAS

Credit dataset

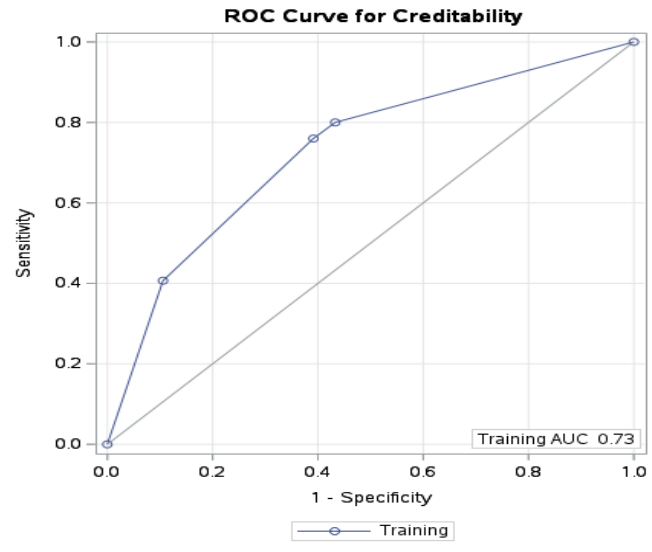
The HPSPLIT Procedure

Model-Based Confusion Matrix

Actual	Predicted 0	Predicted 1	Error Rate
0	122	178	0.5933
1	74	626	0.1057

Model-Based Fit Statistics for Selected Tree

N Leaves	ASE	Mis-class	Sensitivity	Specificity	Entropy	Gini	RSS	AUC
4	0.1760	0.2520	0.4067	0.8943	0.7644	0.3519	351.9	0.7309



Variable Importance

Variable	Training Relative	Training Importance	Count
Account Balance	1.0000	6.9217	1
Duration of Credit (month)	0.5171	3.5792	1
Value Savings/Stocks	0.3923	2.7156	1

Figure 4.2.18 Credit data report by SAS

The decision tree model was trained using entropy as the split criterion and cost-complexity as the pruning method, with a maximum tree depth requested and achieved at 10. After pruning, the tree's complexity was significantly reduced from 141 to 4 leaves, simplifying its structure.

5.2.3 Confusion Matrix

Actual	Predicted	Error Rate
0	0	122
1	0	74

- True Positives (TP): 626 instances of class 1 were correctly predicted as class 1.
- False Positives (FP): 178 instances of class 0 were incorrectly predicted as class 1.
- True Negatives (TN): 122 instances of class 0 were correctly predicted as class 0.
- False Negatives (FN): 74 instances of class 1 were incorrectly predicted as class 0.

The AUC value of 0.7309 indicates that the ROC curve plots true positive rate against the false positive rate, and the model's ability to distinguish between positive and negative classes is fairly good. An AUC value closer to 1 indicates better performance. Therefore, the model seems to have reasonable discrimination ability.

The variable "Payment Status of Previous Credit" was excluded from the analysis because it contains more than 32 characters, which is a limitation in the analysis tool (SAS).

The important variables identified from the analysis are: Account Balance
Duration of Credit(month)
Value Savings/Stocks

- "Account Balance" is considered the most important variable, with a relative count of 1.0000 and an importance score of 6.9217.
- "Duration of Credit (month)" follows, with a relative count of 0.5171 and an importance score of 3.5792.
- "Value Savings/Stocks" is the third most important variable, with a relative count of 0.3923 and an importance score of 2.7156.

These variables are considered significant for the analysis based on their relevance and impact on the outcome of the model being developed by SAS.

The decision tree analysis was conducted on modified data with a focus on six key variables: 'Account Balance', 'Duration of Credit (month)', 'Payment Status of Previous Credit', 'Purpose', 'Credit Amount', and 'Value Savings/Stocks' for post predictive analysis.

	precision	recall	f1-score	support
0	0.59	0.18	0.27	95
1	0.71	0.94	0.81	205
accuracy			0.70	300
macro avg	0.65	0.56	0.54	300
weighted avg	0.67	0.70	0.64	300

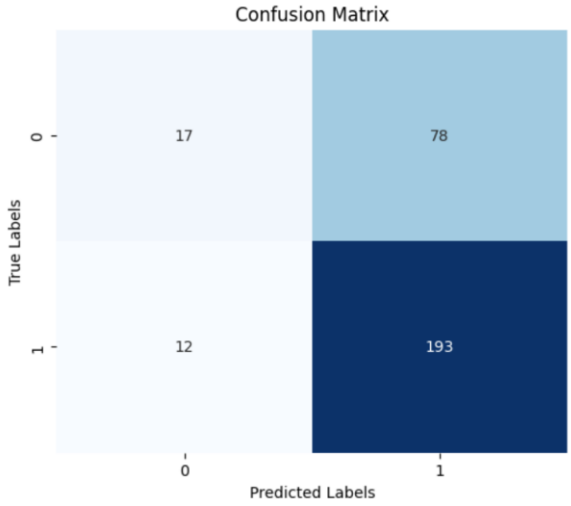


Figure 5.2.19. Confusion Matrix of Label Encoding(modified data)

	precision	recall	f1-score	support
0	0.59	0.18	0.27	95
1	0.71	0.94	0.81	205
accuracy			0.70	300
macro avg	0.65	0.56	0.54	300
weighted avg	0.67	0.70	0.64	300

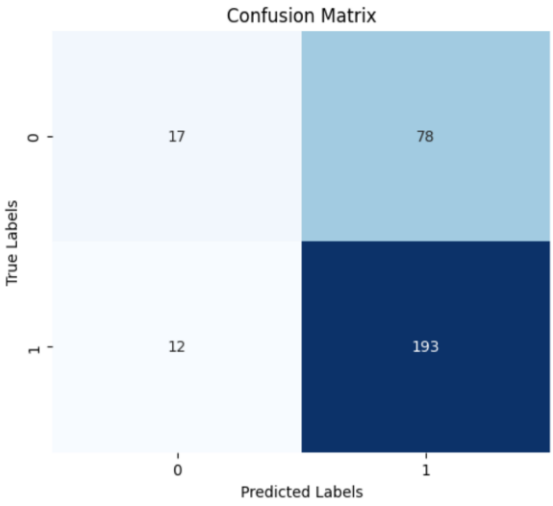


Figure 5.2.20. Confusion Matrix of One-hot Encoding(modified data)

6. Classification Using Naive Bayes

Naive Bayes is a probabilistic machine learning model based on Bayes' theorem. it assumes independence between features and calculates the probability of a given input belonging to a particular class.

With the Naive Bayes classifier, 2 fundamental assumptions are made: Each feature makes an independent and equal contribution to the outcome.

We first run the naive bayes classifier on the original German Credit card dataset. We run the algorithm using both the weka package and the sklearn package in python. The Naïve Bayes algorithm was used over the 20 attributes and 1 class attribute, Creditability. Using the same train-test split strategy as with the decision tree, the results for the Naïve Bayes classification is as follows:

6.1 Naive Bayes using Weka

```
Correctly Classified Instances         225               75      %
Incorrectly Classified Instances        75               25      %
Kappa statistic                          0.4059
Mean absolute error                      0.298
Root mean squared error                  0.4233
Relative absolute error                 68.5394 %
Root relative squared error             87.4642 %
Total Number of Instances              300
```

```
Classes at different positions are  @attribute Creditability {0,1}
confusion Matrix
[[ 50.  56.]
 [ 19. 175.]]

Evaluation from the perspective of class at position 0
TP  0.4716981132075472
FP 0.0979381443298969
Precision  0.7246376811594203
Recall  0.4716981132075472
F1 Score:  0.5714285714285714

Evaluation from the perspective of class at position 1
TP  0.9020618556701031
FP 0.5283018867924528
Precision  0.7575757575757576
Recall  0.9020618556701031
F1 Score:  0.8235294117647057
```

The precision, recall, accuracy, and F1 scores are as follows:

	Precision	Recall	F1 Score
0	**0.72**	**0.47**	**0.57**
1	**0.76**	**0.90**	**0.82**
Accuracy	**75%**		

From the results, the model accurately classified 75% of the instances. This is good, as a high accuracy value indicates that our model is making a large proportion of correct predictions. Focusing on class 1, we recorded high values for precision, recall and f1 score. The precision value of 0.76 indicates that the model accurately identified positive instances and did not make too many false positive (FP) predictions. Recall was 0.90, indicating that the model did not miss many positive instances.

6.2 Naive Bayes using sklearn

As with the decision tree, Naive bayes uses label encoding to convert categories to ordinal values. This means, it used different values of each feature as frequencies which semantically may or may not be correct, as a higher value (like 2 ,3 and more) would mean more weight. We will also run the naive bayes classifier on the one hot encoded dataset and compare the results.

Naive Bayes output with Label Encoded Data

```
Number of features used  20
Classes  ['0' '1']
Number of records for classes  [209. 491.]
Log prior probability for classes  [-1.20874608 -0.35463621]
Log conditional probability for each feature given a class
 [[-7.66053824 -5.09289346 -7.52920224 -7.2442712  -0.02385625 -7.77176387
  -7.16792253 -7.18335493 -7.34322437 -8.16470272 -7.27277271 -7.3616411
  -4.77281549 -7.37473764 -7.66053824 -7.98653918 -7.23113472 -8.16470272
  -7.97617639 -8.28352364]
 [-6.96381256 -5.09018055 -7.03265538 -7.02592638 -0.03094527 -7.19521844
  -6.79713562 -6.95892257 -7.02443718 -7.89278845 -6.9965557  -7.21662871
  -4.4432513  -7.02816435 -7.36613136 -7.65976479 -6.96451308 -7.8874834
  -7.68825674 -7.98173594]]
```

	precision	recall	f1-score	support
0	0.39	0.41	0.40	91
1	0.74	0.72	0.73	209
accuracy			0.63	300
macro avg	0.56	0.56	0.56	300
weighted avg	0.63	0.63	0.63	300

```
Confusion Matrix
[[ 37  54]
 [ 58 151]]
TP:  151 , FP:  54 , TN:  37 , FN: 58
```

Naive Bayes output with One Hot Encoded Data

```
Number of features used  64
Classes  ['0' '1']
Number of records for classes  [205. 495.]
Log prior probability for classes  [-1.22807036 -0.34652257]
Log conditional probability for each feature given a class
 [[ -5.07717035  -0.02043849  -7.11378092  -4.73218569  -7.94918926
   -8.1045397   -9.0550218   -9.30014426 -11.1789151  -10.05044985
  -10.6323714  -10.5810781   -8.84942256 -10.53228794  -9.96589246
   -9.46593651 -11.09190373 -11.96737246  -9.86323831  -9.77014789
  -12.47819809 -11.3795858  -10.80422166 -12.8836632  -10.53228794
   -8.55292986 -10.5810781  -11.3795858  -11.78505091 -10.44131616
  -10.74359703  -9.72666278  -9.32831514 -10.24460587  -9.70560937
  -10.74359703  -9.17009113  -9.01246219 -10.93775305  -8.33506336
  -11.09190373 -11.63090023 -10.31871384  -9.37211776  -9.96589246
   -9.20736252  -9.81561026  -9.68499008  -9.34270387  -9.70560937
   -9.93922422 -10.86876018  -8.53338526  -9.60651846  -8.74849664
  -10.175613   -12.19051602  -9.81561026  -8.77278933  -9.88793092
   -8.6715356   -9.30014426  -8.26854268 -11.96737246]
 [ -5.04925111  -0.02529     -6.93020402  -4.4202028   -7.65971884
   -7.86276653  -9.67035747  -9.40146789 -10.71714469  -8.69619935
  -11.64870289 -11.3804389   -8.6684748  -10.50008018  -9.06036065
   -9.58867944 -10.02399751 -12.2677421   -9.63894127  -9.18977173
  -12.2677421  -11.7287456  -11.16912981 -12.42189278 -10.65830419
   -8.58963474 -10.30162924 -10.63013331 -10.68729172  -9.57892326
  -10.71714469  -9.93698613  -9.06615777  -9.67035747  -9.35383984
  -11.1226098   -9.20301695  -8.58244047 -10.34245124  -8.09555505
  -11.3804389  -10.91781538  -9.97954574  -9.26489236  -9.8442044
   -8.86654472  -9.18977173  -9.4096312   -9.10770677 -10.11930769
  -10.2246682  -11.03559842  -8.17339754  -9.79481164  -8.28939645
  -10.50008018 -11.64870289  -9.6185324   -8.45791003  -9.95097237
   -8.55766044  -8.86179412  -8.05879415 -10.95555571]]
```

```
              precision    recall  f1-score   support

           0       0.45      0.53      0.49        95
           1       0.76      0.70      0.73       205

    accuracy                           0.65       300
   macro avg       0.61      0.61      0.61       300
weighted avg       0.66      0.65      0.65       300
```

```
Confusion Matrix
[[ 50  45]
 [ 61 144]]
TP:  144 , FP:  45 , TN:  50 , FN: 61
```

6.3 Comparison of Naive Bayes results

	Naive Bayes (Weka)	Naive Bayes (Label encoding)	Naive Bayes (One Hot encoding)
Precision (Class 0)	0.72	0.39	0.45
Recall (Class 0)	0.47	0.41	0.53
F1 score (Class 0)	0.57	0.40	0.49
Precision (Class 1)	0.76	0.74	0.76
Recall (Class 1)	0.90	0.72	0.70
F1 score (Class 1)	0.82	0.73	0.73
Accuracy	75%	63%	65%

The accuracy rate slightly improves with one hot encoding with an accuracy rate of 65% compared to the 63% obtained when label encoding was used.

However, the naive bayes algorithm performed better when run using the python weka package with an accuracy value of 75%.

6.4 Data Split Comparison

We questioned the impact of our data split strategy on the results. For all analysis, we split the dataset using a 7:3 ratio. 70% of the original dataset for training and 30% for testing. We will be running the naive bayes algorithm in weka with different split percentages and comparing the accuracy values.

% allocated for training	55%	60%	66%	75%	80%	85%	90%
Accuracy(%)	76.67%	74.5%	75%	77.2%	77.5%	77.33%	75%

Splitting the dataset into 66% for training and 34% for testing made no change to the previous results.

Splitting the dataset into 60% for training and 40% for testing saw a slight drop in the accuracy rate to 74.5%. However, we recorded an increase in False Negative instances from 22 to 25 and False positives from 63 to 77.

Interestingly, further reducing the split to 55% for training caused an increase in accuracy to 76.67%.

We increased the training set to 75% of the original dataset and we recorded an increase in the accuracy to 77.2%. However, after increasing the training set to 85 and 90%, the accuracy rates dropped.

We are unable to come to a definite conclusion on how the data splitting affects the model. From the table above, there is no clear pattern and there does not seem to be a clear correlation between them.

7. Analysis

7.1 Performance Metrics Comparison and Conclusion

Initially, we conducted a performance metrics analysis of the original dataset using various models to determine the optimal model fit for the original dataset.

Metric	Decision Tree Label Encoding	Decision Tree One-hot Encoding	Decision Tree by SAS	Naive Bayes Label Encoding	Naive Bayes One-hot Encoding
TP	201	154	626	151	144
FP	62	48	178	54	45
TN	29	47	122	37	50
FN	8	51	74	58	61

Figure 7.1. Performance Results of Confusion Matrix

The Decision Tree model with Label Encoding achieved the highest True Positive (TP) rate (201), but also had a relatively high False Positive (FP) rate (62). Similarly, the Naive Bayes model with Label Encoding had a high False Negative (FN) rate (58), indicating that it misclassified a significant number of positive instances. The decision tree model generated by SAS with 1000 entries had a (TP) rate of (626) and an (FP) rate of (178).

So, In our further analysis, we prioritized metrics like precision, recall, and F1-score specifically for class 1, which represents good credit applicants. This focus allowed us to evaluate the models' performance in correctly identifying individuals likely to have good credit. We aimed to determine the superior model for credit approval, distinguishing between class 0 (representing applicants with bad credit) and class 1 (representing applicants with good credit).

Metric	Decision Tree Label Encoding	Decision Tree One-hot Encoding	Naive Bayes Label Encoding	Naive Bayes One-hot Encoding
Precision (Class 0)	78%	48%	39%	45%
Recall (Class 0)	32%	49%	41%	53%

F1-score (Class 0)	45%	49%	40%	49%
Precision (Class 1)	76%	76%	74%	76%
Recall (Class 1)	96%	75%	72%	70%
F1-score (Class 1)	85%	76%	73%	73%
Accuracy	77%	67%	63%	65%
Macro Average Precision	77%	62%	56%	61%
Macro Avg Recall	64%	62%	56%	61%
Macro Average F1-score	65%	62%	56%	61%
Weighted Avg Precision	77%	67%	63%	66%
Weighted Avg Recall	77%	67%	63%	65%
Weighted Average F1-score	73%	67%	63%	65%

Figure 7.2. Performance Results of Different Classifiers

For precision (class 1), all models have relatively high values (76%), Decision Tree with Label Encoding having the highest Macro Average Precision (77%).

For recall (class 1), Decision Tree with Label Encoding has the highest value (96%), indicating that it correctly identifies a high proportion of actual good credit cases.

For F1-score (class 1), Decision Tree with Label Encoding also has the highest value (85%), indicating a good balance between precision and recall for predicting good credit cases.

The Decision Tree model with Label Encoding yielded the highest accuracy at 77%, outperforming other models tested.

Based on these observations, we decided the Decision Tree model with Label Encoding appears to be the best choice for credit approval, as it achieves high precision, recall, and F1-score for predicting good credit cases with the original data.

8. Post Predictive Analysis

We will be comparing the results from the decision tree and naive bayes classifiers on both the original and filtered data sets using the following performance metrics: Accuracy, Precision, Recall, and F1 Score.

8.1 Feature selection

We implemented 3 methods of selecting features to filter our dataset: Correlation Matrix, BestFirst Algorithm, and Chi-Square Test

8.1.1 Correlation Matrix

During data preparation, we were able to identify 3 attributes with statistical analysis that showed low or no correlation between them and the target variable. Using the correlation matrix, we decided on these 3 attributes to filter out from the original dataset:

1. Duration in current address
2. Telephone
3. Type of apartment

The new filtered dataset will contain 18 variables(including the target variable) and we will rerun the algorithms on the filtered dataset.

For predicting class '1'	**Decision Tree** Original Dataset	**Decision Tree** Filtered Dataset	**Naive Bayes** Original Dataset	**Naive Bayes** Filtered Dataset
Accuracy	77%	77%	75%	75.67%
Precision	0.76	0.76	0.76	0.76
Recall	0.96	0.96	0.90	0.90
F1 Score	0.85	0.85	0.82	0.83

For predicting class '0'	**Decision Tree** Original Dataset	**Decision Tree** Filtered Dataset	**Naive Bayes** Original Dataset	**Naive Bayes** Filtered Dataset
Accuracy	77%	77%	75%	75.67%
Precision	0.78	0.78	0.72	0.73

Recall	0.32	0.32	0.47	0.49
F1 Score	0.45	0.45	0.57	0.58

8.1.2 BestFirst Algorithm

With the BestFirst algorithm implemented using the weka package in python, 3 attributes were selected as the most consistent, non-redundant, and relevant features to use in constructing an optimized model: Account Balance, Duration of Credit (month), and Payment Status of Previous Credit.

We filtered the dataset on these attributes, and ran the algorithm. Results are shown below:

For predicting class '1'	**Decision Tree** Original Dataset	**Decision Tree** Filtered Dataset	**Naive Bayes** Original Dataset	**Naive Bayes** Filtered Dataset
Accuracy	77%	70%	75%	73.3%
Precision	0.76	0.73	0.76	0.73
Recall	0.96	0.90	0.90	0.94
F1 Score	0.85	0.81	0.82	0.82

For predicting class '0'	**Decision Tree** Original Dataset	**Decision Tree** Filtered Dataset	**Naive Bayes** Original Dataset	**Naive Bayes** Filtered Dataset
Accuracy	77%	70%	75%	73.3%
Precision	0.78	0.50	0.72	0.76
Recall	0.32	0.23	0.47	0.36
F1 Score	0.45	0.32	0.57	0.49

8.1.3 Chi-Square Test

We implemented another method for feature selection called the chi-square test. The chi-square (χ^2) test is a statistical test used to determine whether there is a significant association between two categorical variables. We used the chi-square test to select the top 6 features and ran the naive bayes classifier using these features: Credit Amount, Duration of Credit(month), Account Balance, Value Savings/Stocks, Age (years), Payment Status of Previous Credit.

selcted features:	Feature	Chi2 Score	P-Value
4	Credit Amount	58264.415475	0.000000e+00
1	Duration of Credit (month)	321.030795	8.637197e-72
0	Account Balance	75.474269	3.702000e-18
5	Value Savings/Stocks	37.937451	7.304944e-10
12	Age (years)	30.178268	3.941008e-08
2	Payment Status of Previous Credit	24.103752	9.128229e-07
11	Most valuable available asset	9.503534	2.050765e-03
6	Length of current employment	5.799899	1.602710e-02
13	Concurrent Credits	2.243436	1.341825e-01
7	Instalment per cent	2.204619	1.375979e-01
8	Sex & Marital Status	1.452292	2.281605e-01
3	Purpose	0.860039	3.537278e-01
15	No of Credits at this Bank	0.495516	4.814772e-01
19	Foreign Worker	0.231483	6.304266e-01
18	Telephone	0.228056	6.329691e-01
16	Occupation	0.157484	6.914836e-01
9	Guarantors	0.125806	7.228210e-01
14	Type of apartment	0.047817	8.269070e-01
10	Duration in Current address	0.003766	9.510663e-01
17	No of dependents	0.001031	9.743885e-01

For predicting class '1'	**Decision Tree** Original Dataset	**Decision Tree** Filtered Dataset	**Naive Bayes** Original Dataset	**Naive Bayes** Filtered Dataset
Accuracy	77%	74%	75%	73%
Precision	0.76	0.76	0.76	0.73
Recall	0.96	0.92	0.90	0.94
F1 Score	0.85	0.83	0.82	0.82

For predicting class ‘0’	**Decision Tree** Original Dataset	**Decision Tree** Filtered Dataset	**Naive Bayes** Original Dataset	**Naive Bayes** Filtered Dataset
Accuracy	77%	74%	75%	73.3%
Precision	0.78	0.65	0.72	0.76
Recall	0.32	0.34	0.47	0.35
F1 Score	0.45	0.45	0.57	0.48

8.2 Summary of Post Predictive Analysis

We compare the results from running the 2 classifiers on the filtered data sets gotten from the 3 different feature selection methods:

Decision Tree for class ‘0’ and ‘1’	Correlation Matrix (CM)	BestFirst Algorithm (BF)	Chi-Square Test (CS)
Accuracy	77%	70%	74%
Precision (class ‘0’)	0.78	0.50	0.65
Recall (class ‘0’)	0.32	0.23	0.34
F1 Score (class ‘0’)	0.45	0.32	0.45
Precision (Class ‘1’)	0.76	0.73	0.76
Recall (Class ‘1’)	0.96	0.90	0.92
F1 Score (Class ‘1’)	0.85	0.81	0.83

The decision tree seems to have better success with the filtered dataset gotten from the correlation matrix with an accuracy value of 77%. Comparing the precision values, we see that the decision tree using the CM filtered dataset performed better at correctly labeling high and low creditability customers. With predicting class 1, the precision value is 0.76, meaning the algorithm was able to correctly label highly credible customers as having high creditability. The recall value of 0.96 signifies that the model did close to perfect in not mission any true positives.

Furthermore, for predicting class ‘0’ across the 3 filtered datasets, we recorded relatively low values (<0.5) for both recall and F1 score. We can deduce from this, that the model was only able to identify a small proportion of positive instances and is missing many positive instances, in this case, instances of low creditability.

Naïve Bayes for class ‘0’ and ‘1’	Correlation Matrix (CM)	BestFirst Algorithm (BF)	Chi-Square Test (CS)
Accuracy	75.67%	73.3%	73%
Precision (class ‘0’)	0.73	0.76	0.76
Recall (class ‘0’)	0.49	0.36	0.35
F1 Score (class ‘0’)	0.58	0.49	0.48
Precision (Class ‘1’)	0.76	0.73	0.73
Recall (Class ‘1’)	0.90	0.94	0.94
F1 Score (Class ‘1’)	0.83	0.82	0.82

As with the decision tree, the naive bayes classifier recorded better success with the filtered dataset obtained from the correlation matrix with an accuracy value of 75.67%, which is not far off from the decision tree,indicating that the model was able to make a large proportion of correct predictions. Comparing the precision values, we see that the naive bayes model recorded a lower precision value of 0.73 for class ’0’ using the CM dataset than the other 2. However, this disadvantage did not carry through the other metrics even for class ‘1’. Low recall and F1 score values (< 0.5) were recorded for predicting class ‘0’.

With predicting class 1, the precision value is 0.76, meaning the algorithm was able to correctly label highly credible customers as having high creditability. The recall value of 0.90 signifies that the model did close to perfect in not mission any true positives. However, with the other 2 datasets, we observe higher recall values of 0.94.

An observation worth mentioning is that both models seem to be missing many positive instances in predicting class ‘0’, low creditability. This means our models are doing better at correctly identifying customers with high creditability than low creditability.

Now, we compare the results of the 2 classifiers on both the original dataset and the filtered dataset using the correlation matrix

	Decision Tree (Original Dataset)	Decision Tree (Filtered Dataset)	Naive Bayes (Original Dataset)	Naive Bayes (Filtered Dataset)
Accuracy	77%	77%	73%	75.67%
Precision (class '0')	0.78	0.78	0.72	0.73
Recall (class '0')	0.32	0.32	0.47	0.49
F1 Score (class '0')	0.45	0.45	0.57	0.58
Precision (Class '1')	0.76	0.76	0.76	0.76
Recall (Class '1')	0.96	0.96	0.90	0.90
F1 Score (Class '1')	0.85	0.85	0.82	0.83

From the previous comparisons of the results of running both algorithms on the filtered datasets, we concluded that the algorithms obtained overall better results when building models on the algorithm results from the original dataset with the featured dataset, shown in the table above. The decision tree algorithm obtained the same results in both cases, with an accuracy of 77% each time. However, the naive bayes performed better on the filtered data set with an accuracy of 75.67% than on the original dataset with an accuracy value of 73%, even though the precision, recall and F1 score values are only slightly higher.

We can conclude from this observation that running the decision tree algorithm on either the original or the filtered dataset gets us the most accurate model in this scenario. We could also conclude that the reason for obtaining the same result is that the features that were filtered out of the dataset, have little or no correlation with the target attribute. So, their presence in the dataset is inconsequential.

The question of how many features to include in the dataset to get the best model comes up. From our analysis, running either algorithm on 3 best or 6 best features did not obtain a better result than running the algorithm on 17 features.

9. Conclusions and Recommendations

After our analysis and modeling, we would recommend to the client that the following features can be excluded from the model: Duration in current address, Telephone and Type of apartment as they have been proven to not have adverse effects on the model created. Also, the client can mitigate their risk by using the decision tree model to correctly assess the risk customers pose when making loan approval decisions. Though from testing, we ascertained that the filtered data produced the same result as the original dataset.

Another recommendation would be to continuously refine and optimize the feature selection process to identify the most predictive features for risk credit assessment. In real life, the dataset may be a lot larger, therefore, we require more testing to validate whether these are the only features necessary to create an optimal model. The dataset might neglect some other information that would be helpful in validating the clients. For example, it might be helpful to know the customer's family income, previous year's tax notice of assessment, or current credit score.