SpotLight Report: Credit Cards Risk Assessment

Maheen Asim

4/15/2024

1. Introduction

Credit has become a cornerstone of modern economics, underpinning countless transactions in today's financial landscape. Since the inception of currency and lending, the concept of creditworthiness has been pivotal in fostering trust between borrowers and lenders, catalyzing the growth of commerce and personal finance. The lineage of credit can be traced back to ancient civilizations, yet it is in our modern era, replete with its intricate economic webs, that the assessment of credit risk has evolved into a sophisticated art. Credit risk analysis is not only fundamental to the stability of banks and financial institutions but also to the economic empowerment of individuals seeking to navigate the complexities of modern financial landscapes.

The evolution of credit risk assessment has been marked by an increasing reliance on data-driven insights. From simple ledger entries to complex algorithms processing vast datasets, the tools and techniques employed in determining creditworthiness have undergone significant refinement. In this context, datasets like the German credit data provide a critical foundation for understanding and innovating in the field of risk management. They represent a distilled compilation of human economic behaviors, aspirations, and limitations, each entry encapsulating a story of financial trust. Assessing a dataset that classifies individuals as good or bad credit risks offers more than just a glimpse into their financial reliability; it opens a dialogue on the broader socio-economic factors at play. It underscores the multifaceted nature of financial health and challenges the analytical mind to discern patterns that govern financial trustworthiness. Therefore, the goal of this report is to classify people as good or bad credit risks described by all given set of attributes and decide on strategy that would be better for the financial institutions.

2. Data Preparation

The dataset in question, known as the German Credit Data, is sourced from the UCI Machine Learning Repository and serves the purpose of evaluating credit risk, classifying individuals into 'good' or 'bad' credit risks based on various attributes. To refine this dataset for analysis, an initial cleaning process was executed which includes, removing empty strings and any records with missing values were omitted. Additionally, the variable 'X' was identified as redundant, merely representing a row count, and thus was removed, leaving us with 10 variables and 522 observations.

```
set.seed(1234)
credit_card<- read.csv("~/Mscs 341 S24/Submit/Spotlight/gc.csv")
credit_card <-as_tibble(credit_card)%>%
  mutate(across(everything(), ~na_if(str_trim(.), "")))
credit_card <-na.omit(credit_card)%>%
  subset(select = -c(X))
```

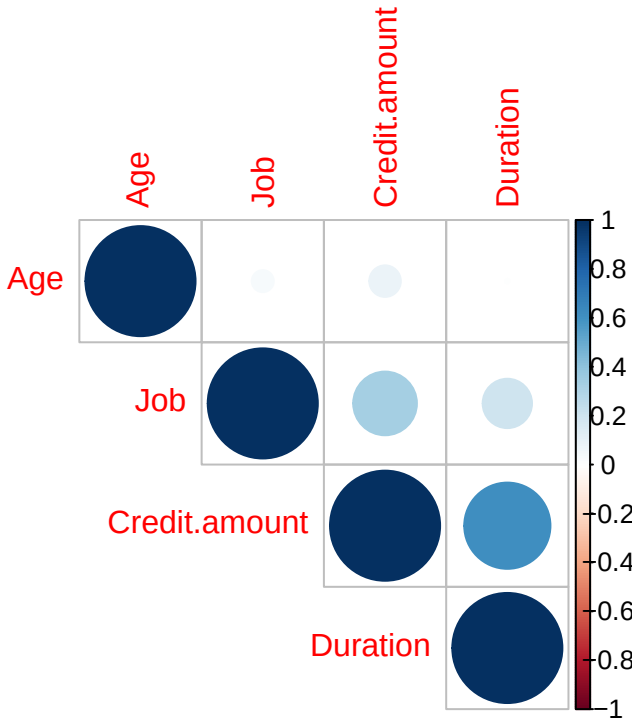
Table 1: Variables Description

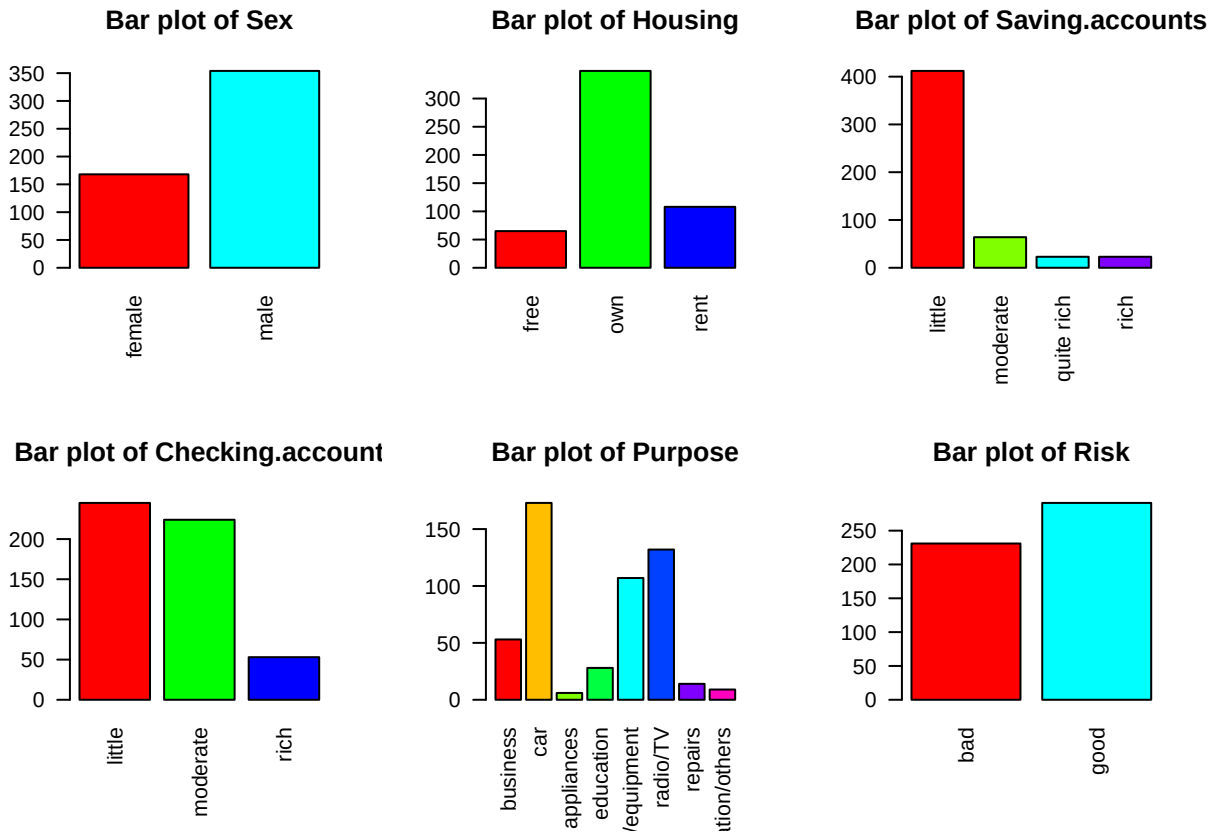
	Variable_Name	Variable_Class	Description
Age	Age	character	Age
Sex	Sex	character	Gender (Female/Male)
Job	Job	character	Number of Jobs
Housing	Housing	character	Type of Hoosuing (own, rent, free)
Saving.accounts	Saving.accounts	character	Condition of Savings Account (little, moderate, rich, quite rich)
Checking.account	Checking.account	character	Condition of Checkings Account (little, moderate, rich, quite rich)
Credit.amount	Credit.amount	character	Amount of money in Account
Duration	Duration	character	Number of Months for Account established
Purpose	Purpose	character	Purpose of account creation
Risk	Risk	character	Risk (Good/Bad)

3. Explanatory Data Analysis

Primary explanatory analysis is executed to examine the dataset to understand an overall picture, probing into distributions, spotting trends, and observing correlations. Histograms illustrated the age distribution, loan amounts, and loan durations, suggesting a younger demographic and smaller loan sizes are more common. The bar plots for categorical variables like job, housing, and sex showed variances in frequency, while a correlation matrix revealed insightful relationships, such as between credit amount and loan duration. Overall, this initial EDA helped to uncover patterns and connections within the data, laying a foundational understanding for subsequent, more intricate analyses.

```
## named list()
```





4. Modeling

In this section of the analysis, I have employed five distinct regression methods to delve into the German Credit Data: K-Nearest Neighbors (KNN), Ridge Regression, Lasso Regression, Random Forest, and Boosting. Each of these techniques brings a unique perspective to the modeling process, allowing for a comprehensive understanding of the data's predictive capacity. Prior to applying these models, I took the necessary step of transforming categorical variables into dummy variables and converting them into factors to ensure they are appropriately interpreted by the regression algorithms. With these modifications in place, I proceeded to partition the dataset into training and testing sets. A random 70% split was implemented to construct the training dataset, ensuring that it contains a majority of the data for model learning. The remaining 30% formed the testing dataset, which will be used to evaluate the models' performances.

```
#Divide the dataset into training and testing datasets
set.seed(123456)
credit_split <- initial_split(credit_card, prop= 0.7)
credit_train_tbl <- training(credit_split)
credit_test_tbl <- testing(credit_split)
```

In the analysis, I created a universal recipe to preprocess the data in preparation for modeling. This recipe was crafted to ensure consistency across the different models applied.. It includes dummy coding of all nominal (categorical) variables to transform them into a format suitable for regression analysis while excluding the outcome variable, 'Risk'. The recipe also identifies and removes any predictor variables with zero variance, which are of no value to the models, and could potentially skew the results. Lastly, it normalizes all predictor variables, which is a crucial step to standardize the range of independent variables, ensuring that each one

contributes equally to the analysis. This preparatory step is critical as it not only streamlines the modeling process but also helps in enhancing the performance and interpretability of the models.

```
data_recipe <- recipe(Risk ~ ., data = credit_train_tbl) %>%
  step_dummy(all_nominal(), -all_outcomes()) %>%
  step_zv(all_predictors()) %>%
  step_normalize(all_predictors())
```

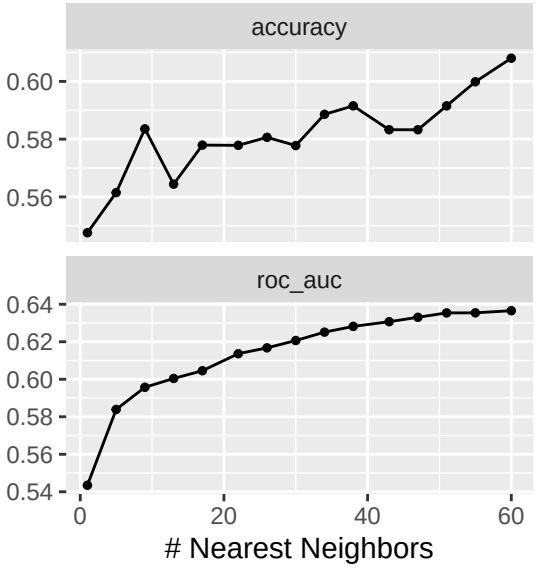
a. K-nearest Neighbors Technique

For the initial modeling approach, I employed the K-nearest Neighbors (KNN) technique, utilizing it to forecast whether the 'Risk' variable would be classified as 'good' or 'bad' based on the other variables present in the data. Recognizing that the training set for each fold in our cross-validation scheme contains just over 300 observations, I configured a tuning grid to optimize the 'k' parameter, which determines the number of neighbors considered in the algorithm. This grid spans a range from 10 to 300 neighbors, divided into 10 distinct levels. By systematically varying 'k', I aim to identify the optimal number of neighbors that yields the highest prediction accuracy, balancing the model's ability to generalize without overfitting to the training data.

```
set.seed(1234)
#Knn model specification
knn_model_f <- nearest_neighbor(neighbors = tune()) %>%
set_engine("kknn") %>%
set_mode("classification")
knn_workflow_f <- workflow() %>%
add_recipe(data_recipe) %>%
add_model(knn_model_f)

# #Prepare the tuned results using cross-validation
credit_knn_fold <- vfold_cv(credit_train_tbl, v = 10) #starta?
  neighbors_grid_tbl <- grid_regular(neighbors(range = c(10, 300)), levels = 10)
  tune_results_1 <- tune_grid(object = knn_workflow_f, resamples = credit_knn_fold, grid = neighbors_gri
  #autoplot(tune_results_1)
```

The plot showed that the accuracy is decreasing with higher values of k, so I decide to focus our efforts using smaller values of k (say from 1 to 60).



Finally we decide to optimize our value of knear based on roc_auc:

Now we can finally check the ROC curve of our model and calculate the performance of our model using roc_auc using our testing dataset.

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.635
```

The ROC-AUC plot and value are key indicators of the K-Nearest Neighbors model's performance. The ROC curve of the model shows a moderate discriminatory ability with an AUC of 0.635. This value, close to 1 suggests that the KNN model has a fair degree of predictive power in distinguishing between good and bad credit risks.Now lets see the confusion matrix:

```
##            Truth
## Prediction bad good
##       bad   34   24
##       good  40   59
```

The confusion matrix offers a concrete look at the model's predictive accuracy, presenting the number of true positives, true negatives, false positives, and false negatives. In the matrix, we see that the model predicted 'good' credit risk correctly for 59 cases but incorrectly classified 24 'bad' risks as 'good'. Similarly, it correctly identified 34 'bad' risks but labeled 40 'good' risks as 'bad'. This highlights the balance model strikes between sensitivity and specificity from roc-auc. Lets explore a different model now.

b. Tree-Based Modeling: Random Forest

For a more comprehensive analysis, I next implemented a tree-based modeling technique, specifically Random Forest, which is renowned for its effectiveness in classification tasks. Utilizing a 10-fold cross-validation approach, this method was applied to the credit risk dataset. Random Forest operates by constructing a multitude of decision trees during training and outputting the class that is the mode of the classes (classification) of the individual trees.

In practice, the more trees the model includes, the more robust it becomes against overfitting. By building 300 trees, given the 365 observations in the training data, each tree has a chance to learn from a slightly different subset of the data, thus increasing the model's overall diversity in terms of the features and data points considered. This diversity is a strength of the Random Forest technique, ensuring that individual model idiosyncrasies are less likely to influence the overall prediction, yielding a more stable and generalized performance. The use of 300 trees is particularly suitable in this context as credit train table split is 365 values so choosing 300 trees. There are 9 predictors in our dataset, so the range for parameter mtry is from 1 to 9.

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.693
```

```
##           Truth
## Prediction bad good
##       bad   21    6
##       good  53   77
```

The ROC-AUC plot and the associated AUC value of 0.693 for the Random Forest model indicate a slight improvement in the model's ability to differentiate between good and bad credit risks compared to the KNN model. The ROC curve being closer to the top-left corner suggests that the Random Forest model has better discriminative power. The model correctly predicted 'good' credit risk 77 times, but misclassified 'good' credit as 'bad' in 53 instances. It accurately identified 'bad' credit risk 21 times, yet labeled 6 'bad' credit cases as 'good'. This performance contrasts with the earlier KNN model, which showed a different balance between sensitivity (correctly identifying 'bad' cases) and specificity (correctly identifying 'good' cases). While the Random Forest model achieved a higher ROC-AUC score of 0.693, indicating better overall ability to distinguish between good and bad credit risks compared to the KNN model's 0.635, it demonstrates challenges particularly in minimizing the false classification of 'good' credits as 'bad'. This suggests that the Random Forest model, while having a higher overall predictive accuracy, might be overly cautious, potentially leading to higher false negatives (good credits classified as bad).

c. Tree-Based Modeling: Boosting

In continuing with the tree-based approaches, I transitioned to Boosting, specifically utilizing the XGBoost engine, which is a powerful and widely-used machine learning algorithm. The model was set up with 300 trees, aligning with the Random Forest approach to maintain a consistent comparative basis. The Boosting model was tuned for several hyperparameters: tree depth, learning rate, loss reduction, and minimum number of observations required in the nodes. These parameters were tuned using a 10-fold cross-validation method to ascertain the most effective combination that maximizes predictive performance while preventing overfitting. A grid of potential values was created, assigning five levels to each hyperparameter, creating a comprehensive search space for the optimal model settings.

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.691
```

```
##           Truth
## Prediction bad good
##       bad   37   15
##       good  37   68
```

The Tree-Based Boosting model yielded an ROC-AUC value of 0.691. The Boosting model's ROC curve's ascent towards the top-left corner suggests an incrementally better true positive rate relative to the false positive rate, which is a desirable trait in a predictive model. From the confusion matrix, the model correctly predicted 'bad' credit risk 37 times and 'good' credit risk 68 times. However, there were 15 instances where a 'bad' credit risk was incorrectly classified as 'good', and 37 cases where a 'good' credit risk was labeled as 'bad'. In comparison to the Random Forest model which had an accuracy of 77 correct predictions for 'good' credit risk and 21 for 'bad' credit risk with fewer miss-classifications of 'bad' as 'good', the Boosting Tree model shows an improvement in accurately identifying 'bad' credit risks (37 correct predictions compared to Random Forest's 21), but it has increased errors in falsely predicting 'good' credit as 'bad' (37 instances)

d. Ridge Regression

Now, Ridge Regression is employed to predict credit risk, with a particular focus on optimizing the penalty parameter through 10-fold cross-validation. This approach aims to balance model complexity and predictive accuracy. By tuning the penalty, the model seeks to minimize overfitting while retaining the predictive power, a critical balance in credit risk assessment.

Now we are ready to optimize our penalty value.

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.685
```

```
##           Truth
## Prediction bad good
##       bad   41   20
##       good  33   63
```

The results from the Ridge Regression model present an ROC-AUC value of 0.685, which indicates a slight decrease in predictive accuracy when compared to both the Random Forest and Boosting models and KNN. I suggests that Ridge Regression might not capture the complexities of the data set as effectively as the other methods. It accurately predicted 'bad' credit risk 41 times and 'good' credit risk 63 times. However, it incorrectly classified 20 instances of 'bad' credit risk as 'good', and 33 instances of 'good' credit risk as 'bad'. Despite these results being similar to those of the Boosting Tree model in terms of the number of correct predictions, the Ridge Regression model has a lower ROC-AUC score of 0.685. This suggests that while Ridge Regression manages to capture some central tendencies within the data, it does not provide the same level of nuanced decision-making as the more complex tree-based models.

e. Lasso Regression

In the Lasso Regression analysis, I tailored the model to optimize the balance between fit and complexity by adjusting the penalty term, focusing exclusively on regularization to encourage a sparse solution. The mixture parameter was set to 1, indicating a pure lasso model where some coefficients can be shrunk to zero, performing implicit feature selection.

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.315
```

```
##            Truth
## Prediction bad good
##       bad   41   20
##       good  33   63
```

The Lasso Regression model's performance, as reflected in the ROC-AUC value of 0.307, indicates a significant decline in its ability to distinguish between good and bad credit risks. This value is considerably lower than what was observed in the other models, suggesting that Lasso Regression may not be capturing the complexity or the patterns in the dataset effectively. The confusion matrix from the Lasso Regression model shows that it correctly identified 'bad' credit risk 41 times and 'good' credit risk 63 times. However, it incorrectly classified 20 instances of 'bad' credit risk as 'good', and 33 instances of 'good' credit risk as 'bad'. This performance suggests that the Lasso Regression model, similar to the Ridge Regression in terms of correct and incorrect classifications, also struggles with a high number of false negatives (good credits predicted as bad) and false positives (bad credits predicted as good).

5. Variable Importance Plots

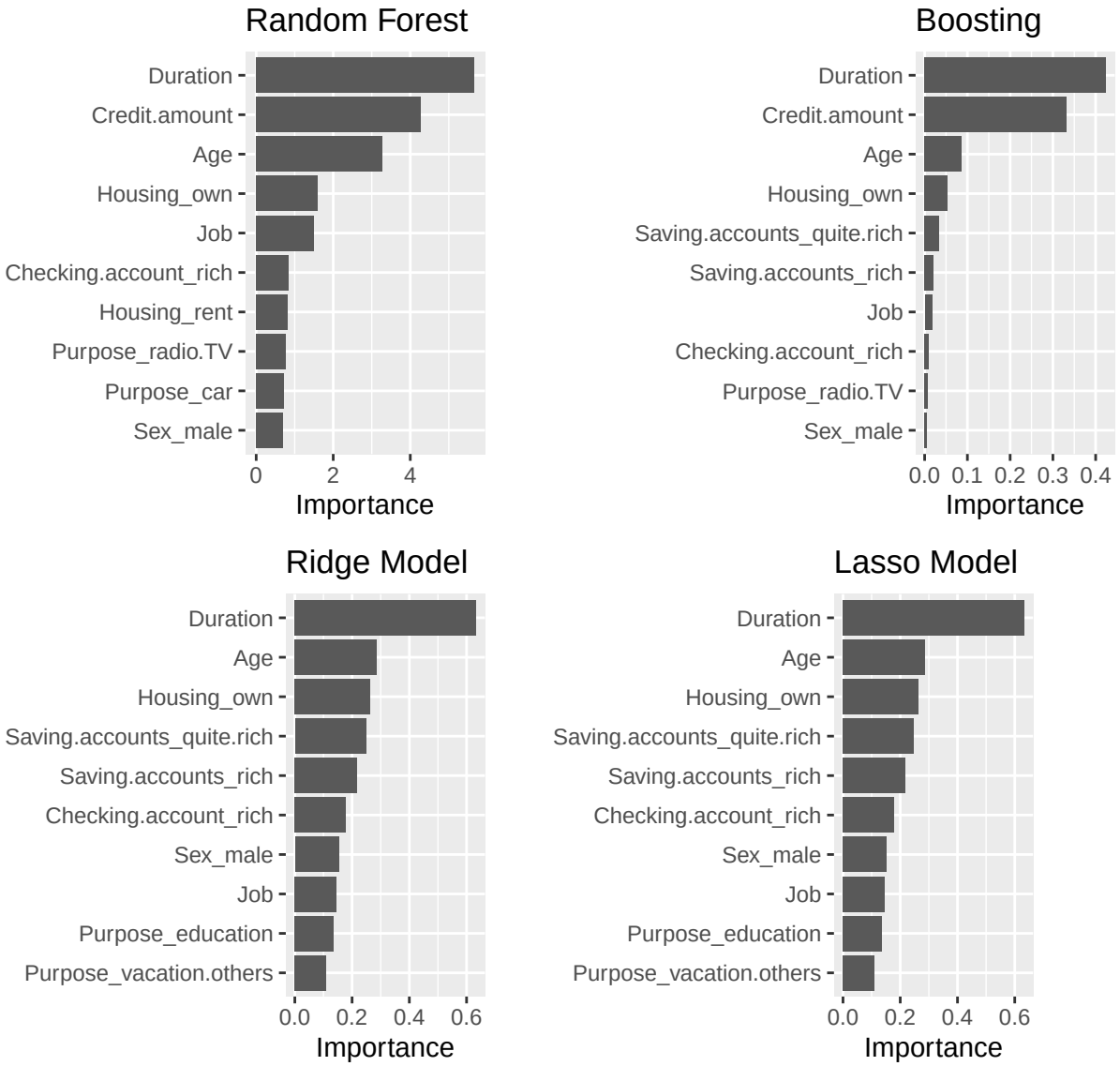
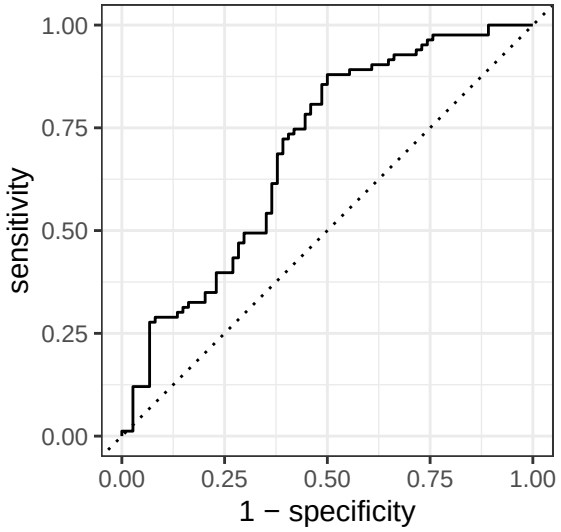


Table 2: Models Summary

Model	ROC AUC	Top 5 Important Variables
KNN Model	0.635	N/A
Tree-Based: Random Forest Model	0.693	Duration, Credit.amount, Age, Housing_own, Job
Tree-Based: Boosting Model	0.691	Duration, Credit.amount, Age, Housing_own, Savings.accounts_quite.rich
Lasso Model	0.307	Duration, Age, Housing_own, Savings.accounts_quite.rich, Savings.accounts_rich
Ridge Model	0.685	Duration, Age, Housing_own, Savings.accounts_quite.rich, Savings.accounts_rich

6. Conclusion

In conclusion, the comparative analysis of five different models revealed varied performance levels in predicting credit risk within the German Credit Data. The *Tree models* overall performed the *best* at the classification with both Boosting model with 0.691 ROC-AUC and the Random Forest delivered a marginally better ROC-AUC value of 0.693, outperforming all other models. The Lasso model performed the worst with the ROC-AUC value of 0.307. Now, for the best model I would like to explore the ROC of the model and whether I can change the parameters to change specificity/sensitivity.



Choosing the optimal threshold from a ROC curve involves balancing the trade-off between sensitivity (true positive rate) and specificity (1 - false positive rate) based on the specific needs. We saw that the Random Forest demonstrates challenges particularly in minimizing the false classification of 'good' credits as 'bad'so focusing on improving sensitivity (or recall) would be beneficial. Sensitivity measures the ability of the model to correctly identify true positives, in this case, 'good' credits. By aiming to enhance sensitivity, we'll reduce the rate of false negatives, meaning fewer good credit applications will be mistakenly labeled as bad. The Youden's Index helps find the optimal cutoff threshold that maximizes the difference between true positive rate and false positive rate (J = sensitivity + specificity - 1).Calculating for each point on the ROC curve

and choose the threshold with the highest Youden's Index and choosing Optimal Threshold Selection.

```
## # A tibble: 1 x 4
##   threshold sensitivity specificity youdens_index
##       <dbl>       <dbl>       <dbl>         <dbl>
## 1     0.531       0.880         0.5         0.380
```

The ROC AUC score has now decreased from 0.693. This indicates that while the chosen threshold might maximize the balance between sensitivity and specificity at that particular point, it may not necessarily optimize the model's overall discriminatory power across all thresholds. The new ROC value of 0.53, suggests that this point maximizes the difference between true positive rate and false positive rate. However, a specificity of 0.5 implies that the model is only as good as random at identifying negative cases ('bad' credits) at this threshold. Implications of this are that The threshold maximizes a specific trade-off between sensitivity and specificity, which might be aligned with specific business goals (e.g., reducing the number of 'good' credits falsely classified as 'bad'). Tailored Decision Making: Useful in scenarios where the cost of false negatives (good credits wrongly labeled as bad) is deemed higher than the cost of false positives. However, A lower ROC AUC score indicates that the overall ability of the model to distinguish between 'good' and 'bad' across all thresholds is reduced. This might mean that while the model performs well at the chosen threshold, it is less effective overall. It also has the Risk of Overfitting to a Specific Criterion. However, I suggest in terms of context of credit card risk assessment would be to one with a higher specificity than sensitivity, in simpler words - that the model is very effective at correctly identifying "bad" risk cases as "bad." As our primary concern is to avoid falsely classifying "bad" risks as "good," then a high specificity (nearly 100%) might be desirable. For instance, in financial institutions where avoiding bad loans is critical, ensuring almost no loan classified as safe (good) turns out bad is paramount. However, many potential "good" risks (safe loans, reliable customers, etc.) will be classified as "bad." This conservative approach minimizes risk but at the cost of potential gains from these "good" cases. Therefore, we should prefer our initial model of 0.693 with higher specifity than sensitivity because trying to optimize it reduces the quality of results for interpretations.

Advantages: The model is highly effective at identifying applicants who are bad risks, meaning it almost never approves applicants who are likely to default. This can be critical for minimizing financial losses from bad debts.R It reduces the risk of credit losses by preventing credit card approvals for high-risk individuals, potentially leading to lower charge-offs and better financial health for the credit issuer.

Disadvantages: The model fails to identify the vast majority of "good" applicants, classifying them as "bad." By not granting credit to viable candidates, the issuer misses out on potential revenue from interest, fees, and customer engagement with other financial products.