REPORT ON CREDIT RISK DATASET

Data Science and Analytics DATA8001

Student Name: Edward Duffy
Student Number: R00257226
Munster Technological University

Contents

Introduction:

For this task I was provided with a dataset which detailed the credit worthiness of current customers within a financial institution. The goal of this task was to take this current dataset and develop a model which can be used by the company to determine the credit worthiness of potential customers in the future. This was done through the use of:

- Performing Exploratory Data Analysis
- Key decision making as to how we cleaned the dataset
- Perform entropy calculations to determine the root and second node splits for our decision tree's.
- Utilize random forests to develop a model that could predict a customer's credit standing by using the historical data provided by the company.
- Assess our models using cross fold validation, bootstrapping and random sampling.
- Attempting to improve our model by developing new features through interaction.
- Developing a GDPR compliant model without compromising the accuracy on unseen data.
- Developing a strategy to detect incorrect gradings.

The dataset I was provided with contained an estimated 15,368 data points and contained the following variables:

ID – *Unique Identifier for each customer – Numeric (Integer):*

Checking.Acct – *Describes the checking account status of each customer - Categorical (Character):*

Credit.History – *Describes the credit history of each customer - Categorical (Character):*

Loan.Reason – *Contains the reason for the loan application - Categorical (Character):*

Savings_Acct – *Describes the savings account status of each customer - Categorical (Character):*

Employment – *applicants current employment status - Categorical (Character):*

Personal_Status – *Provides information on applicants personal status - Categorical (Factor):*

Housing – *Provides information on applicants housing situation - Categorical (Character):*

Job.Type – *Provides information on applicants job status - Categorical (Character):*

Foreign.National – *information as to if applicant is a foreign national - Categorical (Character):*

Months.since.Checking.Acct.opened – *Information on account age - Continuous (Numeric)*

Residence.Time.In.current.district – *how long customer has resided - Continuous (Numeric)*

Age – *Provides customers age – Continuous (Numeric)*

Credibility_score – *Score of the credit worthiness of an individual – Continuous (Numeric)*

Check – *A field created by a data analyst as a check on Credit.Standing – Numeric (Integer)*

Reg.State – *Indicates registration state of applicant - Character (Categorical)*

Credit.Standing - *Credit standing of customer Good / Bad – Character (Categorical)*

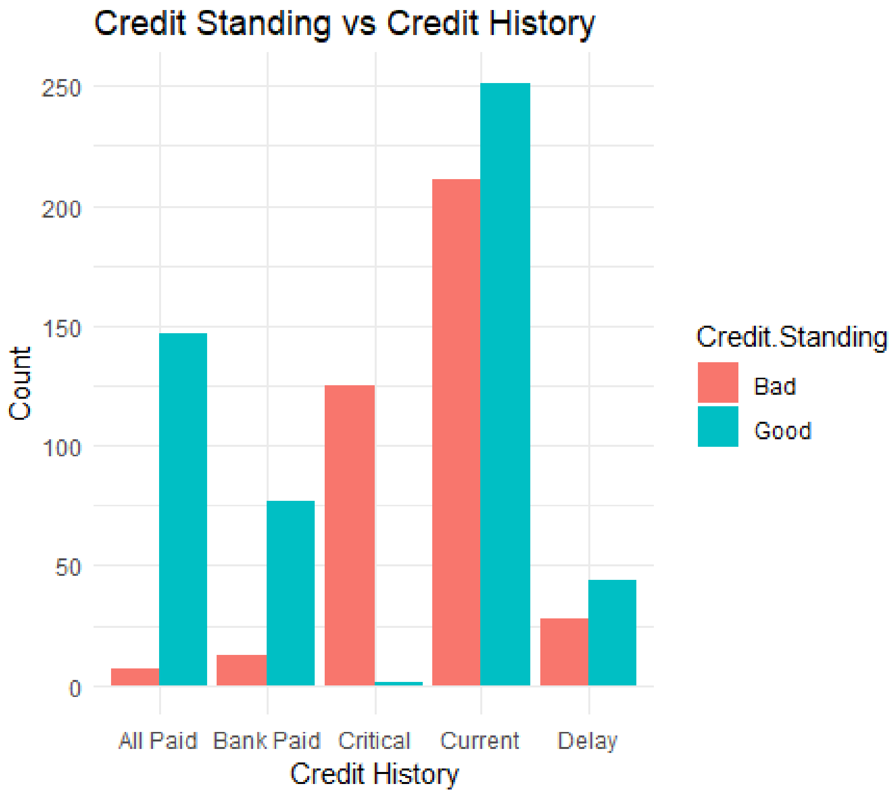
Exploratory Data Analysis:

Prior to portioning the dataset into training and test, a comprehensive exploratory data analysis (EDA) was conducted. The aim was to elucidate the underlying structure and relationships within the dataset to inform our imputation strategies. Performing EDA before imputing missing values is crucial for reasons such as:

- Identify patterns or anomalies that would be obscured following imputation.
- Insights into distribution and relationships of variables to make more informed decisions as to how imputation should be approached.
- Pre-imputation analysis ensures that changes made during cleaning are informed and do not inadvertently introduce bias.

Variable relationship explored:

- Credit standing vs Credit history.

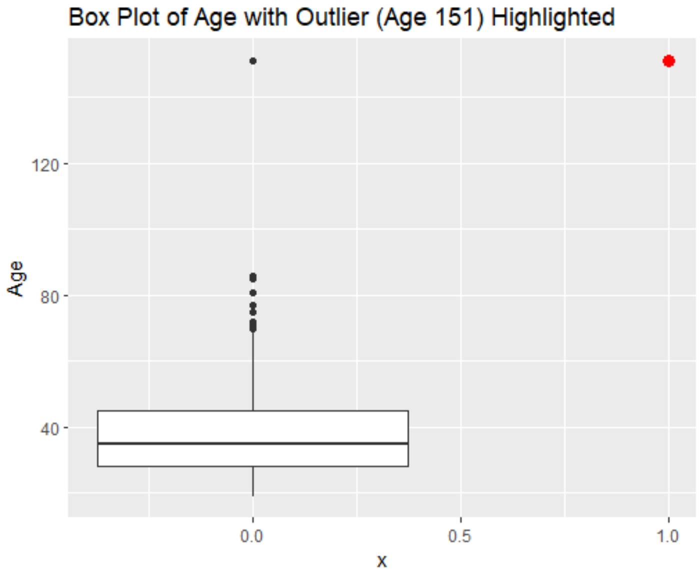


One of the key relationships was between credit history and standing. Our analysis indicated a discernible influence of credit history on credit standing. Categories representing positive standing such as ‘All Paid’, ‘Bank Paid’, predominantly aligned with ‘good credit standings. In contrast to ‘Critical’ and ‘Delay’ which suggested bad credit standings. This is one of the key relationships we will utilize when building our models.

Dataset Integrity Analysis:

Analysing the integrity of the dataset highlighted many issues that required rectification before proceeding such as:

- **Consistency in categorical values:** Standardizing of values such as ‘single’ instead of ‘Single’
- **Employment Anomalies:** I identified that people classed as ‘Unemployed’ were down as having jobs in ‘Management’, I avoided fixing this as it may introduce inaccuracies when modelling.
- **Negative Values in Residence Time:** I discovered that negative values were in the dataset, these were replaced with their corresponding positive value to maintain logical consistency.
- **Missing Values:** I imputed missing categorical values with the mode and numeric with the mean.
- **Outliers:** I replaced extreme outliers (e.g age 151) with the mean to prevent skew in analysis.



I removed the outliers within the dataset for the following reasons:

- **Enhanced model accuracy:** They can skew our results and lead to inaccurate models.
- **Improve data representation:** Our models better represent the majority of our data
- **Avoid misleading trends:** Can create misleading trends that don’t represent the data

Imputation Considerations:

The advantages are:

- **Mean Imputation:** Preserves the sample mean and maintains central tendency within our numeric data.
- **Mode Imputation:** For categorical data, using the mode ensures that imputed values are the most likely occurrences, maintaining the distribution.

Disadvantages are:

- **Mean Imputation:** Reduces variability and can distort the true distribution.
- **Mode Imputation:** May not always be suitable if the data has multiple modes so can lead to the true representation not being shown.

Splitting dataset into training and test sets

It's very important before creating our models to split our dataset into both training and test sets, the primary reason for the splitting of our dataset is to evaluate our model performance on unseen data. The training set that will be used to help build our models through the use of cross fold validation, bootstrapping and random sampling. The test set will be used to simulate how our models perform on outside or unseen data.

By keeping a separate test set to use once we're satisfied with the model is to help ensure that the models performance assessment is not biased, as our model might perform exceptionally well on the training set especially if we're overfitting on it but poor on the test set. The use of this test set will help detect if we are overfitting or not on the training set.

We split the dataset into a 75%-25% (Training data- Test Data), this split provides us with a very good balance between having enough data to train the model effectively (75%) and enough data to test and validate our models performance (25%).

With the 75% of the data being used for the training, it provides the model a substantial amount of information to learn from which will be helpful when we test our model on unseen data / future customers as the company requested.

By allocating 25% for testing we ensure that the model's performance is tested on a large set of unseen examples (loan applicants) which will provide a more accurate representation of how our models perform in practice.

Before splitting my data into the training and test sets I used the set seed function, this makes the random processes such as splitting the data reproducible, which means when the code is re-run with the same seed the same training and testing sets will be used. This also allows for consistent evaluation of changes and improvements when building our models. The seed value I set was the last three digits of my student number (R00257**226**) ensured a unique and reproducible split.

Overall, performing this step is absolutely crucial when building a robust and generalizable model. By training on one set and testing on an unseen dataset, we are ensuring that the model(s) make accurate predictions and simulate a real life scenario.

Splitting categorical predictor variables using entropy

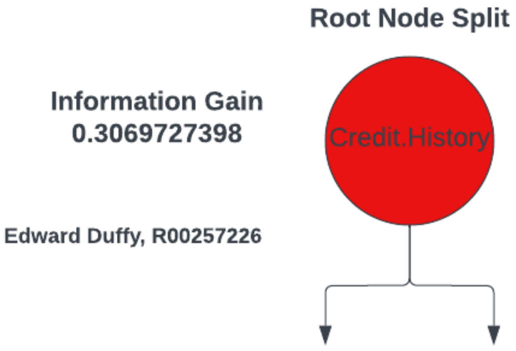
Entropy in terms of machine modelling, is the measure of purity or impurity of a set of examples. It allows for the ability to quantify the uncertainty or randomness within our data. It effectively measures how much disorder we have in our dataset with respect to the target variable.

In decision tree's, the entropy is used to determine how a node should split the data. A high level of entropy means for us that the data in the node is mixed and can't be well separated, while a low entropy means that our data is more homogeneous.

We are using entropy as it will help us with identifying the best splits that will most effectively categorize the data into homogeneous or pure subsets, making the decision tree more effective with classification, which help us to identify if a customer has a 'good' or 'bad' credit standing. By using the decision tree in conjunction with entropy we are reducing the uncertainty at each node allowing us to make clearer distinction between classes in the target variable (Credit Standing).

The best way to select which predictor variable to use as the root split is to target the lowest information gain which is the calculation as to which feature provides us the most information regarding a certain class. Higher information gain for an attribute means its more significant in classifying the data correctly.

When I ran my R code provided in the appendix, the variable with the highest information gain (approx. 0.307) was the 'Credit.History' predictor variable which indicates that it's the best choice for the root node split of the decision tree.



This result implies that 'Credit.History' is the most significant variable in distinguishing between different classes of 'Credit Standing', this was observed earlier when performing our exploratory data analysis of this dataset.

By using this variable as the root node split it will significantly reduce entropy and uncertainty and create subsets that are more homogeneous in terms of 'Credit.Standing' . Splitting on this variable effectively segments the dataset into groups that are more predictive of the outcome and thereby making our decision tree more accurate and efficient in its prediction.

Using Binary Splits:

Binary Splits in a decision tree refer to the dividing of the data at a decision node into exactly two groups based on specific conditions and criteria. Each split within our tree will separate the data into two distinct subsets.

We are applying binary splits because they are easier to understand, implement and result in a tree structure that is easier to interpret while being very efficient in terms of speed and handling larger datasets in the future. By using these binary splits we are again reducing the chances of overfitting on our training data by reducing the complexity of our decision tree.

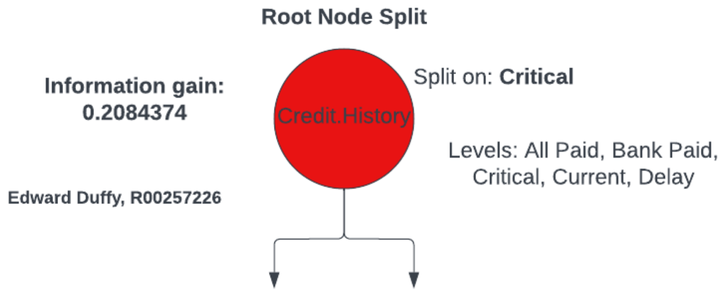
The advantage of using binary splits for this dataset is that its flexible with predictor variable types so it’s ideal for datasets such as this that contain numeric and non-numeric data.

When it comes to our results by using binary splits we are being lead to more specific and targeted divisions of the dataset which can potentially increase the purity of the resulting subsets.

In order to achieve this task I used the code detailed within the appendix that performs the following:

- Calculate the information gain for all possible binary splits of each categorical variable.
- Identifies the split with the highest information gain for each variable and selects the best overall best binary split available.

From running the code we received an information gain of approximately 0.2084 for the split on Credit.History, which suggests that this predictor variable is highly effective when making the distinction between someone having a ‘good’ or ‘bad’ credit history and is a highly effective variable to reduce our entropy to differentiate between the different classes.


Root Node Split
Information gain: 0.2084374
Credit.History
Split on: **Critical**
Levels: All Paid, Bank Paid, Critical, Current, Delay
Edward Duffy, R00257226

The implications of this result are that it will assist in guiding the construction of the decision tree, suggesting that subsequent splits and analysis should be primarily focused on the subsets created by this primary binary split.

In summary, by using binary splits we are offered a balance between simplicity and effectiveness, in the case of applying it to this dataset we identified that ‘Credit.History’ as a crucial variable for the initial decision making process in determining an applicant’s ‘Credit Standing’.

Adding Continuous numeric predictor variables:

It's very important to add continuous variables is because they add comprehensive analysis, increase our predictive power and add a real-world representation.

By adding these continuous variables they can provide vital information that might not be captured by the categorical variables alone. Including them ensures a more comprehensive analysis of our dataset.

As also mentioned it increases our predictive power, by adding these continuous variables that possess a wide range of values they can increase the predictive power of our models by providing them with more granular data.

Many real-world phenomena are continuous in nature and by incorporating these variables within our models they become more reflective of real-world scenarios which is important in the context of this objective as the models we develop later will be used on unseen applicants.

Although adding these continuous variables are very helpful for our models, in the context of binary splits adding these variables can come with their own impacts and challenges such as determining the split points, handling overfitting and being able to balance complexity and interpretability.

One of the main challenges was being able to identify the optimal split point for each of the continuous variables. This required calculating the potential information gain at various points which can become computationally intensive.

Often using continuous variables can lead to overfitting if they aren't managed correctly, as they can create splits that are far too specific to the training data, which is not desirable when we are developing models that are to be used on unseen loan applications.

While these variables are adding depth to our models, they also have the potential to increase its complexity which can have an impact on interpretability of the model.

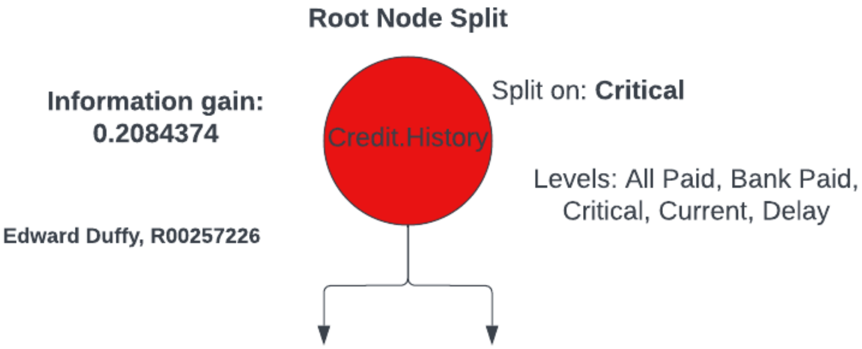
The methodology I used for handling binary splits with continuous variables was to sort and identify the split points, calculate the information gain and select the best split available.

For each continuous variable, the data is sorted and the potential split points are identified as the midpoints between consecutive values.

The information gain for each potential split point was calculated. This involved partitioning the data based on the split point and calculating the weighted entropy of the resulting subsets.

For each variable, the split point that yielded the highest information gain is identified. Then, the best split points for the continuous variables were compared with the best categorical splits in order to determine the best overall split.

From the results obtained the best split identified was again on the ‘Credit.History’ variable, specifically between the ‘Critical’ category and all the other categories, with an information gain of approximately 0.2084374.


Root Node Split
Information gain: 0.2084374
Credit.History
Split on: Critical
Levels: All Paid, Bank Paid, Critical, Current, Delay
Edward Duffy, R00257226

This result is significant as despite considering the continuous variables, ‘Credit.History’ still remains the most significant predictor variable for our initial split, underscoring its importance within our data to determine if an applicant has a ‘good’ or ‘bad’ credit standing.

This split distinctly classified ‘Credit.History’ into ‘Critical’ and ‘non-critical’ groups, indicating a clear and significant division within the dataset relevant to being able to accurately predict the credit standing of a loan applicant.

This result suggests that when building our models it should firstly consider the ‘Credit.History’ predictor variable when making any decisions, as based on our findings has the most substantial impact on reducing uncertainty regarding the credit standing of loan applicants.

An information gain of approximately 0.2084374 is relatively high, implying that splitting on the ‘Credit.History’ predictor variable significantly reduces entropy compared to the other variables, including continuous. It indicates to us that it is the best feature for differentiating between outcomes within our data which makes it an ideal starting point for the decision tree.

The inclusion of these continuous variables in our analysis provided a thorough examination of all the potential predictors. However, the categorical variable ‘Credit.History’ emerged as the most influential, demonstrating that it has a critical role in predicting the credit standing of loan applicants.

Second split of the decision tree:

Investigating the second split of the decision tree is important as it refines our predictions, captures complex relationships and increases our accuracy.

- By analysing the second split we refine the predictions made by the decision tree after the root node split, which is at ‘Credit.History’ on ‘Critical’. By doing this we further segregate our data to improve the homogeneity of the resulting subsets.
- We can capture the more complex relationships that are not apparent after the first split, this leads us to a better understanding of the interactions between our variables which can potentially be used to perform feature engineering to make our models more accurate.
- Iteratively splitting our data it allows for our decision trees to increase the overall accuracy of the models we look to develop.

The information we can gain from the second split:

- Feature Interaction: The second split can provide critical information on how other predictor variables within our data interact with the categories formed by the first split.
- Actionable insights: The insights we gain from these subsequent splits can assist with the tailoring of strategies for different segments identified through these splits.
- Model Complexity: It helps us to determine if adding more complexity (additional splits) to the model is justified by a significant gain in performance.

Challenges finding second split

- Combinational Complexity: As more splits are being considered within the decision tree, the number of potential combinations for binary splits increases, making the problem more computationally complex. This means that this trade-off needs to be considered when adding complexity through more splits.
- Overfitting: As the tree depth will increase so will the risk of overfitting, especially if the dataset is quite noisy.
- Data Sparsity: The subsets we create may become too small after several splits. This can lead to them becoming unreliable.

Using Information from the first split:

- **Subset Creation:** I used the outcome of the first split *‘Credit. History’ – ‘Critical’* to create two subsets of the data, those that fall into the first split and those that do not.
- **Targeted Analysis:** I performed targeted analysis on each of the subsets in order to determine the best second split available, considering only the relevant observations.

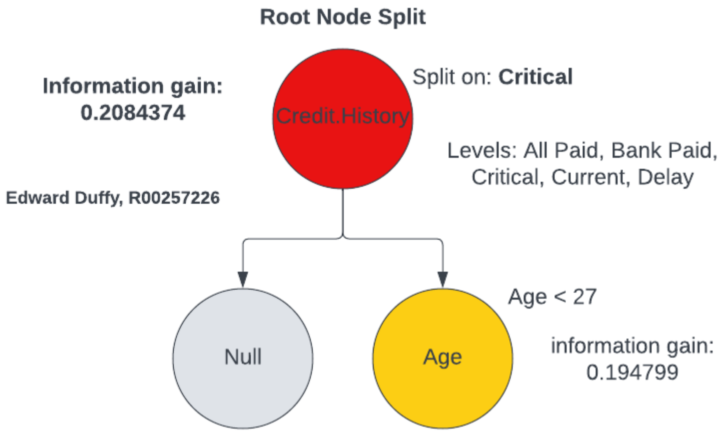
The conceptual challenges with writing the code:

- **Understanding Entropy and Information gain:** I found one of the conceptual challenges was grasping how entropy is a measure of disorder and how the information gain quantifies the reduction in entropy due to a split.
- **Handling the different data types within our dataset:** One of the other challenges was managing the binary splits for both of the categorical and continuous variables. For the categorical variables these were converted into factors which I learnt while debugging my code after running into errors with my code.

Obstacles Overcame:

- **Debugging the NA values:** I implemented checks within my code to handle the NA values that can arise during the calculation of entropy, this is especially prevalent when a subset has a zero variability in the target variable.
- **Ensuring Data Integrity:** I had to make sure that the subsets were non-empty and contain the necessary variables for the splitting calculations.

Results of second split


Root Node Split
Information gain: 0.2084374
Credit.History
Split on: Critical
Levels: All Paid, Bank Paid, Critical, Current, Delay
Edward Duffy, R00257226
Null
Age
Age < 27
information gain: 0.194799

Interpretation of or results:

- **Result of the Critical Subset:** The 'Null' subset in our result indicates that there is no second split that meaningfully separates the classes further. This could be due to homogeneity or insufficient data points.
- **Result of the Non-Critical:** The best split is now on the 'Age' variable with a split point of approximately 27.05 years and an information gain of 0.194799 suggests that 'Age' is a significant predictor of the non-critical subset.

Significance of the Results obtained:

- **Significance of the Age predictor variable:** The fact that 'Age' is identified as the best second split indicates its importance in conjunction with 'credit history' in predicting the credit standing of future loan applicants.
- **The Quality of the information gain:** The information gain of 0.194799 is quite substantial although not very high. It indicates a moderate separation power of the 'Age' variable at the chosen threshold.

In conclusion the decision to split on 'Age' following 'Credit.History' indicates that these two predictor variables in conjunction are significant predictors for the credit standing. The 'Null' result for the critical subset suggests that within this subset, 'Credit.History' might already be a strong predictor, or there isn't enough data to find a meaningful further split. The moderate information gain for the 'Age' split suggests that while the Age predictor variable does contribute to predicting the credit standing of loan applicants, it may not be as strong a predictor as 'Credit.History' by itself.

Why I will not be using the check variable to build our decision tree’s and models

I avoided using the ‘check’ variable as I didn’t want any data leakage occurring which is when information from outside of the training dataset is used to create our models. This happens when a feature inadvertently contains data from the future or even derived from the target variable. The consequences of this can lead to overestimating the performance of our models because the model will have access to information it wouldn’t have in a real life scenario, which would could potentially be the case when if this ‘check’ variable were to be used as per its description below:

Check – *The data analyst created this field as a check on Credit Standing and had access to all historical data.*

Why avoiding ‘Check ‘ was prudent:

- **Historical Data Access:** If ‘Check’ has been derived from all of the historical data, including the target variable ‘Credit.Standing’, It’s likely to be a post-event indicator that would not be available at the time of making real predictions.
- **Model Integrity:** By excluding ‘Check’, we are ensuring the integrity of the model by preventing it from having an unfair advantage during the training process.
- **Real-World Applicability:** The exclusion of the check variable guarantees that the model’s performance metrics are reflective of it’s real-world applicability and are not artificially inflated.

Practices I used:

- **Predictive Modelling:** I decided to only use the data that would be available at the time of prediction to avoid introducing future information into the past.
- **Selecting features when building models:** I carefully selected feature that are causative and not merely correlated of the structure of our data.

Excluding features that could lead to data leakage is crucial when developing our models that accurately reflect its predictive capabilities. It’s very important for the reliability and validity of our models performance and evaluation, ensuring that when the model is deployed in a real world scenario it’s performance is in line with company expectations.

Building our decision tree

It’s very important to build decision tree’s before building our models for the following reasons:

- **Interpretability:** By building and using decision tree’s it provides us with a clear visualization of the decision-making process, and makes it easy to understand how the model arrives at its predictions as to whether or not a loan applicant’s credit standing is good or bad.
- **Feature Importance:** Decision tree’s help to identify the most significant variables that are impacting the outcome, this is indicated by the depth of the variables within our tree.
- **Discovering non-linear relationships:** Decision tree’s also help us to capture non-linear relationships between features and the target variable without requiring any transformation.
- **Segmenting our data:** Theres also the ability for segmentation of our data into homogenous groups which makes it easier to analyze and target for specific interventions.

I also used the decision the tree structure before building our models because they can provide us with information such as:

- **The decision rules being implemented:** Each node in our decision tree will represent a decision rule and the path from root to a leaf within our tree will represent a classification rule.
- **Interactions between the variables:** The structure of the decision provides us with insights as to how predictor variables are interacting with one another.
- **Classification paths being used:** By plotting our decision tree we are provided with a visual representation of the various paths that are taken in order for a decision to be determined.

Importance of using Decision Tree’s before building our models:

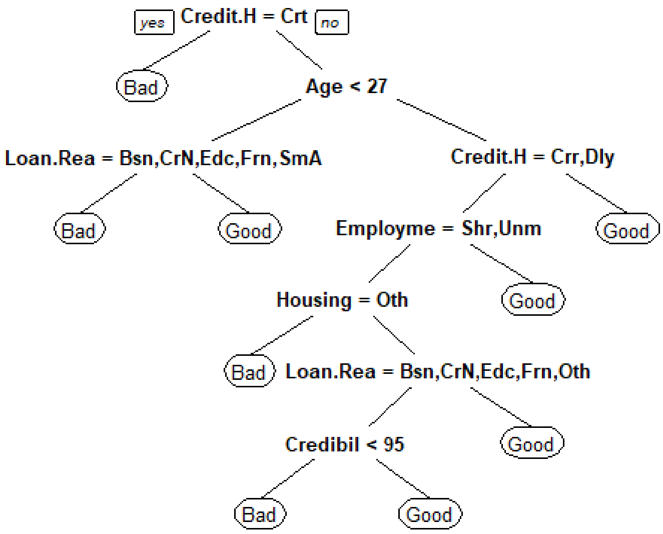
- **Provides a baseline model:** An advantage of using a decision tree is that it provides us with a good baseline model to compare against the models we will be building later on such as the random forest model.
- **Feature selection:** It helps us with feature selection by identifying the relevant variables when trying to predict a loan applicants credit standing.

Methodology Used:

- **Recursive Partitioning:** The rpart function was used to implement recursive partitioning, which involves the splitting of the data based on variable values that result in the most distinct subsets in terms of the credit standing target variable.

- **Predicting Classes:** I used the prediction function to generate predictions from the model which I then used to compare against the actual true values to assess the accuracy.

Decision Tree Plot and Results:



Significance of the result:

- From this decision tree visualization we can see that the results from this and the investigations for the root node and second node split are consistent and the same results are observed.
- We can see that the two significant predictor variables for credit.standing are both the Credit History and Age on Critical and under 27 years of age respectively.
- Using this decision tree can assist the company with key decision making and obtain a better understanding of the more significant variables when assessing or predicting the Credit Standing of a loan applicant.
- The interpretability of this decision tree will be useful for stakeholders to understand the factors influencing credit standing predictions of loan applicants.

Comparison to our previous analysis for root and second node splits:

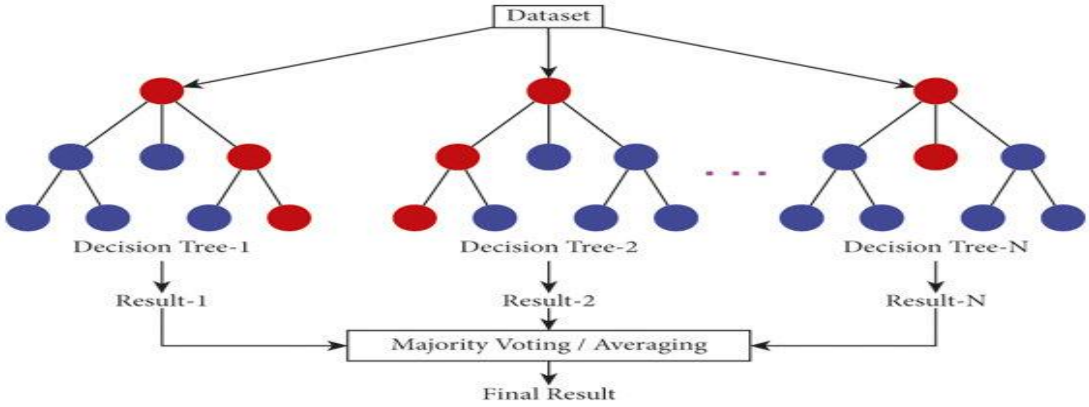
- There is a consistency between the decision tree and our analysis earlier which indicates the robustness of these features for the prediction of credit standing.
- By using the decision tree we are visually representing the binary splits and the sequential order in which these features are being used to predict credit standing, starting with credit history, followed by age. This sequential ordering is a fundamental aspect of our decision tree model with the tree confirming the priority of the splits.

In summary, our decision tree confirms our findings from our earlier binary split analysis, emphasizing significance of credit history and age as primary splitting variables in predicting credit standing.

Building a random forest model:

After building our decision tree and understanding the important predictor variables to determine the credit standing of loan applicants within our data, I built a random forest model using the randomForest library and assessed its performance using the training set through cross fold validation, bootstrapping and random sampling along with the final test set.

The random forest itself is an algorithm that combines predictions from multiple decision tree's to make more accurate and stable predictions with each tree built from a sample drawn with replacement from the training data.



Advantages of using a random forest:

- **Accuracy:** By applying an ensemble of decision trees a higher accuracy can be obtained.
- **Versatility:** They can be used for classification and possess the ability to handle different data types such as the ones within our dataset.
- **Robustness:** Has the ability to handle outliers and non-linear data which in turn adds a higher level of robustness compared to creating an individual decision tree as we did earlier.
- **Parallelizable:** Has the ability to perform parallel processing which can help us speed up the training time.

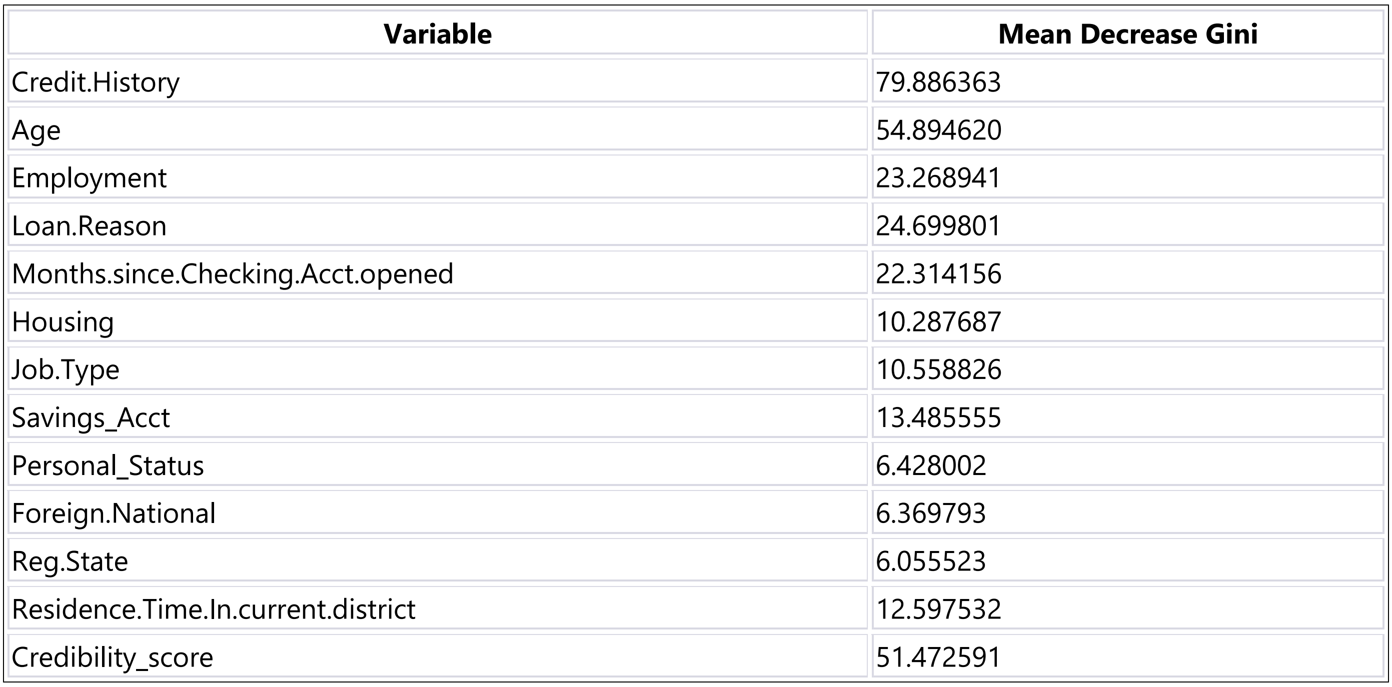
Disadvantages of using a random forest:

- **Interpretability:** While an individual tree as plotted earlier is easy to interpret, but due to a random forest using an ensemble its very hard to interpret due to large amount of tree's being built at any one time.
- **Complexity and Size:** Due to the nature of the decision tree and how it builds a large number of trees at any one time, this comes with a trade off with complexity and high memory consumption.

When I started building my random forest, I used all of the variables initially besides the 'ID' and 'Check' variables as mentioned previous. I did this so that I could use the importance function to find what the most significant variables were for our model. I used this data provided by the importance function to reduce the complexity of my formula for my random forest so I wouldn't be overfitting on the training and could still achieve an accurate score on both of the training and test sets.

CreditStanding=f(CreditHistory,Age,Employment,LoanReason,MonthsSinceCheckingAcctOpened,Housing,JobType,SavingsAcct,PersonalStatus,ForeignNational,RegState,ResidenceTimeInCurrentDistrict,CredibilityScore)

When I assessed the importance of the variables I used in my randomForest using the importance function I got the following results.



Variable	Mean Decrease Gini
Credit.History	79.886363
Age	54.894620
Employment	23.268941
Loan.Reason	24.699801
Months.since.Checking.Acct.opened	22.314156
Housing	10.287687
Job.Type	10.558826
Savings_Acct	13.485555
Personal_Status	6.428002
Foreign.National	6.369793
Reg.State	6.055523
Residence.Time.In.current.district	12.597532
Credibility_score	51.472591

The Mean Decrease Gini is a metric used by the random forest to quantify the importance of each feature in predicting the target variable.

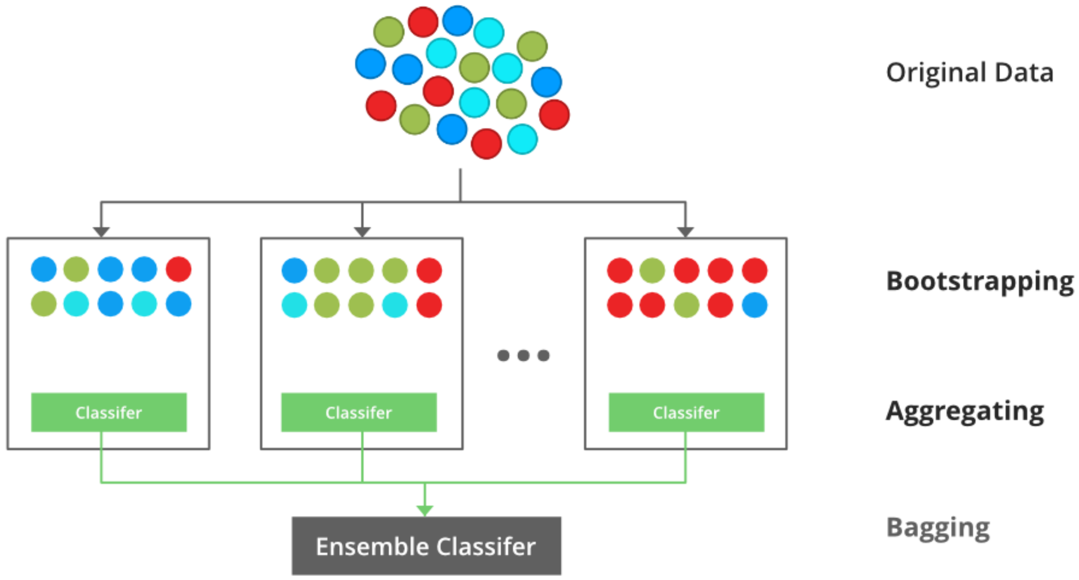
From this output we can see that the variables 'Credit.History', 'Age', 'Credibility_score', 'Loan.Reason' and 'Months.since.checking.acct.opened' are very important predictor variables and were used in a revised formula when building my final random forest model.

CreditStanding=f(CreditHistory,Age,CredibilityScore)

Once I was happy with my model formula I used different techniques using the training data to assess the models performance to simulate how my model worked on unseen data as it would in a real-world scenario, before moving onto the test set using techniques such as bootstrapping, manual Subsets, cross-validation and confusion matrices

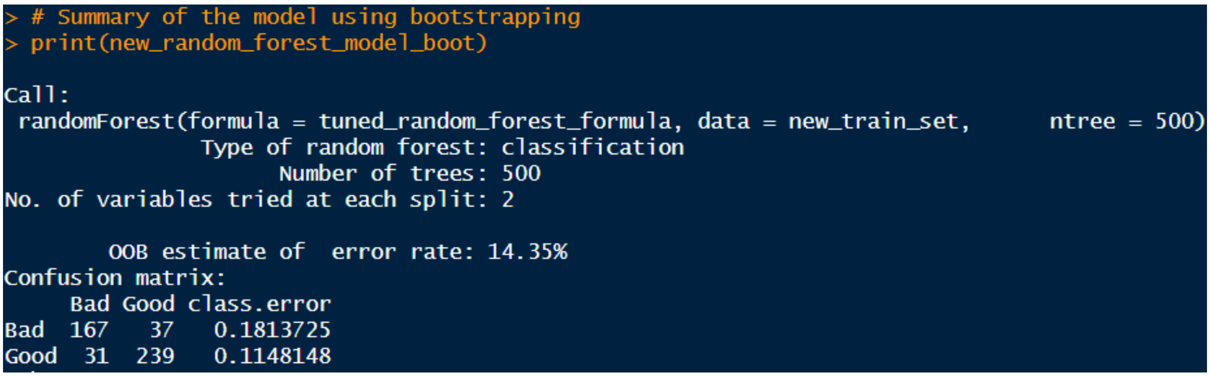
Bootstrapping:

Bootstrapping is a resampling technique that involves repeatedly drawing samples with replacement from the training data and fitting our models to these samples and allows for estimating the variability of the models predictions. I used the bootstrapping technique during the training phase as it has the following advantages:



- **Robustness:** using bootstrapping adds robustness by reducing the variance and providing a better estimate of our models performance.
- **Model Stability:** Assists in assessing the stability of our models by evaluating its performance across different samples.
- **Error Estimation:** Offers us a way to estimate the prediction error without the need for an external validation set.

Results of using Bootstrapping:



```
> # Summary of the model using bootstrapping
> print(new_random_forest_model_boot)

Call:
 randomForest(formula = tuned_random_forest_formula, data = new_train_set,      ntree = 500)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 14.35%
Confusion matrix:
     Bad Good class.error
Bad  167   37   0.1813725
Good  31  239   0.1148148
```

Model Specification:

- The model was trained on 500 trees and at each split in the tree two variables were considered.

Out-of-Bag (OOB) Error Estimate:

- The OOB error rate is 14.35%, this is the estimate of the models error rate when predicting on unseen data. It is calculated by using each tree to predict the data points that were not included in the bootstrap sample for that tree.

Confusion Matrix:

- The confusion matrix shows the model's predictions against the actual values for the two classes 'good' and 'bad' for credit standing.
- For the 'Bad' class there were 167 true positives (Correctly predicted as bad) and 37 false negatives (incorrectly predicted as 'Good').

Class Error:

- 18.14% of the 'bad' instances were misclassified while 11.48% of the 'Good' instances were misclassified.

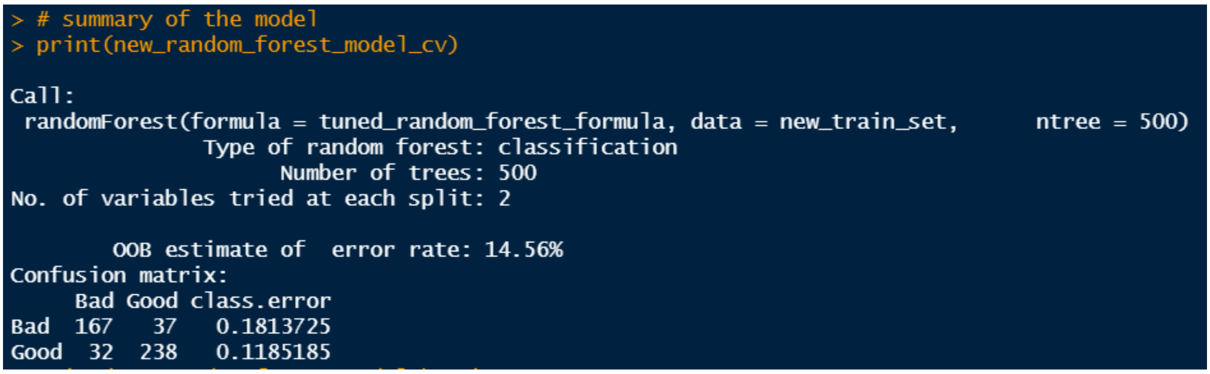
Significance of our bootstrapping result:

- The OOB error rate provides us with an unbiased estimate of the classification error to expect on unseen data given that the new unseen data has a similar distribution to that of the training data.
- The relatively low OOB error rate suggests that the model is performing well and has generalized sufficiently to the underlying relationships within the data.
- The class error rates indicate that the model seems to be better at predicting the 'Good' class than the 'Bad' class. This could be down to class imbalance or the 'Good' class being easily separable based on the features used in our model.

Why Cross-Fold Validation was used and results obtained:

Cross fold validation is the technique of assessing how the results of a statistical analysis will generalize to an independent dataset through the partitioning of data into complementary subsets. I implemented this technique during the training phase due to its advantages such as reduction of overfitting, generalizability and efficient use of data.

Significance of results:



```
> # summary of the model
> print(new_random_forest_model_cv)

Call:
 randomForest(formula = tuned_random_forest_formula, data = new_train_set,      ntree = 500)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 14.56%
Confusion matrix:
     Bad Good class.error
Bad  167   37   0.1813725
Good  32  238   0.1185185
```

The results I obtained from cross fold validation were very similar to that of bootstrapping earlier.

- The OOB rate of 14.56% suggests that there is a reasonably good performance with room for improvement.
- For the confusion matrix we can see that there is slightly better performance in correctly predicting the ‘Good’ class in comparison to the ‘Bad’ class.
- The results demonstrate the model’s strengths and weaknesses in classifying between ‘Good’ and ‘Bad’ credit standings, highlighting areas where the model could require further tuning.

Manual Subsets:

The last technique I implemented was the use of manual subsets which is the process of dividing the data into separate training and validation sets. I used this as the results provide an understanding of how the model performs on a specific curated set. It’s also useful for mimicking real-world situations that can provide s with practical insights on its performance.

Results obtained and their significance:

- **Overall Accuracy:** model has an accuracy of 0.8382, meaning it correctly predicted the credit grading 83.82% of the time. This is significantly better than the No Information Rate of 0.6176, which is the baseline accuracy if one were to predict the most frequent class for all observations.
- **Confidence Interval:** The 95% confidence interval for accuracy is between 0.704 and 0.886, indicating that if the model were applied to different samples from the population, the accuracy would likely fall within this range 95% of the time.
- **Kappa Statistic:** The Kappa value of 0.6567 suggests a substantial agreement between the predicted and actual values. It accounts for the possibility of the agreement occurring by chance.
- **Sensitivity:** Sensitivity, which is the true positive rate, is 0.7821, meaning that the model is relatively good at identifying 'Bad' loans. Specificity is 0.8730, indicating that the model is also good at recognizing 'Good' loans.
- **Predictive values:** Positive Predictive Value (or precision) is 0.7922, showing that when the model predicts 'Bad,' it is correct about 79.22% of the time. The Negative Predictive Value is 0.8661, meaning the model's prediction of 'Good' is correct 86.61% of the time.
- **Prevalence:** The prevalence of the 'Bad' class is 0.3824, indicating that 'Bad' loans constitute approximately 38.24% of the cases in the dataset.
- **Balanced Accuracy:** an average of the sensitivity and specificity, resulting in 0.8275, which gives an idea of the model's overall ability to avoid bias toward either class.

The positive class is ‘Bad’, meaning that the ‘Bad’ loans are being considered the focus of the predictive analysis. The high accuracy and substantial kappa value indicate that the model is performing well above the baseline level with good ability to distinguish between ‘good’ and ‘bad’ credit standings. However there could still be room for improvement with concentration on reducing the number of false positives and negatives to increase accuracy.

Testing my random forest model on the test set and why techniques were used during training:

Once I was satisfied with my model during the training phase I tested my model on the unseen test set. It was important to do the techniques as mentioned previously during the training phase for the following reasons:

- **Model Validation:** The use of these techniques are essential for validating the model in a manner that is independent of the test set, which the goal is to use at the very end of the modelling process. Otherwise we would be fitting our model to training when it should be tuned to work on any unseen applicant data.
- **Hyperparameter Tuning:** These techniques allow for effective hyperparameter tuning without contamination of the test set which could inadvertently lead to overly optimistic performance estimates.

Significance of our result on the test set:

- **High Accuracy:** An accuracy of 88.94% is quite a high result. This indicates that our random forest model is correctly predicting the credit standing of unseen loan applicants for a large majority of the test set.
- **Model Generalization:** The high accuracy on the test set suggests that our model has generalized well from the training data to the unseen test data. This is a strong indication of our models robustness and its capability to make reliable predictions in real-world scenarios.
- **Reliability for deployment:** For a random forest in a credit standing context, a high accuracy rate on the unseen test set indicates the model's reliability in a real-world setting. I would be cautious to deploy this in the real-world as we have only trained it on a limited set of data and it may be very beneficial to train our model on even more training data.

Comparing our results with random forest to that of our decision tree:

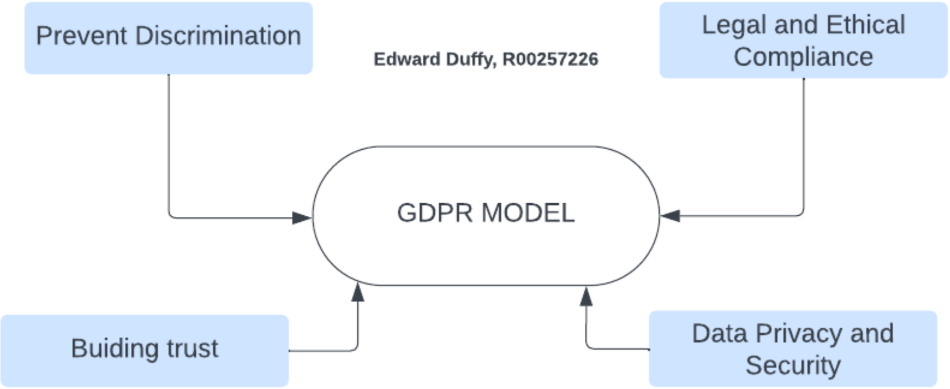
When comparing the results obtained from both the decision tree and the random forest on the test set we can say that:

- The random forest achieved an accuracy of approximately 88.94% on the test set, which is significantly higher than the decision tree's 75.22%.
- The substantial difference in performance can be seen as a clear quantitative indication of the superior predictive power of the random forest model for this dataset.

The inherit advantages of random forest, such as its ensemble learning capabilities and handling overfitting, result in a more accurate and generalizable model when compared to the simple decision tree that we built earlier as reflected in our results on the test set.

Building a GDPR Compliant Model:

As part of our task for this company I had to build a General Data Protection Regulation (GDPR) model that didn't use predictor variables such as Age, Personal Status and Foreign National.



It's crucial that our predictive models are GDPR compliant as a this can have an impact on individuals lives such as in credit scoring. It's very important to build a GDPR compliant model for the following:

- **Prevention of Discrimination:** the aim of building a model that is GDPR compliant is to prevent unfair or discriminatory practices in automated decision making. By restricting the use of sensitive variables such as age, personal status and foreign national, the model is less likely to inadvertently discriminate against individuals based on these attributes. This is crucial in maintaining fairness and ethical standards.
- **Legal and Ethical Compliance:** GDPR compliance is a legal and ethical requirement. Adhering to these regulations would ensure that our model respects individuals' privacy and data rights.
- **Data Privacy and Security:** There is also a strong emphasis on the privacy and security of personal data. By aligning the model development process with GDPR, the risk of data breaches and misuse of personal data is minimised, this protects both the individuals data and the organisation's integrity.
- **Building Trust:** This will also help build trust with the companies loan applicants in the future.

Revising the training and test sets:

Before building my models both the training and the test sets had to be modified to exclude the sensitive variables Age, Personal Status and Foreign National. Once these variables were removed I built a random forest containing all of the variables that remained and ran my model through the importance function in order to analyse the Mean Decreasing Gini of the remaining variables that we had to work with. As age is one of our important predictor variables the next important predictor variable needed to be discovered.

Importance of variables within our new GDPR model:

Variable Category	Variable	Mean Decrease Gini
High Importance	Credit.History	79.628734
	Credibility_score	69.549314
	Employment	24.268289
Moderate Importance	Loan.Reason	24.232994
	Months.since.Checking.Acct.opened	21.358502
	Checking.Acct	12.484563
	Savings_Acct	13.120680
	Residence.Time.In.current.district	12.502108
	Job.Type	10.489516
Lower Importance	Housing	9.571953
	Reg.State	6.394369
Not Meaningful Variables	ID	31.328089

From this importance data I built a new formula for my model using the predictor variables that possessed both a high and moderate importance. The formula I developed for my model was the following:

CreditStanding=f(CreditHistory,CredibilityScore,Employment,LoanReason,MonthsSinceCheckingAcctOpened)

Just as I did with the previous random forest model, I performed techniques during the training stage such as bootstrapping, cross-fold validation, manual subsets to simulate how this model would perform in the real world on unseen data before moving onto the unseen test set.

Analysis of training results:

Cross Fold Validation vs Bootstrapping:

- Both techniques showed the highest accuracy with a mtry of 12.
- Cross Fold Validation yielded slightly higher accuracies compared to that of bootstrapping.
- The results obtained suggest that considering a moderate number of variables at each split delivers the best model performance.

Manual Subsets:

- Using manual subsets delivered an accuracy of 77.94% which was lower compared to Cross fold validation and bootstrapping.
- This difference in accuracy could be down to the specific distribution of the data in the manual subsets or due to the model's variability in performance across different data segments.

Overall Assessment:

- The results during the training phase were good which indicates a robustness with our model's ability to generalize across different validation methods.
- The accuracies obtained are in the high 70%-80% range. This could be seen as a good or bad result depending on what the model is being used for in a real-world scenario. As this is being used to determine the credit standing of customers which is quite sensitive it would need to be determined by a domain expert.

Results on the Test Data:

- Once I was satisfied with my results from the training phase using the different techniques as discussed earlier, I proceeded to test my model on the unseen test set.
- The accuracy on the test data is 75.66% which means ¾ of the predictions the model makes are accurate. As age was quite a powerful predictor variable in our earlier models this could explain why the test accuracy is lower than previous.

Comparing random forest model and GDPR Compliant Model:

Cross-Validation results:

- **GDPR Model:** Showed an accuracy of 84.39%
- **Previous Model:** Performed best with 'mtry = 12' with an accuracy of 82.16%
- **Observation:** The GDPR model demonstrated a slightly higher accuracy in cross-validation compared to the previous random forest model.

Bootstrapping:

- **GDPR Model:** It's best accuracy was on 'mtry = 12' with 81.71%
- **Previous Model:** The accuracy was very similar to the GDPR model with 81.59%
- **Observation:** Both models performance in bootstrapping were similar, with a marginal improvement in the GDPR model.

Manual Subsets:

- **GDPR Model:** Achieved an accuracy of 83.82%
- **Previous Model:** Also chieved an accuracy of 83.82%
- **Observation:** Identical performance in manual subset validation.

Test Data Results:

- **GDPR Model:** The accuracy on the test data was 77.43%.
- **Previous Model:** A slightly lower accuracy on the test data of 75.66%.
- **Observation:** The GDPR model slightly outperforms the previous model on the test set. This is good as this would be the model that would have to be deployed in the real-world, if the previous model was better at predictions than the GDPR compliant than we would have an issue.

Analysis of the Differences:

- **Impact of removing variables:** The removal of sensitive variables such as age, personal status and foreign national in the GDPR model would have had an expected effect of reducing the model's performance, especially considering the high importance of the age variable as highlighted earlier. However, the results show that the GDPR model performs comparably and even slightly better in some aspects than our previous model.
- **Robustness:** The results obtained indicate that the GDPR random forest model is robust to the exclusion of these sensitive variables, maintaining good performance even with a reduced feature set.

Conclusion on findings:

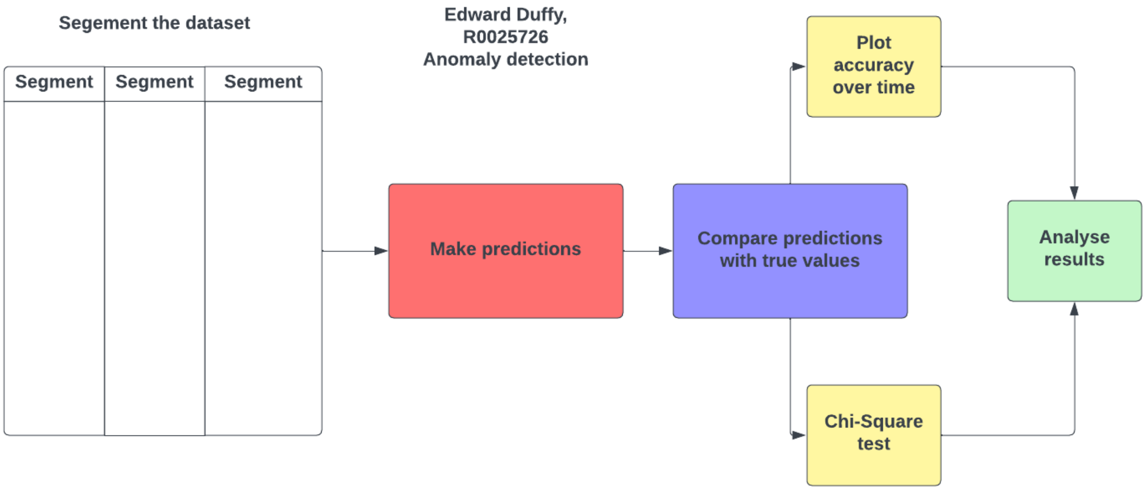
The comparison between the two models shows that the GDPR-compliant model maintains strong predictive power despite exclusion of sensitive variables. The slight improvement could be attributed to the models ability to generalize better with simplified features. This analysis underpins importance of feature selection and robustness.

Developing a strategy to find patterns of suspicious gradings:

Approach used and methodology:

Data Preparation:

- **Combining Data:** I merged the ID numbers (used as timestamps), actual grades and predicted grades into a single dataset labelled as 'combined data'.
- **Verification:** I then proceeded to check the structure of the combined dataset.



Segmentation and Accuracy Analysis:

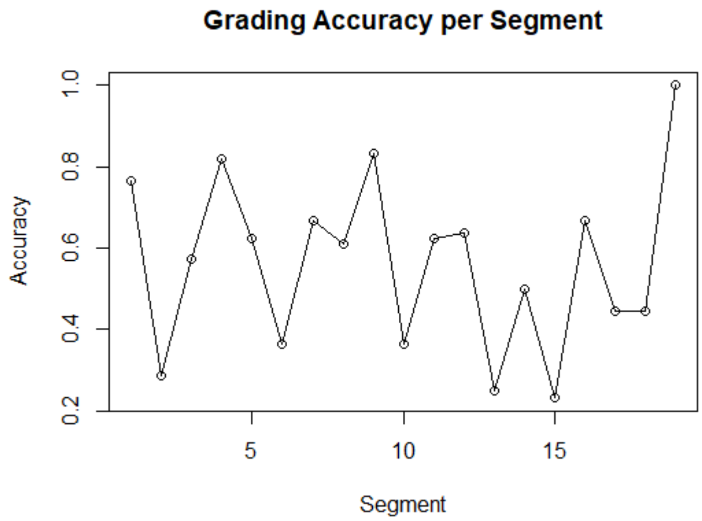
- **Defined Segments:** I divided the data into segments based on the ID numbers, with each segment encompassing 50 ID's.
- **Accuracy Calculation:** For each of the segments I calculated the accuracy as the proportion of matching actual predicted grades.
- **Visual Analysis:** I then proceeded to plot these accuracies to visually inspect any segments with abnormal accuracy levels, which could indicate periods were inconsistent grading occurs.

Statistical Testing:

- **Creating Observed Table:** I proceeded to build a contingency table labelled 'observed' showing the count of correct and incorrect gradings per segment.
- **Chi-squared Test:** I then performed a chi-squared test on this table to statistically assess the distribution of grading accuracies across the segments. By using this test it helps identify segments where grading accuracy significantly deviates from the expected norm.

Analysis of Grading Accuracy per Segment:

This plot of the grading accuracy per segment shows the variation in grading accuracy across each of the segments. From plot we can identify any segments with unusual accuracy which could point to periods of inconsistent grading.



Observations from the plot:

Variability in Accuracy:

- This plot shows us that there is significant variability in accuracy across different segments. The accuracy seems to fluctuate greatly from one segment to another, with some segments having an accuracy close to 1.0 and others much lower than that of around 0.2.

Identification of outliers:

- Particular segments stand out to us with notably low accuracy, which may suggest to us that there are periods where the grading process is less reliable. These segments are potential areas of concern that Siobhán and the company are suspicious of which could warrant a further investigation.

Trends and Patterns:

- There's no trend over time and accuracy spikes are irregular

Further analysis:

- **Next steps:** From this plot, the next step would be to examine the segments with the lowest accuracy more closely. This would involve the process of looking at specific ID's within these segments to better understand if there's any commonality among them, such as being graded during the same time period or by the same individual in the company.
- **Data Context:** When considering the context of this data, if certain segments correspond to periods of change within the company this could explain the inconsistencies in grading.
- **Further Statistical Testing:** Additionally statistical tests could be conducted on the segments identified as outliers to ascertain the likelihood that their accuracy rates are due to chance or a systemic issues.

Conclusions we can make:

- From this visual representation of grading accuracy per segment, it has been highlighted that there are areas the correspond to Siobhán and the companies suspicions of poor performance in the loan grading process. By performing this exploratory analysis it has provided the foundation for a deeper analysis as to what is causing these inconsistencies.

Chi-Squared Test and Results:

What is Chi-Squared Test:

- The Chi-Squared is a statistical test used to determine if there is any significant differences between the expected frequencies and the observed frequencies in one or more categories of a contingency table. It helps us to determine whether the distribution of correct and incorrect loan gradings across different segments is as expected or if there are any anomalies that could indicate periods of grading inconsistency.

Why it was helpful to use it for this problem:

- **Helps Identify inconsistencies:** If certain segments have a significantly different number of correct/incorrect loan gradings than expected, it would indicate periods where the grading system was not performing as expected.
- **Quantifying evidence:** Chi-Squared provides us with a p-value to quantify the evidence against the null hypothesis. A low p-value suggests that the observed distribution of grading accuracy is not likely due to random chance, which could indicate potential issues in the loan grading process.
- **Guiding Investigation:** Highlights segments that significantly deviate from expectations, the test can guide further investigation into what might be causing these inconsistencies in loan gradings such as either system or human error.

Chi-Squared Test Results:

- **Statistical Significance:** The chi-squared test resulted in a p-value of 0.0002951, indicating a statistically significant difference in grading accuracy across segments.
- **Observed vs Expected Frequencies:** The comparisons of observed counts with expected frequencies under a random distribution suggests certain segments had significantly different grading accuracy.
- **Standardized Residuals:** These values are highlighting specific segments where the difference between observed and expected frequencies is pronounced.

Retrospective analysis on the approach we used:

- **Effectiveness of the approach used:** The combined use of visual analysis and statistical testing is effective in identifying periods with potential grading issues. The segmentation based on the ID numbers allows for a time based analysis on the grading accuracy.
- **Strengths:** This method I used offers a comprehensive view, where visual plots aid in the quick detection of anomalies, and the chi-squared test provides statistical validation of these observations.

Limitations:

- **Segment Size:** One of the limitations of the approach I used was the segment size as this can have an influence on the results. Smaller segments might be sensitive to minor fluctuations, while larger segments could mask short-term inconsistencies.
- **Assumption of independence:** The chi-squared test has a limitation in that it assumes independence between the observations, which might not be true in a time-series context like ID's.

Conclusion on this approach to detect anomalies in gradings:

The methodology I used to try to solve this problem I feel is suited for identifying periods of grading inconsistencies. It allows for the detection of both broad trends and specific segments where grading may not align with the overall pattern. However, the interpretation of these findings would have to consider the context of the grading system and any external factors that might have an influence grading accuracy during specific periods.

Overall, the solution I developed for the problem the company was having, provides both a detailed and holistic view of grading accuracy over time.

Conclusion:

From this report and our findings we can see that there are many factors that influence a loan applicants credit standing

When we analysed the dataset further before building our model to ensure its integrity we found many issues such as:

- Consistency of categorical values
- Anomalies with the employment of applicants
- Missing values within our data
- Outliers
- Negative values.

Having these issues within our dataset we were correct to question the integrity of the data, through the use of data cleaning and imputation some of these present issues were resolved without changing the original dataset significantly which is crucial when building our models off this raw data from the company.

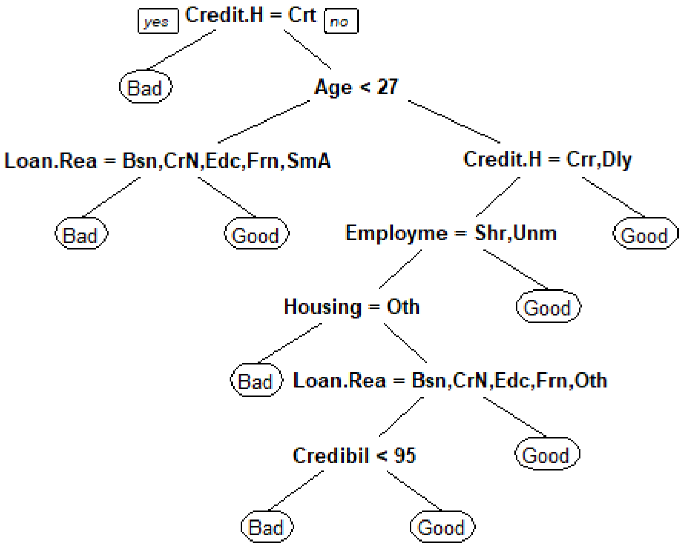
Upon further analysis we saw that variables such as ‘check’ and ‘ID’ could not be used in our model building due to:

- Check being derived from all historical data so it was likely to be a post event indicator that would not be available for making real predictions and would be a cause of data leakage.
- Preventing unfair advantage during the training phase and real-world applicability.
- For the case of the ‘ID’ variable we couldn’t use it due to its discriminatory nature, we can’t decide if a person is of ‘good’ or ‘bad’ credit just based on their ID alone.

Once we split into our training and test models we came to the following conclusions when investigating the root node split for categorical and numeric values and the second node split:

- Categoric: We found that the best predictor for credit standing with an information gain of and was the best root node predictor variable was Credit History, as we saw in our analysis of the relationship it is the best predictor of an applicant’s credit history.
- Numeric + categoric: Credit history was again the best predictor of an applicant with an information gain of and a split on ‘Critical’
- 2nd node split: We found that ‘Age’ was the 2nd best predictor of age with a split of age < 27 with an information gain of 0.194799.

In order to further investigate and add more confidence to our results when investigating the root and second node splits we got a visualization of our findings which were correct as can be seen in the below plot.



When we tested this model on the test set an accuracy result of 75.22% was obtained, this needed to be improved through the use of random forest. To ensure this through the use of the importance() function and techniques such as bootstrapping, cross-fold validation, manual subsets along with confusion matrix we improved our model on the test set and achieved an accuracy of 88.93%.

While this was a good result we needed to build another model that was compliant with GPDR which meant we couldn't use any of the sensitive features including our second best predictor of credit standing 'Age'. I proceeded to use the same training techniques as I did with the random forest. Through the use of these methods and tuning the parameters used to avoid overfitting an accuracy of 77.43%. I tried to improve this through the use of feature engineering but without domain knowledge I couldn't make any assumptions on the data myself.

The final task we needed to develop a model that detected anomalies with credit score gradings, through the use of segmentation, chi-square tests a method was developed where we could track the grading performance of loans over time by using the real values and comparing the predicted values, from this we saw that the gradings trend was irregular and there were anomalies with the gradings which were either systemic or down to human error.

From this report we have investigated thoroughly the different factors that are needed to determine an applicants credit standing and developed models that were accurate but not in my opinion accurate enough to use in a real setting compliant with GDPR, for this more stronger predictor variables will need to be found along with more intense training techniques.

References

Random Forest Illustration:

- ResearchGate. (n.d.). Illustration of random forest trees [Figure]. Retrieved from https://www.researchgate.net/figure/Illustration-of-random-forest-trees_fig4_354354484

Bootstrapping Illustration:

- Tangellamudi, H. (n.d.). Bootstrapped Aggregation (Bagging). Retrieved from https://medium.com/@hemaanushatangellamudi/bootstrapped-aggregation-bagging-481f4812e3ea

Confusion Matrix:

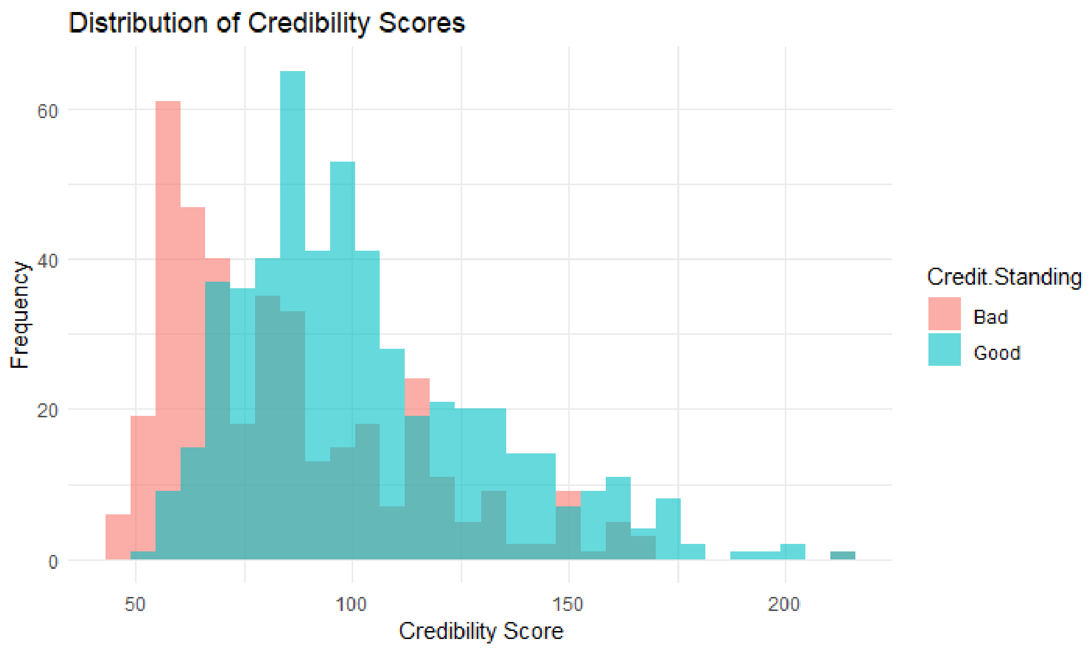
- Tangellamudi, H. (n.d.). Bootstrapped Aggregation (Bagging) [Confusion Matrix illustration]. Retrieved from https://medium.com/@hemaanushatangellamudi/bootstrapped-aggregation-bagging-481f4812e3ea

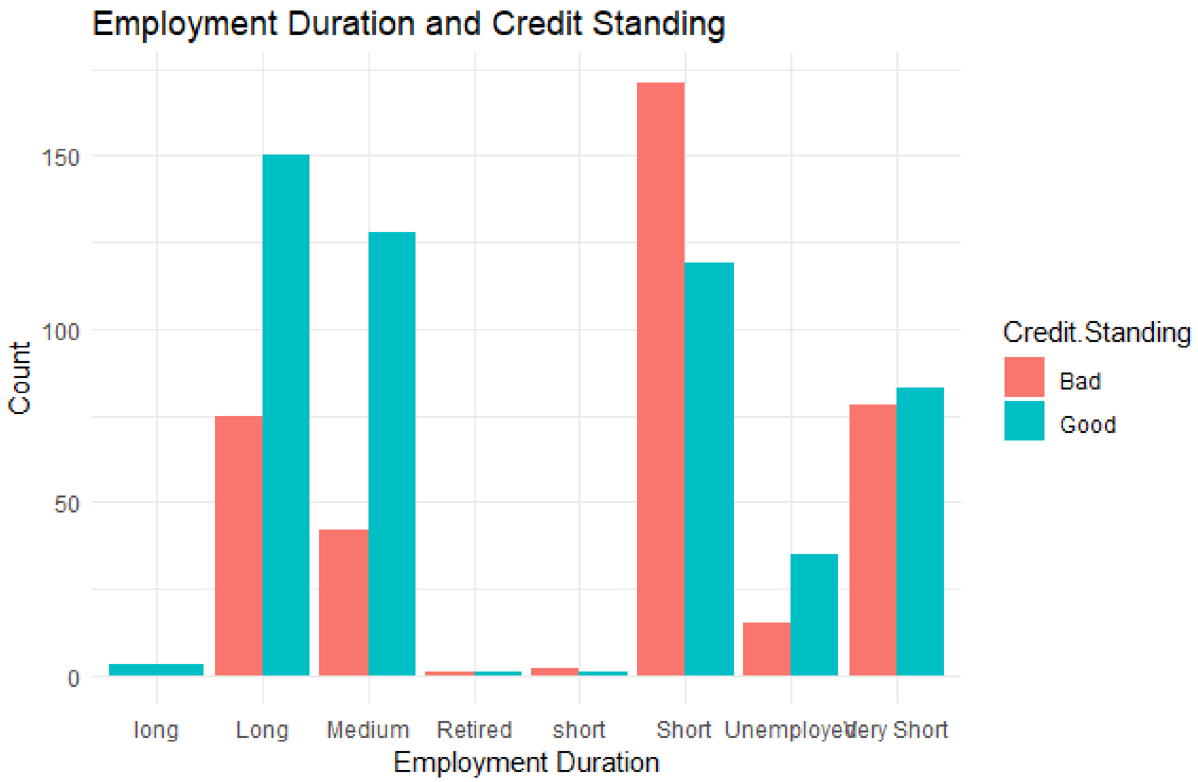
Lucidchart for Constructing Diagrams:

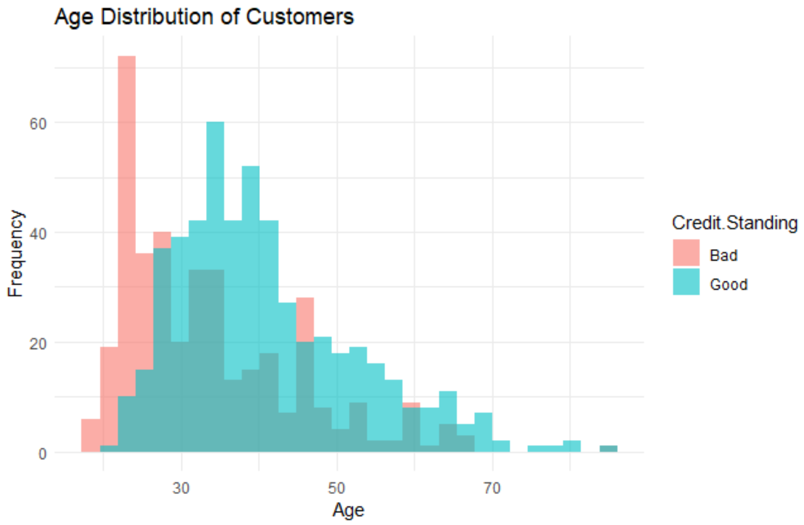
- Lucidchart. (n.d.). Retrieved from https://www.lucidchart.com

Appendix:

1. A: Unused plots






Age Distribution of Customers
Frequency
0
20
40
60
30
50
70
Age
Credit.Standing
Bad
Good