Credit Risk Scoring Coursework

Shaun Robert Commee

C22085988

Contents

Critically Examine what needs to be Considered when Developing a Credit Risk Scoring Model. (Part A)

A credit risk scoring model assesses the likelihood a borrower will default on a debt obligation. Since the 2008 Financial Crash, a great onus has been placed on understanding and managing this risk effectively. This factor coupled with the vast expansion of available data and investment placed into manipulating it, has changed the landscape of how financial institutions assign and understand risk. This, in turn, means channels of credit are being scrutinized in a manner we have never seen before. This scrutiny now predominantly comes in the form of credit risk scoring models. Such a model assesses the risk associated with credit applications based on a vast variety of parameters and statistical techniques. Here we will assess the various considerations required to generate such a model.

The first consideration relates to the framework implemented. The CRISP-DM process is the leading methodology for data mining process models and implements 6 phases in order. Business understanding, Data understanding, Data preparation, Modelling, Evaluation and Deployment. (Altair, 2022) It provides clear project structure from start to finish which push teams to consistently monitor and review business requirements and the data pipeline to ensure business needs are satisfied. A misuse of this process could damage the reliability and effectiveness of any prospective model. Whilst proper implementation will ensure that answers are met for the models' hypotheses, allow for the replication of results and enable stakeholders to collaborate effectively.

The backbone of any model is the data which it uses. The quantity, quality and usefulness of that data is an essential consideration to ensure a relevant and accurate output. It is important to firstly understand the limitations based on potential variables due to the Data Protection Act 2008. These include constraints on reviewing gender, race, religious beliefs and ethnic background etc. Beyond this, the sample size that is obtained must be representative and large enough to be of statistical significance to ensure the model is effective. The importance of data quality can be clearly demonstrated through the regulations imposed by Basel II & Basel III. Banks are required to evaluate the risk associated with default on a loan and retain buffer capital to anticipate future unexpected losses against this. Inaccurate parameters caused by poor data quality will lead to an unreliable evaluation to the probability of default and therefore lead to losses and possibly even bankruptcy of the institution. In a survey conducted amongst 37 financial institutions, 63% of respondents indicated that inconsistency (value and format) and diversity of data sources are the main recurring challenges in Data Quality. Furthermore, they estimate that between 10-20% of the data in their databases is subject to errors. (Helen-Tadesse Mogesa, 2011) These errors and inconsistencies are ultimately inputted into any credit-risk scoring model and run the risk of making decisions based on incorrect evidence. Stakeholders therefore need to consider minimising the loss of data quality by reducing human error, handling outliers and missing values, ensuring data is up to date and running frequent validation sets to ensure the models relevance. (Deloitte, 2016)

It is also vital there are well-defined business requirements to tailor the CRISP-DM process around. A banking institution must carefully consider its own objectives with consideration to tolerance for risk, profit-loss objective, efficiency and recovery process costs for example. (Sabato, 2010)These factors are part of what is used to set-out the aims and objectives of the model requirements. This is essential to ensure that all parties are working in tandem so that business needs are clearly understood to ensure Data Scientists tailor the model appropriately to business requirements. A misunderstanding between business objectives and the model can greatly impact the effectiveness of the model, costing the business time and money.

Binning is a vital component in model development as it divides values of a continuous variable into groups which provide the greatest predictive power. The process can undertake numerous methodologies including

equal-width binning, equal-size binning, and optima binning for example. It is important before binning takes place that missing values are binned separately, each bin should contain at least 5% of observations and that no bins have zero accounts of good or bad outcomes. (Zeng, 2014) A very common approach to binning in credit risk is using Weight of Evidence as a method to obtaining the most statistically powerful approach. Using this method the user will seek to find the bins which provide the greater Information Value and, in turn, enhancing the models predictive capabilities.(Plug & Score, n.d.)

The final consideration that should be given is to which variables will determine an applicant to be deemed 'good' or 'bad'. This consideration can be broken into three parts. The first revolves around the attributes of the customer applying for credit, there are 100's of variables that are typically used in scorecards. To fully understand these requires a high degree of business knowledge and thorough analysis to measure the weight that should be attached to each variable within the model. Which variables should be omitted or included is a vital consideration to the overall performance of the model and having the neccessary domain knowledge will prove very useful to undertake this process. Secondly, the financial institutions objectives heavily impact upon how it wishes to view risk, as there are great potential financial rewards for banks which undertake riskier lending practises. Considerations therefore need to be taken to maximise profit whilst mitigating risk, and this invariably affects the volume of those applicants a bank wishes to lend to. Lastly, models must also attain an understanding of the impact of business cycles upon societies attitude to risk. There are stark differences for the environment for credit dependent upon economic, political and natural impacts. These factors must be realised as it heavily impacts upon the capital requirements calculation which banks use to limit their financial risk.

Explain how, in theory, Cox's Proportional Hazard Model for Survival Analysis can be used for Constructing a Scorecard. Comment on the Relative Popularity of Cox's PH Model versus Logistic Regression in Scorecard Construction.(Part 2A)

Logistic Regression takes explanatory variables and seeks to predict the impact upon a categorical predictor variable by using boolean outcomes. It does this by calculating the impact of an explanatory variable on binary outcomes of 0 or 1, or in non-binary terms, true or false. With respect to Credit Risk Scoring, this enables a lender to determine whether an applicant is suitable for credit given a set of parameters. The model is built by understanding which risk factors are most associated with being 'good' or 'bad' and uses this to predict the outcomes for future applicants.

Survival analysis was firstly introduced in a Credit Risk Scoring context in 1992 as an alternative to Logistic Regression. (Narain, 1992) Cox's Proportional Hazards Model takes multiple predictor variables and seeks to understand the impact each predictor has on the hazard rate. This type of analysis is typically used in medical science but has applications in Credit Risk Scoring, as we seek to understand what impact variables have on the time it takes for an applicant to default on a debt obligation. The model assigns ratios to variables to account for the impact each has on the time it takes an individual to default. This information is then used to generate a scorecard to predict the risk of default.

There has been debate surrounding which statistical method is most applicable to constructing scorecards. Both models exhibit positives and negatives which we seek to discuss here. Firstly, Cox's model is specifically designed to understand the probability of an event occurring within a specific time period. This, by nature, provides lenders with a much greater understanding of prospective lenders. By contrast, logistic regression provides binary outcomes to default within a fixed time period. This means the analysis in many

respects is more limited in scope. A prospective applicant who represents a group of characteristics who's modelled to default several years into a loan, is potentially treated in the same manner as somebody who defaults within the first several months. The omission of time within scorecard construction using logistic regression treats potential customers very differently. This becomes even more prominent given the context of profit. A financial institution that understands the length of time it takes for an individual to default can create a cost-benefit analysis between costs of default and the interest and fees that can be accrued. A typically 'bad' customer defined under logistic regression, is potentially a very profitable customer under a COX's PH Model and more likely to be considered 'good'. (Stepanova, 2001)

The notion of competing risks in Credit Risk Scoring describes multiple possible events which impact upon 'failure'. A lender may view failure in many different manners. For example, early repayment of a loan leading to loss of interest income, default and transfers of debt obligations all represent negative outcomes and are competing risks to a bank's profitability. Cox's PH Model allows for these competing risks to be incorporated within the model and can therefore provide a more relevant and accurate output. The model assumes the consumer is only immune to a subset of risks. This notion is converse to the methodology applied within logistic regression which assumes the consumer is immune to all risks. (Nailong Zhang, 2019) There are however several assumptions that must be satisfied for Cox's PH Model to provide reliable results. One of those is that all hazard ratios must be proportional. (Bellera, 2010) Found that only one in 64 medical journals had verified this assumption and this in turn questions the validity of the outcomes of these studies as it leads to biased outcomes. The assumption implies that the hazard ratio between groups is constant during any point in time. In the case of credit risk, we generally exhibit that as the duration of the loan increases the less likely the probability of default. It is assumed that the predictors are time-independent, and this is rarely the case.

This issue surrounding accuracy of results is extended in the form of censoring. COX's PH Model looks to understand when an event has taken place. If during the observation period an individual does not default they are then censored and not considered within the model. This individual no longer makes up part of the analysis despite their characteristics potentially being of interest. In the case of Logistic Regression, binary outcomes are only provided on loans that have been completed or failed. Those consumers who have ongoing loans are censored from the data and therefore leads to unrepresentative data samples. We observe that ongoing loans (so long as they fall within the observation period) are maintained within the COX's PH Model and arguably provide a more representative account of the consumer base. (Nailong Zhang, 2019)

In summary, we can say that determining which statistical technique to implement depends on the available data alongside the desired outcome. COX's PH Model provides lenders with a more detailed context on their potential and existing customer base. The context of time to default enables a financial institution to undertake a much more comprehensive cost-benefit analysis to lending. The model further incorporates the notion of competing risks which evaluates multiple possible events influencing 'failure'. This increases the depth of the analysis and leads to deeper understanding. Logistic regression on the other hand, is more commonly used given that the interpretation is much simpler than hazard ratios presented in COX's PH Model, easier to explain to non-technical stakeholders, there are less violation of assumptions, no issues surrounding censoring and scorecards generally seek to predict the probability of an event occurring across a fixed time period. Overall, COX's PH Model can be considered the preferred approach given that time-to-event data is available, banks wish to understand the probability of an event occurring in the future and stakeholders have a technical understanding.

A Lender Would Like to Extend its Ability to Offer Credit to Those With Lower Credit Scores and is Considering Doing This Through a Combination of Risk-Based Pricing and the use of More and Different Data in its Credit Scoring Model. Discuss the Implications of These for the Lender in Terms of Both Credit Scorecard Development and Potential Impact on Existing Customers. (Part 3A)

Risk based pricing is an approach to lending in which interest rates are dictated by the risk a borrower presents. The model penalises those individuals who have a low credit score with higher interest rates to counter-act the probability of default and potential associated costs, conversely, it rewards those with higher credit scores with lower interest rates. The outcome of scorecard models is influenced by the data that it utilises, therefore, the introduction of both risk-based pricing and new data to a model will have an influential impact. We will delve into the impact of these factors here.

A model is only as accurate and reliable as the quality and implementation of the data that it uses. A lender that seeks to introduce different and more data must consider several implications. The quality, complexity and relevance of the data will be paramount in determining its impact in reducing information asymmetry. An increase in the volume of data and its type within a model can hamper data quality if not analysed, understood and retrained appropriately. (Helen Tadesse Moges, 2012) This therefore requires careful analysis before introduction to an existing scorecard model. The implications for decreasing the quality of data within a model is substantial for the lender and existing customers by influencing the efficacy of the binary outcomes produced from scorecards. This affects existing customers through the misappropriation of risk and alteration of interest rates upon loan facilities. The impact could then be extended by impacting profit through misplaced risk-based pricing and, increasing the banks' exposure to risk by failing to effectively understand its existing customers risk profiles.

Financial institutions and existing customers can benefit greatly from the implementation of new data into existing models. The more accurate the assignment of risk to existing customers the more effective the scorecard becomes. The introduction of further relevant and quality data sources improves risk assignment. This therefore increases the accuracy at which they assign existing customers as "good" or "bad". As a result, this empowers lenders to feel increasingly confident in offering competitive rates to its existing "good" customer base. This leads to lowering the cost associated with loan facilities. Whilst lenders experience a fall in their risk portfolio through reducing false positives in the model, which, therefore raises the profitability of their lending practises.

The implementation of both risk-based pricing and new data can alter a bank's lending outlook. The ability to tailor a financial facility to the customers risk profile reduces exposure for the bank and therefore, in theory increases its willingness to engage further with potential and existing customers. The lender benefits through increased profitability through risk premiums paid by riskier borrowers, consumers benefit through an increase in available financial products and competitive pricing - rewarding those who act in a "good" manner with lower costs. (Centre for Capital Markets, 2021)

The theoretical basis for risk-based pricing works on the assumption that cost of financial products relate directly to the associated risk. It has been contested that despite the vast increase in availability of data to financial institutions, information asymmetry is still heavily present, and, this factor coupled with the predatory actions of lenders negatively impacts existing and potential customers. (White, 2004) The beforementioned increase in availability comes with stipulations in terms of cost for some in the market. 'Risky' individuals who accessed credit under uniform pricing were financially better off than under a risk-based

pricing model. It can be stated that access has been therefore reduced for some consumers as they are more likely to be priced out from loan facilities or default from ones they obtain.

With respect to the lender, risk-based pricing threatens to de-stabilise its portfolio. With a tool to mitigate risk-associated costs, banks are incentivised to lend to riskier customers who pay large premiums for borrowing to push for profit maximisation. This practise raises the proportion of possible delinquencies and default within a bank's portfolio. It also impacts upon existing customers, as a higher proportion of fixed pools of capital is distributed to riskier new and existing borrowers. This affects the volume of available funds to "good" existing customers leaving them worst off under a model which utilises risk-based pricing. The combination of profit-maximisation analysis under risk-based pricing and the existence of information asymmetry creates risky lending practises. This risk would be extended further through poor data quality (enhancing information asymmetry) therefore demonstrating the importance of intensely scrutinising all data which enters a credit-risk scoring model.

In summary, a lender which extends finance to individuals with lower credit scores through risk-based pricing and an enlargement of its data sources has many considerations. It has the capacity to be a positive force for a lender and, potential and existing customers but caution is required. In a positive manner, if scrutinised and analysed appropriately, an increase in data in the model reduces information asymmetry through extending understanding of potential and existing customers. This gives financial institutions the confidence they understand their customers and their creditworthiness, leading to increased lending and profit through risk-premiums. This has a profound impact on the availability of financial products to potential and existing customers but fundamentally changes the banks portfolio. The lucrative risk-premiums paid by customers pushes banks to profit-maximise through riskier lending practises, pushing lenders away from 'good' customers. These decisions increase the proportion of potential defaults and delinquencies, which, coupled with information asymmetry and unreliable data can leave a lender heavily exposed to risk. Despite the potential benefits in terms of profitability and outreach, failure for a bank to properly monitor data quality and, recognise the limitations of risk-based pricing threatens the banks security through riskier lending practises. It also disadvantages its existing customer base by placing more resources into riskier lending and away from 'good' customers to maximise profit.

Establishing a Scorecard using German Credit Data (Part B)

The dataset contains 1000 entries and 20 variables. Amongst these 20 variables we have 18 categorical variables and 2 numerical. There are also two binary variables within the data which classify whether an applicant is "Good" or "Bad". We will use these variables to be produce a scorecard and interpret our results. We have used a variety of packages in order to achieve this alongside the R Programming Language.

Question 1 - Splitting the Data-Set

In order to answer this question we firstly cleaned the dataset by reviewing if there were any missing values before placing them into subsets. I had noticed 12 missing values within the "Purpose" column and omitted them from the data-frame. I then went on to use the subset function in order to classify the data into the relevant checking scores required.

After sub-setting the values, it was important to check that the subsets lined up with the now remaining rows in the data set. Given the omission of 12 rows from the original 1000 the subsets should have tallied up to 988. Given that Subset1 (532) + Subset2 (456) equaled this amount we are therefore happy to proceed. We

can further state that both subsets have a suitable split between them which will allow for reliable models to be generated.

```
## [1] 532
```

```
## [1] 456
```

```
##
##   0   1
## 235 297
```

```
##
##   0   1
##  60 396
```

Question 2 - Establishing the Training and Validation Sets

A) Principles

Using R we can create a Training and Validation Set for both Subset1 and Subset2 by using the head() and tail() function. Given that the data-set is randomly generated already, we can take the top 70% of the rows (using the head function) and allocate this to our training set and the bottom 30% (using our tails function and assign this to our validation set.

```
train1 <- head(Subset1, round(nrow(Subset1) * 0.7))
validation1 <- tail(Subset1, round(nrow(Subset1) * 0.3))

train2 <- head(Subset2, round(nrow(Subset2) * 0.7))
validation2 <- tail(Subset2, round(nrow(Subset2) * 0.3))
```

Reviewing the splitting process, we can evaluate whether the total split values between good and bad equal the values produced in the subset. In Subset1 we observed 543 observations whilst Subset2 had 456 observations. In training set for Subset2 we observed 41 bads and 278 goods. This, in addition, to the validation sets 19 'bads' and 118 'goods' equating to 456 observations. The same logic was applied to Subset1 to ensure that the data was split appropriately.

```
#Subset1
table(Subset1$Good) # 297 Good / 235 Bad
```

```
##
##   0   1
## 235 297
```

```
table(train1$Good) #163 Good / 209 Bad
```

```
##
##   0   1
## 163 209
```

```
table(validation1$Good) #72 good / 88 Bad
```

```
##
##  0  1
## 72 88
```

```
#Subset 2
table(Subset2$Good) # 396 Good / 60 Bad
```

```
##
##   0   1
##  60 396
```

```
table(train2$Good) #278 Good / 41 Bad
```

```
##
##   0   1
##  41 278
```

```
table(validation2$Good) #118 good / 19 Bad
```

```
##
##   0   1
##  19 118
```

B) Why Both Training and Validation Sets are Needed

We seek to establish a training and validation set in order to compare a models effectiveness against an already determined dataset and outcome. If, we can ascertain that the model is effective we can then use this to model future outcomes. In order to do this, we split the data-frame between Training and Validation. We take most of the data to train the model with, in our case we will take 70%, and, then take it to use against the remaining 30% to compare the results we obtain. During the training process, estimates are made of the variables significance to determine 'good' or 'bad' customers. Taking this information, we can use it against the remaining 30% to compare the reliability of the results for the model.

Question 3 - Variable Selection & Binning Processing

In order to establish which variables are suitable to build a scorecard we can undertake several statistical techniques. It is vital that we narrow down the number of variables or it will reduce its predictive power. In order to establish which variables are suitable we have used Information Value (IV) which seeks to determine the predictive power of variables. It does this by measuring the association between a categorical variable and the response variables. We have used the R package "Information" and its create_infotables() function to provide us with IV values for all of the explanatory variables. The higher the IV score the more predictive power the variable has.

```
##     Variable            IV
## 2   Duration 0.29930256439
## 3    History 0.24927248097
## 6    Savings 0.17479257289
## 13       Age 0.16264046395
## 12  Property 0.15687630110
## 4    Purpose 0.10513025916
## 8   Installp 0.09647408264
## 7    Emploed 0.08340760778
## 9    marital 0.08141873365
## 20   Foreign 0.07178086248
## 5     Amount 0.05472056069
## 15   housing 0.03544007063
## 10     Coapp 0.03047933379
## 11  Resident 0.02873611287
## 16   Existcr 0.01081336560
## 1   Checking 0.01050866482
## 17       Job 0.00998938762
## 18   Depends 0.00736180055
## 14     Other 0.00600315244
## 19 Telephone 0.00006016324
## 21       Bad 0.00000000000
```

```
##     Variable            IV
## 14     Other 0.51390818135
## 4    Purpose 0.42030450290
## 2   Duration 0.39201739017
## 13       Age 0.35441253059
## 5     Amount 0.33938936739
## 7    Emploed 0.28713000551
## 3    History 0.15147954407
## 11  Resident 0.10761922018
## 9    marital 0.08978693811
## 1   Checking 0.08227305988
## 8   Installp 0.07960217457
## 17       Job 0.06994467024
## 10     Coapp 0.04013869582
```

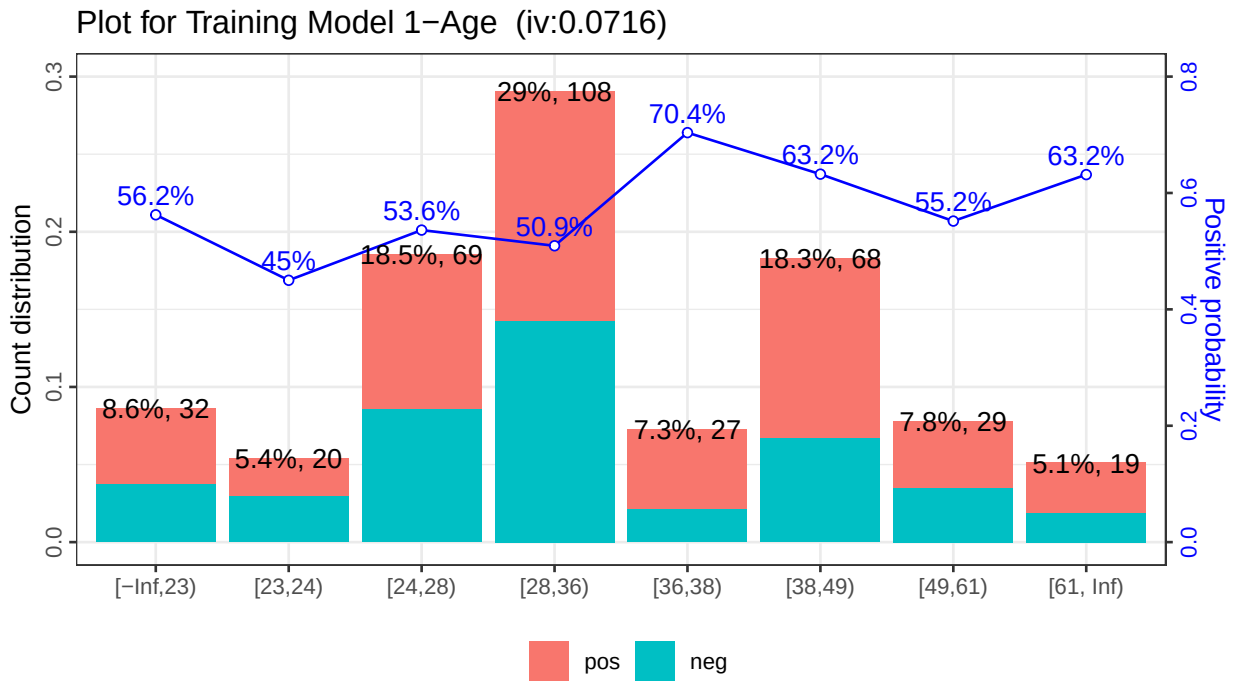
```
## 18   Depends 0.03604060087
## 16   Existcr 0.01952393982
## 6    Savings 0.01453262453
## 12  Property 0.01190890173
## 20   Foreign 0.00190466784
## 19 Telephone 0.00109931109
## 15   housing 0.00004864657
## 21       Bad 0.00000000000
```

We have been requested to extract at least one continuous variable and a categorical variable that has more than two categories. Given the IV values and the specification provided, we have selected Duration, History, Age and Savings for Model 1. It is stated in literature that any values between 0.1 and 0.3 can be said to have a 'medium' degree of predictive power.Extending this practise by using Subset 2 which contains customers with Checking=3 & Checking = 4 we have generated the results provided. Given the criteria requested we move forward with variables Age, Amount, Duration and Purpose. Given the information values we have obtained literature suggests that we have a 'strong' degree of predictive power with the values we have obtained. Applying the same logic to subset 2, using the woebin function we wished to bin variables Age, Duration and Amount by Weight of Evidence.

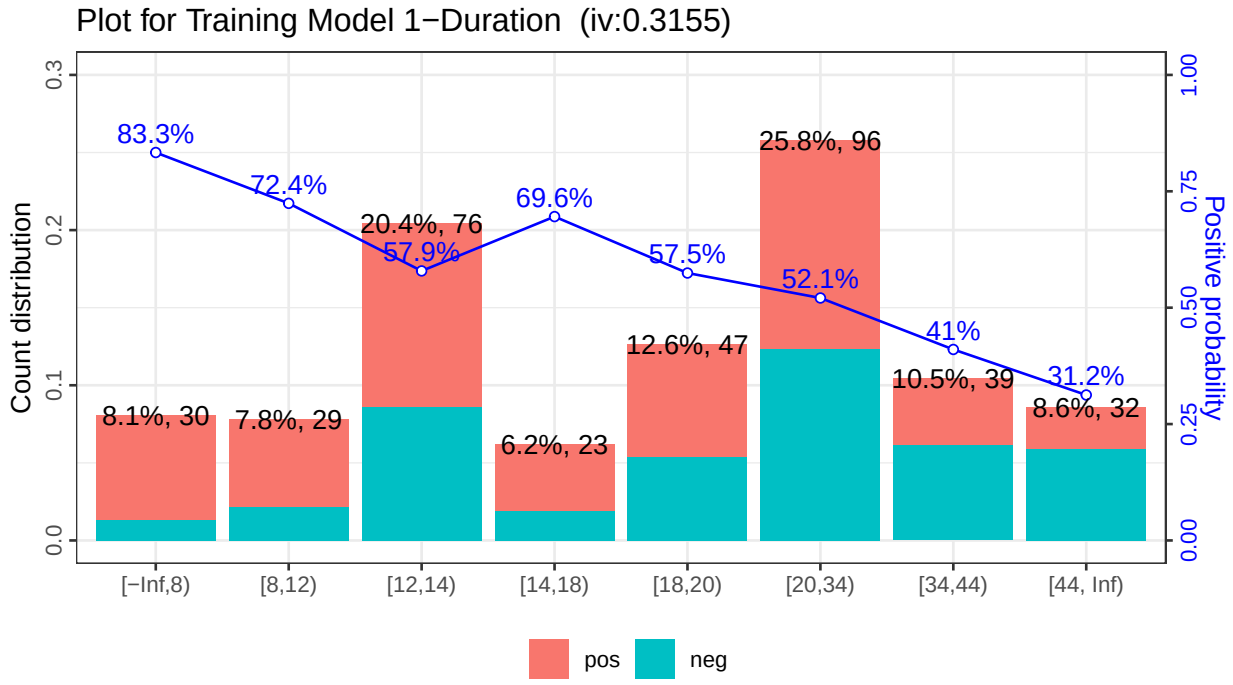
As we wish to understand the explanatory power of group characteristics within the data-set we can bin the variables using the function woebin() from the package "scorecard". We are then able to implement the provided bins into our data-set by using the woebin_ply() function. For subset1 we looked to bin variables "Age" and "Duration" by Weight of Evidence which, in turn, looks for the relevant bins which provide the greatest explanatory power with respect to "Good" applicants. This allows us to be able to categorise groups of people to provide a comprehensive analysis of the behaviour with respect to risk of default. We must apply the same logic to the validation sets to ensure that identical formatting is maintained for use of our models later on.

Below we can observe plots that have been generated from using the woebin_plot() function on our training set. We can see graphically the weight of evidence associated with each individual bin associated with the altered variable. It further demonstrates the number of observations and the percentage of observations that fall within each bin. The line plot highlights the total weight of evidence for that particular binned variable.

```
## $Age
```




```
##
## $Duration
```



Below we can observe the results and the splitting of the data for Training Model 1 as an example to the processes we have taken already.This highlights the bins which have been assigned, alongside their associated Weight of Evidence values. Furthermore, we can observe the number of counts in each bin and the relative distribution of those counts within each bin.

```
## $Age
##    variable       bin count count_distr neg pos   posprob          woe
## 1:      Age [-Inf,23)    32  0.08602151  14  18 0.5625000  0.002730377
## 2:      Age   [23,24)    20  0.05376344  11   9 0.4500000 -0.449254747
## 3:      Age   [24,28)    69  0.18548387  32  37 0.5362319 -0.103402041
## 4:      Age   [28,36)   108  0.29032258  53  55 0.5092593 -0.211542779
## 5:      Age   [36,38)    27  0.07258065   8  19 0.7037037  0.616413386
## 6:      Age   [38,49)    68  0.18279570  25  43 0.6323529  0.293740240
## 7:      Age   [49,61)    29  0.07795699  13  16 0.5517241 -0.040944686
## 8:      Age [61, Inf)    19  0.05107527   7  12 0.6315789  0.290412450
##             bin_iv   total_iv breaks is_special_values
## 1: 0.0000006411782 0.07158168     23             FALSE
## 2: 0.0109719068068 0.07158168     24             FALSE
## 3: 0.0019941627130 0.07158168     28             FALSE
## 4: 0.0131146960477 0.07158168     36             FALSE
## 5: 0.0257841628414 0.07158168     38             FALSE
## 6: 0.0153824107660 0.07158168     49             FALSE
## 7: 0.0001310056892 0.07158168     61             FALSE
## 8: 0.0042026987302 0.07158168    Inf             FALSE
##
## $Duration
##    variable       bin count count_distr neg pos   posprob         woe
## 1: Duration  [-Inf,8)    30  0.08064516   5  25 0.8333333  1.36085386
## 2: Duration    [8,12)    29  0.07795699   8  21 0.7241379  0.71649684
## 3: Duration   [12,14)    76  0.20430108  32  44 0.5789474  0.06986968
## 4: Duration   [14,18)    23  0.06182796   7  16 0.6956522  0.57809452
## 5: Duration   [18,20)    47  0.12634409  20  27 0.5744681  0.05152054
## 6: Duration   [20,34)    96  0.25806452  46  50 0.5208333 -0.16520244
## 7: Duration   [34,44)    39  0.10483871  23  16 0.4102564 -0.61148954
## 8: Duration [44, Inf)    32  0.08602151  22  10 0.3125000 -1.03704141
##           bin_iv  total_iv breaks is_special_values
## 1: 0.1210375789 0.3155416      8             FALSE
## 2: 0.0368270166 0.3155416     12             FALSE
## 3: 0.0009926593 0.3155416     14             FALSE
## 4: 0.0194298948 0.3155416     18             FALSE
## 5: 0.0003342249 0.3155416     20             FALSE
## 6: 0.0070994327 0.3155416     34             FALSE
## 7: 0.0394712041 0.3155416     44             FALSE
## 8: 0.0903495732 0.3155416    Inf             FALSE
```

Question 4 - Scorecard Generation

Taking the data-set that we have now created with the appropriate bins and maximised weight of evidence, we can now train the respective models. Taking Model 1 and applying logistic and linear regression using "Good" as the dependent variable we seek to train the model to understand which characteristics of the selected variables a "Good" applicant has. Taking these trained models we then create a scorecard using the scorecard() function. This assigns points to each predictor variable depending upon their contribution to the "Good" outcome using the regression coefficients. We can then apply this scorecard to the data-set by using the scorecard_ply() function to show scorecard scores given the model we have created. We can see from the figure provided below a scorecard demonstration for the linear regression for Training Model 1. Each bin has been assigned points dependent upon their relationship with the dependent variable "Good". This can then be used to generate an overall score and determine the creditworthiness of an individual. We are also able to observe the scores that have been assigned to each applicant within Training Model 1. I will move forward using a regression model which was trained directly on the bins provided as opposed to Weight of Evidence values. I have demonstrated the scorecard function for completeness with different regression models to highlight its point assignment. I will not be using this particular model going forward as we require the process to be assigned by bins as opposed to weight of evidence for ROC Curves and appropriate training. (Scorecard function takes Weight of Evidence values and converts them to bins and assigns points based on its impact upon the provided predictor variable)

```
#Training Model 1 - Logistic Regression
logmodel1bins <- glm(formula = Good ~ Duration_bin + History + Savings + Age_bin,
family="binomial", data = train1bins) #Scorecard Model going forward
logmodel1 <- glm(formula = Good ~ Duration_woe + History + Savings + Age_woe,
family="binomial", data = trainwoe) #Model to use in Scorecard analysis
card1 <- scorecard(bins, logmodel1) #Scorecard generation
my_scores1 <- scorecard_ply(creditdata, card1) #Show Scorecard scores

#Training Model 1 - Linear Regression
linearmodel1bins <- lm(formula = Good ~ Duration_bin + History + Savings
+ Age_bin, data = train1bins) #Scorecard Model going forward
linearmodel1 <- lm(formula = Good ~ Duration_woe + History + Savings + Age_woe,
data = trainwoe) #Model to use in scorecard analysis
card2 <- scorecard(bins, linearmodel1) #Create Scorecard
my_scores2 <- scorecard_ply(creditdata, card2)#Show Scorecard scores

#Training Model 2 - Logistic Regression
logmodel2bins <- glm(formula = Good ~ Duration_bin + History + Savings
+ Age_bin,family="binomial", data = train2bins) #Scorecard Model going forward
logmodel2 <- glm(formula = Good ~ Duration_woe + Amount_woe + Purpose + Age_woe,
family = "binomial", data = train2woe) # Model to use in scorecard analysis
card3 <- scorecard(bins2, logmodel2) #Create Scorecard
my_scores3 <- scorecard_ply(creditdata, card3) #Show Scorecard scores

#Training Model 2 - Linear Regression
linearmodel2bins <- lm(formula = Good ~ Duration_bin + History + Savings
+ Age_bin,data = train2bins) #Scorecard Model going forward
```

```
linearmodel2 <- lm(formula = Good ~ Duration_woe + Amount_woe + Purpose
+ Age_woe, data = train2woe) #model used in scorecard analysis
card4 <- scorecard(bins2, linearmodel2) #Create Scorecard
my_scores4 <- scorecard_ply(creditdata, card4)#Show Scorecard scores
```

```
## $basepoints
##      variable bin woe points
## 1: basepoints  NA  NA    466
##
## $Duration
##    variable       bin count count_distr neg pos   posprob         woe
## 1: Duration  [-Inf,8)    30  0.08064516   5  25 0.8333333  1.36085386
## 2: Duration    [8,12)    29  0.07795699   8  21 0.7241379  0.71649684
## 3: Duration   [12,14)    76  0.20430108  32  44 0.5789474  0.06986968
## 4: Duration   [14,18)    23  0.06182796   7  16 0.6956522  0.57809452
## 5: Duration   [18,20)    47  0.12634409  20  27 0.5744681  0.05152054
## 6: Duration   [20,34)    96  0.25806452  46  50 0.5208333 -0.16520244
## 7: Duration   [34,44)    39  0.10483871  23  16 0.4102564 -0.61148954
## 8: Duration [44, Inf)    32  0.08602151  22  10 0.3125000 -1.03704141
##          bin_iv  total_iv breaks is_special_values points
## 1: 0.1210375789 0.3155416      8             FALSE    -90
## 2: 0.0368270166 0.3155416     12             FALSE    -47
## 3: 0.0009926593 0.3155416     14             FALSE     -5
## 4: 0.0194298948 0.3155416     18             FALSE    -38
## 5: 0.0003342249 0.3155416     20             FALSE     -3
## 6: 0.0070994327 0.3155416     34             FALSE     11
## 7: 0.0394712041 0.3155416     44             FALSE     40
## 8: 0.0903495732 0.3155416    Inf             FALSE     68
##
## $History
## Empty data.table (0 rows and 13 cols): variable,bin,count,count_distr,neg,pos...
##
## $Savings
## Empty data.table (0 rows and 13 cols): variable,bin,count,count_distr,neg,pos...
##
## $Age
##    variable       bin count count_distr neg pos   posprob          woe
## 1:      Age [-Inf,23)    32  0.08602151  14  18 0.5625000  0.002730377
## 2:      Age   [23,24)    20  0.05376344  11   9 0.4500000 -0.449254747
## 3:      Age   [24,28)    69  0.18548387  32  37 0.5362319 -0.103402041
## 4:      Age   [28,36)   108  0.29032258  53  55 0.5092593 -0.211542779
## 5:      Age   [36,38)    27  0.07258065   8  19 0.7037037  0.616413386
## 6:      Age   [38,49)    68  0.18279570  25  43 0.6323529  0.293740240
## 7:      Age   [49,61)    29  0.07795699  13  16 0.5517241 -0.040944686
## 8:      Age [61, Inf)    19  0.05107527   7  12 0.6315789  0.290412450
##             bin_iv   total_iv breaks is_special_values points
## 1: 0.0000006411782 0.07158168     23             FALSE      0
```

```
## 2: 0.0109719068068 0.07158168     24             FALSE     25
## 3: 0.0019941627130 0.07158168     28             FALSE      6
## 4: 0.0131146960477 0.07158168     36             FALSE     12
## 5: 0.0257841628414 0.07158168     38             FALSE    -34
## 6: 0.0153824107660 0.07158168     49             FALSE    -16
## 7: 0.0001310056892 0.07158168     61             FALSE      2
## 8: 0.0042026987302 0.07158168    Inf             FALSE    -16
```

```
##       score
##   1:   360
##   2:   534
##   3:   463
##   4:   490
##   5:   479
##  ---
## 984:   473
## 985:   461
## 986:   445
## 987:   559
## 988:   540
```

Question 5 - Deriving ROC Curves and using Gini Coefficient Alongside Kolmogorov-Smirnov Statistic to Test Efficacy of Models

A ROC Curve uses cumulative distribution functions to compare and evaluate True Positive Rates (A prediction of "Good" that in fact turned out to be "Good") against False Positive Rates (A prediction of "Good" that in fact turned out to be "Bad"). This relationship can therefore demonstrate the accuracy of a model through comparisons between sensitivity (True Positives / True Positives + False Negatives) and specificity (True Negative / False Positive + True Negative).

With respect to Credit Risk Scoring, we can therefore determine how effective a scorecard is at correctly determining "Good" customers using this methodology. Taking the ROC curve we can then use Gini and KS test to mathematically evaluate this by looking at the area under the curve and the divergence of the CDF's of "Good" and "Bad". The Kolmogorov-Statistic takes the maximum absolute difference in the CDF's of the Good and Bad outcomes. The greater the KS-score the better the predictive capabilities of the model as the divergence increases between the CDF's. It is suggested in (Siana Halim, 2014) that values that KS-values that fall between 28-35 demonstrate average degrees of separation, 35-45 show high degrees of separation and 45+ shows a very high-quality application scorecard. With respect to the Gini coefficient, this takes values between 0 and 1 and evaluates the area between the Lorenz curve and the line of equality. (the point at which the model has no predictive accuracy) The greater the area under the curve the more predictive accuracy a model shows in discriminating between "Good" and "Bad" outcomes.

Using the scorecard models that we created we now are going to use these on the validation sets that we created before. This will allow us to review the predictive accuracy of these models through the ROC curves. To do this, we use the predict() function which takes our model and applies it to the validation set and makes predictions on the dependent variable of the model, in this case, whether an applicant is "Good" or not. We then use the prediction() function from the pROC package to compare these predictions against the actual

outcome in the validation sets. Taking the performance() function we can measure the True Positive Rate against the False Positive Rate. Plotting such results creates a ROC curve and enables evaluation of the performance of the model.

When comparing the multiple models that we have created we can observe that Linear Regression and Logistic Regression models used for Model 1, which took customers with checking attributes of 1 and 2, performed similarly. We observed a Gini Coefficient of 0.46 and KS-Score of 0.33 for linear regression and a 0.46 Gini score alongside a KS-Score of 0.33 for Logistic Regression. Both models when reviewing the literature demonstrate above average degrees of separation and therefore model accuracy. (Bee Wah Yap, 2011) The second model contained individuals who obtained checking attributes of 3 and 4. For the Linear Regression model we observed a Gini Coefficient of 0.19 and KS-score of 0.19. For Logistic Regression, we observed a Gini of 0.21 and KS-score of 0.21 as well.

Resulting from this, predictions made using linear and logistic regression for model 1 performed above average given the KS scores and high Gini scores amongst both models. We observed weak predictive accuracy for linear and logistic regression for model 2 and we could therefore state that these could not be well relied upon for discriminating between "Good" and "Bad" applicants.

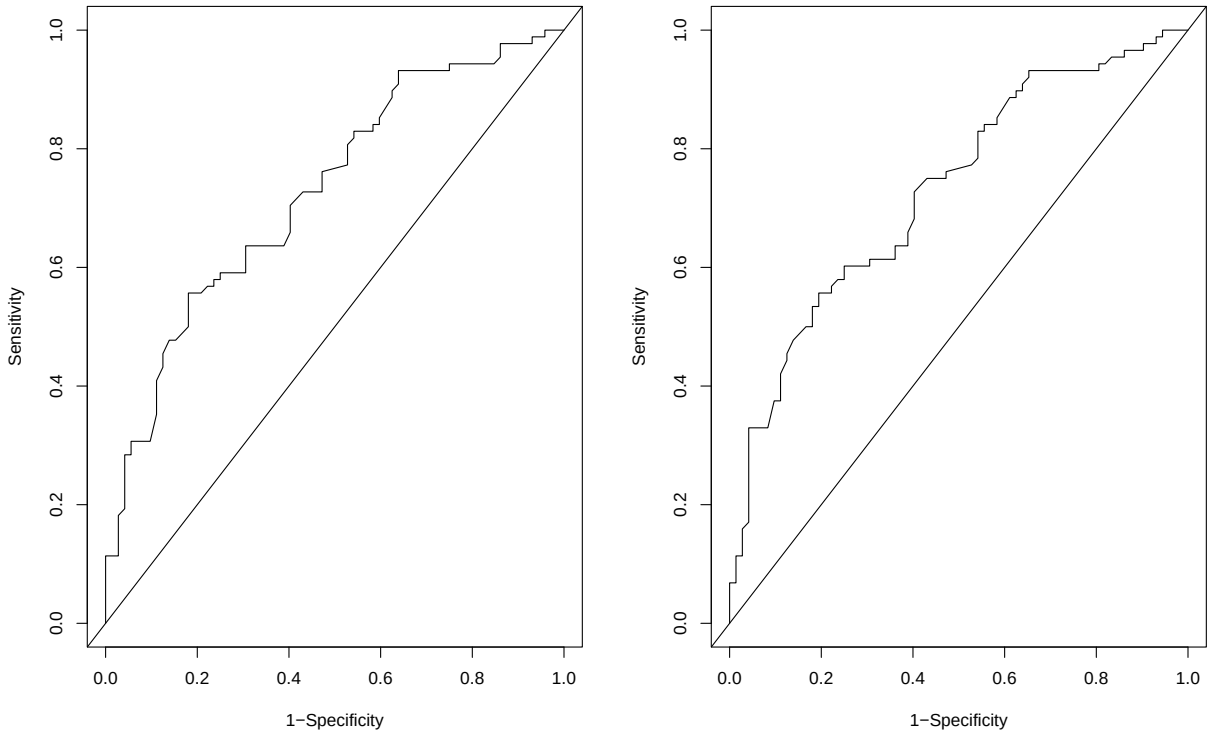


Figure 1: ROC Curves for Linear (left) and Logistic Regression (right) for Applicants with Checking Values 1 and 2

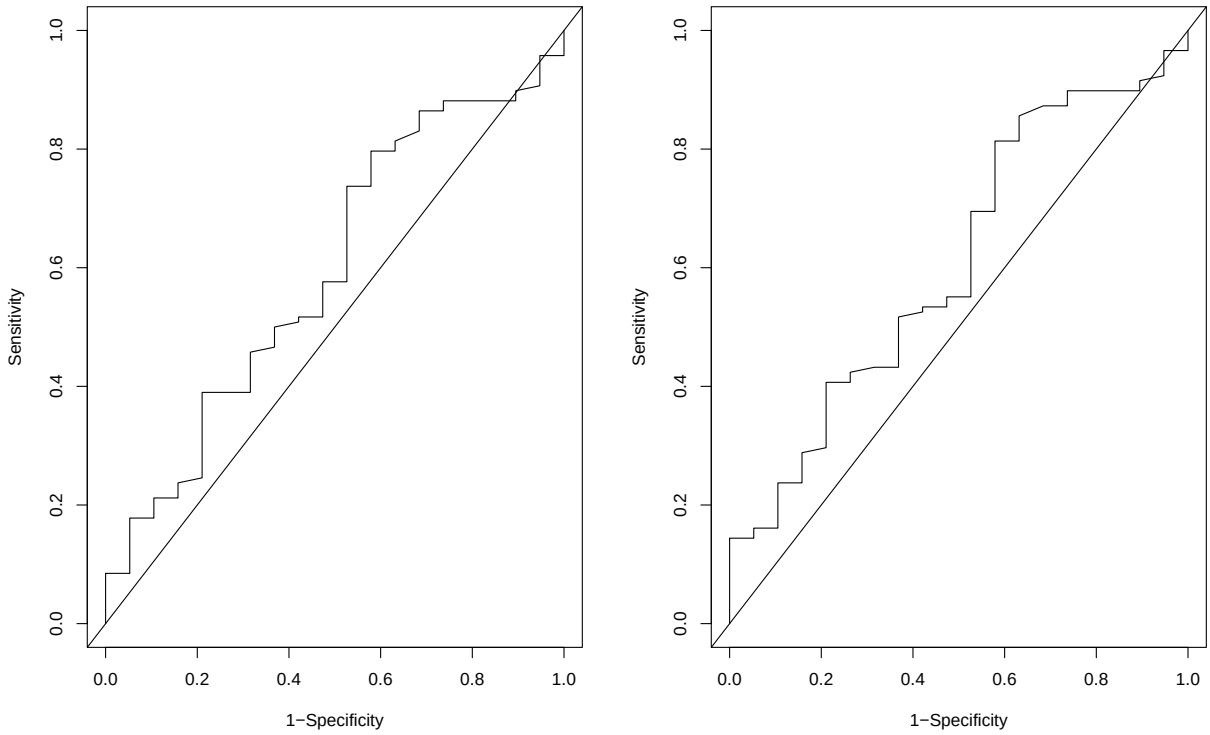


Figure 2: ROC Curves for Linear (left) and Logistic Regression (right) for Applicants with Checking Values 3 and 4

References

Altair. (2022, May 18th). Credit Scoring Series Part Two: Credit Scorecard Modeling Methodology. Retrieved from Altair: https://www.altair.com/newsroom/articles/credit-scoring-series-part-two-credit-scorecard-modeling-methodology Bee Wah Yap, S. H. (2011). Using data mining to improve assessment of credit worthiness via credit scoring models. Elsevier, 1-10.

Bellera, C. M. (2010). Variables with time-varying effects and the Cox model: Some statistical concepts illustrated with a prognostic factor study in breast cancer.

Centre for Capital Markets. (2021). The Economic Benfits of Risk-Based Pricing. Retrieved from https://www.centerforcapitalmarkets.com/wp-content/uploads/2021/04/CCMC_RBP_v11-2.pdf

Deloitte. (2016, April). Credit scoring - A Case Study in Data Analytics. Retrieved from https://www2.deloitte.com/content/dam/Deloitte/global/Documents/Financial-Services/gx-be-aers-fsi-credit-scoring.pdf

Helen Tadesse Moges, K. D. (2012). A multidimensional analysis of data quality for credit risk management: new insights and challenges. Elsevier .

Helen-Tadesse Mogesa, K. D. (2011). Data Quality for Credit Risk Management: New Insights and Challenges. JO - ICIQ 2011 - Proceedings of the 16th International Conference on Information Quality, 1-15.

Nailong Zhang, Q. Y. (2019). A new mixture cure model under competing risks to score online consumer loans. Quantitative Finance, 1243-1253.

Narain, B. (1992). Survival Analysis and the Credit Granting Decision. Thomas, L.C., Crook, J.N. Plug & Score. (n.d.). Scorecard Development Stages. Retrieved from Plug & Score: https://plug-n-score.com/learning/scorecard-development-stages.html

Sabato, G. (2010). Credit Risk Scoring Model. The Journal of Risk Finance, 10-11.

Siana Halim, Y. V. (2014). Credit Scoring Modeling. Jurnal Teknik Industri, Vol.16, No.1, 1-9.

Stepanova, M. (2001). Using survival analysis methods to build credit scoring models.

White, A. M. (2004). Risk-Based Mortgage Pricing: Present and Future Research. Housing Policy Debate, 503-531.

Zeng, G. (2014). A Neccessary Condition for a Good Binning Algorithm in Credit Scoring. Applied Mathematical Sciences, 1-14.

Appendix

Binary Variables:

Checking 1 and 2:

Purpose

1: Car (new)

2: Car (used)

3: Furniture/equipment

4: Radio/Television

5: Domestic appliances

6: Repairs

7: Education

8: (Vacation - does not exist?)

9: Retraining

10: Business

11: Others

Duration Bins

1:Duration [-Inf,8)

2:Duration [8,12)

3:Duration [12,14)

4:Duration [14,18)

5:Duration [18,20)

6:Duration [20,34)

7:Duration [34,44)

8:Duration [44, Inf)

History Bins

1: No credit history/all credits paid back duly

2: All credits at this bank paid back duly

3: Existing Credits paid back duly till now

4: Delay in paying in the past

5: Critical account/other credits existing (not at this bank)

Savings Bins

1: < 100 DM

2: 100 <= . . . < 500 DM

3: 500 <= . . . < 1000 DM

4: .. >= 1000 DM

5: unknown/ no savings account

Age Bins

1: Age [-Inf,23)

2: Age [23,24)

3: Age [24,28)

4: Age [28,36)

5: Age [36,38)

6: Age [38,49)

7: Age [49,61)

8: Age [61, Inf)

Checking 3 and 4:

History Bins

1: No credit history/all credits paid back duly

2: All credits at this bank paid back duly

3: Existing Credits paid back duly till now

4: Delay in paying in the past

5: Critical account/other credits existing (not at this bank)

Savings Bins

1: < 100 DM

2: 100 <= . . . < 500 DM

3: 500 <= . . . < 1000 DM

4: .. >= 1000 DM

5: unknown/ no savings account

Age Bins

Age: [-Inf,25)

Age: [25,27)

Age: [27,30)

Age: [30,32)

Age: [32,36)

Age: [36,39)

Age: [39,50)

Age: [50, Inf)

Amount Bins

1: Amount [-Inf,800)

2: Amount [800,2000)

3: Amount [2000,2200)

4: Amount [2200,2600)

5: Amount [2600,2800)

6: Amount [2800,3800)

7: Amount [3800,6000)

8: Amount [6000, Inf)

Duration Bins

1:Duration [-Inf,8)

2:Duration [8,10)

3:Duration [10,12)

4:Duration [12,14)

5:Duration [14,16)

6:Duration [16,34)

7:Duration [34,38)

8:Duration [38, Inf)

Regression Model Output:

Linear regression with Checking Values 1 or 2:

```
##
## Call:
## lm(formula = Good ~ Duration_bin + History + Savings + Age_bin,
##     data = train1bins)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9284 -0.4691  0.1631  0.3962  0.7994
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            0.52021    0.14171   3.671 0.000279 ***
## Duration_bin[12,14)   -0.18447    0.10534  -1.751 0.080778 .
## Duration_bin[14,18)   -0.10314    0.13317  -0.774 0.439169
## Duration_bin[18,20)   -0.20057    0.11275  -1.779 0.076111 .
## Duration_bin[20,34)   -0.24110    0.10176  -2.369 0.018356 *
## Duration_bin[34,44)   -0.34637    0.11676  -2.967 0.003216 **
## Duration_bin[44, Inf) -0.43184    0.12574  -3.434 0.000664 ***
## Duration_bin[8,12)    -0.06218    0.12629  -0.492 0.622796
## History                0.08537    0.02392   3.569 0.000408 ***
## Savings                0.05101    0.01729   2.951 0.003378 **
## Age_bin[23,24)        -0.21070    0.13656  -1.543 0.123742
## Age_bin[24,28)        -0.03777    0.10230  -0.369 0.712222
## Age_bin[28,36)        -0.07514    0.09694  -0.775 0.438776
## Age_bin[36,38)         0.03976    0.12665   0.314 0.753751
## Age_bin[38,49)        -0.01428    0.10481  -0.136 0.891709
## Age_bin[49,61)        -0.05847    0.12299  -0.475 0.634785
## Age_bin[61, Inf)      -0.02430    0.13909  -0.175 0.861396
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4723 on 355 degrees of freedom
## Multiple R-squared:  0.1353, Adjusted R-squared:  0.09632
## F-statistic: 3.472 on 16 and 355 DF,  p-value: 0.00000874
```

Logistic regression with Checking Values 1 or 2:

```
##
## Call:
## glm(formula = Good ~ Duration_bin + History + Savings + Age_bin,
##     family = "binomial", data = train1bins)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0976  -1.1248   0.5968   0.9873   1.7991
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)            0.24940    0.70370   0.354 0.723024
## Duration_bin[12,14)   -1.09352    0.56976  -1.919 0.054952 .
## Duration_bin[14,18)   -0.70347    0.69821  -1.008 0.313678
## Duration_bin[18,20)   -1.16528    0.59949  -1.944 0.051922 .
## Duration_bin[20,34)   -1.33531    0.55751  -2.395 0.016615 *
## Duration_bin[34,44)   -1.78348    0.61206  -2.914 0.003569 **
## Duration_bin[44, Inf) -2.20456    0.66076  -3.336 0.000849 ***
## Duration_bin[8,12)    -0.49623    0.67684  -0.733 0.463462
## History                0.40968    0.11511   3.559 0.000372 ***
## Savings                0.24821    0.08397   2.956 0.003119 **
## Age_bin[23,24)        -0.99995    0.61643  -1.622 0.104770
## Age_bin[24,28)        -0.16996    0.45143  -0.377 0.706543
## Age_bin[28,36)        -0.34838    0.42793  -0.814 0.415585
## Age_bin[36,38)         0.19528    0.59484   0.328 0.742696
## Age_bin[38,49)        -0.05373    0.46744  -0.115 0.908480
## Age_bin[49,61)        -0.28402    0.55911  -0.508 0.611455
## Age_bin[61, Inf)      -0.09089    0.63895  -0.142 0.886886
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 510.00  on 371  degrees of freedom
## Residual deviance: 455.09  on 355  degrees of freedom
## AIC: 489.09
##
## Number of Fisher Scoring iterations: 4
```

Linear regression with Checking Values 3 or 4:

```
##
## Call:
## lm(formula = Good ~ Duration_bin + History + Savings + Age_bin,
##     data = train2bins)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.00362  0.04961  0.10207  0.15171  0.41850
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)             0.897747   0.092518   9.703  < 2e-16 ***
## Duration_bin[10,12)    -0.084283   0.094702  -0.890 0.374180
## Duration_bin[12,14)    -0.080383   0.069871  -1.150 0.250873
## Duration_bin[14,16)    -0.030031   0.091914  -0.327 0.744097
## Duration_bin[16,34)    -0.144892   0.064393  -2.250 0.025160 *
## Duration_bin[34,38)    -0.295537   0.087550  -3.376 0.000833 ***
## Duration_bin[38, Inf)  -0.137957   0.094364  -1.462 0.144791
## Duration_bin[8,10)     -0.108900   0.100763  -1.081 0.280672
## History                 0.010894   0.018381   0.593 0.553825
## Savings                 0.008348   0.011786   0.708 0.479312
## Age_bin[25,27)          0.057744   0.080189   0.720 0.472022
## Age_bin[27,30)         -0.029058   0.079759  -0.364 0.715870
## Age_bin[30,32)          0.109766   0.093067   1.179 0.239155
## Age_bin[32,36)          0.075737   0.073136   1.036 0.301236
## Age_bin[36,39)          0.020782   0.081761   0.254 0.799533
## Age_bin[39,50)          0.022359   0.071007   0.315 0.753066
## Age_bin[50, Inf)        0.068109   0.078298   0.870 0.385064
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3336 on 302 degrees of freedom
## Multiple R-squared:  0.0595, Adjusted R-squared:  0.009671
## F-statistic: 1.194 on 16 and 302 DF,  p-value: 0.2712
```

Logistic regression with Checking Values 3 or 4:

```
##
## Call:
## glm(formula = Good ~ Duration_bin + History + Savings + Age_bin,
##     family = "binomial", data = train2bins)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8120   0.2828   0.4455   0.5661   1.2083
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)            2.97177    1.21047   2.455  0.01409 *
## Duration_bin[10,12)   -1.50651    1.26810  -1.188  0.23483
## Duration_bin[12,14)   -1.44193    1.10867  -1.301  0.19340
## Duration_bin[14,16)   -0.59705    1.45186  -0.411  0.68090
## Duration_bin[16,34)   -2.03629    1.05499  -1.930  0.05359 .
## Duration_bin[34,38)   -2.91292    1.11785  -2.606  0.00917 **
## Duration_bin[38, Inf) -1.96570    1.20862  -1.626  0.10387
## Duration_bin[8,10)    -1.71565    1.28017  -1.340  0.18019
## History                0.08757    0.17251   0.508  0.61170
## Savings                0.07682    0.11200   0.686  0.49278
## Age_bin[25,27)         0.47857    0.69230   0.691  0.48939
## Age_bin[27,30)        -0.20803    0.65313  -0.319  0.75009
## Age_bin[30,32)         0.99583    0.90668   1.098  0.27206
## Age_bin[32,36)         0.71062    0.67610   1.051  0.29323
## Age_bin[36,39)         0.17894    0.69992   0.256  0.79822
## Age_bin[39,50)         0.15815    0.61587   0.257  0.79734
## Age_bin[50, Inf)       0.70409    0.78391   0.898  0.36909
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 244.72  on 318  degrees of freedom
## Residual deviance: 225.66  on 302  degrees of freedom
## AIC: 259.66
##
## Number of Fisher Scoring iterations: 6
```

Code for Coursework

```
#Read German Credit Data into R

#include the correct path to the function

creditdata <- readxl::read_xlsx("C:/Users/shaun/Desktop/Credit Risk Scoring/GermanCreditData.xlsx", sheet=1)

creditdata

#Convert to Dataframe

creditdata <- data.frame(creditdata)

#Are there any missing values? - we observe some in the Purpose column

Missing <- creditdata$Purpose=='X'

Missing

sum(Missing)

#There are 12 missing values in our dataset which we chose to remove

creditdata <- creditdata[- grep("X", creditdata$Purpose),]

Missing2 <- creditdata$Purpose=='X'

Missing2

#Question 1 - Creation of Subsets

Subset1 <- subset(creditdata, creditdata$Checking =='1' | creditdata$Checking=='2') #Subset created for observations with checking scores of 1 and 2.

Subset2 <- subset(creditdata, creditdata$Checking =='3' | creditdata$Checking=='4')  #Subset created for observations with checking scores of 3 and 4.

#Check number of observations in each subset

nrow(Subset1) #532 observations

nrow(Subset2) #456 observations


#Question 2 - Validation & Test Sets - Insert information from word document

#Validation & Test sets created for Subset 1

train1 <- head(Subset1, round(nrow(Subset1) * 0.7))

validation1 <- tail(Subset1, round(nrow(Subset1) * 0.3))
```

```
#Validation & Test sets created for Subset 2
train2 <- head(Subset2, round(nrow(Subset2) * 0.7))
validation2 <- tail(Subset2, round(nrow(Subset2) * 0.3))
validation2
#Proportion of Goods

#Subset1
table(Subset1$Good) # 297 Good / 235 Bad
table(train1$Good) #163 Good / 209 Bad
table(validation1$Good) #72 good / 88 Bad

#Subset 2
table(Subset2$Good) # 396 Good / 60 Bad
table(train2$Good) #278 Good / 41 Bad
table(validation2$Good) #118 good / 19 Bad

#Question 2 - Validation & Test Sets - Insert information from word document

#Validation & Test sets created for Subset 1
train1 <- head(Subset1, round(nrow(Subset1) * 0.7))
validation1 <- tail(Subset1, round(nrow(Subset1) * 0.3))

#Validation & Test sets created for Subset 2
train2 <- head(Subset2, round(nrow(Subset2) * 0.7))
validation2 <- tail(Subset2, round(nrow(Subset2) * 0.3))
validation2
#Question 3 - Choosing Relevant Variables using IV

#Code to create more interpretable results
options(scipen=5)
```

```
#Information Value for Training Set 1

IV1 <- create_infotables(train1, y="Good") #Use info tables function to review Information Value of each variable

IV1$Summary


#Information Value for Training set 2

IV2 <-create_infotables(train2, y="Good")

IV2$Summary


#WRITE HERE ABOUT VARIABLE SELECTION RELATING TO IV.


#Taking the variables and binning them using chi-merge to see the highest IV for each set of bins.


#Training Model 1 binning


bins <- woebin(train1, y="Good", x=c("Age", "Duration"), method="chimerge")


woebin_plot(bins) #Check for WOE distribution given the relevant bins.


trainwoe <- woebin_ply(train1, bins) #Apply the WoE binning to the data-set.

train1bins <- woebin_ply(train1, bins, to="bin") #Creation of Different Training Model to produce P-values for bin analysis

train1bins <- train1bins[, c('Amount', 'Duration_bin', 'Purpose','Good', 'Age_bin', 'History', 'Savings')]


train1bins


#Training model 2 binning - variables Age, Amount, Duration and Purpose.

bins2 <- woebin(train2, y="Good", x=c("Age", "Duration", "Amount"), method="chimerge")

woebin_plot(bins2) #Check for WoE binning graphically

trainwoe2 <- woebin_ply(train2, bins2) #apply function to bin by WoE

train2bins <- woebin_ply(train2, bins2, to="bin")

train2bins <- train2bins[, c('Amount_bin', 'Duration_bin','Good', 'Age_bin', 'History', 'Savings')]
```

```
#Validation Set 1 - Binning to allign with Training set

age_breaks <- c(-Inf, 23, 24, 28, 36, 38, 49, 61, Inf)

duration_breaks <- c(-Inf, 8, 12, 14, 18, 20, 34, 44, Inf)

binsval <- woebin(validation1, y="Good", x=c("Age", "Duration"), breaks_list= list(Age = age_breaks, Duration = duration_breaks))

validation1 <- woebin_ply(validation1, binsval, to="bin")

validation1new <- validation1[, c('Amount', 'Duration_bin', 'Purpose','Good', 'Age_bin', 'History', 'Savings')]

validation1new

#Binning as follows before 23, 23-24, 24-28, 28-36,36-38,38-49,49-61, 61+

#Duration as follows before 8, 8-12, 12-14, 14-18, 18-20, 20-34, 34-44, 44+

binsval

#Validation Set 2 - Binning to allign with Training Set

age_breaks2 <- c(-Inf, 25, 27, 30, 32, 36, 39, 50, Inf)

duration_breaks2 <- c(-Inf, 8, 10, 12, 14, 16, 34, 38, Inf)

Amount_breaks <- c(-Inf, 800, 2000, 2200, 2600, 2800, 3800, 6000, Inf)

binsval2 <- woebin(validation2, y="Good", x=c("Age", "Duration", "Amount"), breaks_list= list(Age = age_breaks2, Duration = duration_breaks2, Amount = Amount_breaks))

validation2 <- woebin_ply(validation2, binsval2, to="bin")

validation2new <- validation2[, c('Amount_bin', 'Duration_bin','Good', 'Age_bin', 'History', 'Savings')]

validation2new

bins

par(mfrow=c(1, 2))

woebin_plot(bins, title="Plot for Training Model 1")

#Question 4 -Scorecards - remember appendix


#Scorecard function already takes arguments for bins and so doesn't provide neccessary regression analysis... both regressions provided here to give scorecard outcomes and point assignment in addition to regression outcomes.


#Training Model 1 - Logistic Regression
```

```
logmodel1bins <- glm(formula = Good ~ Duration_bin + History + Savings + Age_bin,
family="binomial", data = train1bins) #Model to show p-values in Logistic Regression

logmodel1 <- glm(formula = Good ~ Duration_woe + History + Savings + Age_woe, family="binomial",
data = trainwoe) #Model to use in Scorecard analysis

card1 <- scorecard(bins, logmodel1) #Scorecard generation

my_scores1 <- scorecard_ply(creditdata, card1) ##Show Scorecard scores



#Training Model 1 - Linear Regression

linearmodel1bins <- lm(formula = Good ~ Duration_bin + History + Savings + Age_bin,data =
train1bins) #Model to show p-values in Linear Regression

linearmodel1 <- lm(formula = Good ~ Duration_woe + History + Savings + Age_woe,data = trainwoe)

card2 <- scorecard(bins, linearmodel1) #Create Scorecard

my_scores2 <- scorecard_ply(creditdata, card2) #Show Scorecard scores



#Training Model 2 - Logistic Regression

logmodel2bins <- glm(formula = Good ~ Duration_bin + History + Savings + Age_bin,
family="binomial", data = train2bins) #Model to show p-values in Logistic Regression

logmodel2 <- glm(formula = Good ~ Duration_woe + Amount_woe + Purpose + Age_woe, family =
"binomial", data = trainwoe2)

card3 <- scorecard(bins2, logmodel2) #Create Scorecard

my_scores3 <- scorecard_ply(creditdata, card3) #Show Scorecard scores



#Training Model 2 - Linear Regression

linearmodel2bins <- lm(formula = Good ~ Duration_bin + History + Savings + Age_bin,data =
train2bins) #Model to show P-values in Linear Regression

linearmodel2 <- lm(formula = Good ~ Duration_woe + Amount_woe + Purpose + Age_woe, data =
trainwoe2)

card4 <- scorecard(bins2, linearmodel2) #Create Scorecard

my_scores4 <- scorecard_ply(creditdata, card4)#Show Scorecard scores
```

#Question 5- Derive ROC Curves for all scorecards

#Linear Regression Model - ROC Performance - Model 1

pred <- predict(linearmodel1bins, validation1new) #Make predictions on the validation set whether good or bad

prediction1 <- prediction(pred, validation1new$Good) #turn into a prediction type object #Make prediction of the amount of goods we can expect from validation set given our knowledge of those from the original date-set.

performance1 <- performance(prediction1, measure='tpr', x.measure='fpr') #use performance function to evaluate the efficacy of the predictions made against actual goods from validation 1.

plot(performance1, xlab='1-Specificity', ylab='Sensitivity') #plot the performance.

abline(a=0,b=1)

#Gini Coefficient

gini <- performance(prediction1, measure="auc") #Area under the curve - higher the value the better than predictive capacity of the results

gini <- gini@y.values[[1]]

gini <- 2*gini-1 # Gini Coefficient = 0.452178

gini

#kolmogorov-Smirnov Statistic

bads <- unlist(performance1@x.values) #change from performance object

goods <- unlist(performance1@y.values)

ks.test(goods, bads) #reports of KS score of 0.33.043

#Logistic Regression Model - ROC Performance - Model 1

predlog <- predict(logmodel1bins, validation1new) #Predict given our understanding of "Good" what probability for "Good" is in Validation 1

predictionlog<- prediction(predlog, validation1new$Good)

performancelog<- performance(predictionlog, measure='tpr', x.measure='fpr') #Predict the number of "Goods" given the model

plot(performancelog, xlab='1-Specificity', ylab='Sensitivity') #Plot these on a RoC Curve

abline(a=0,b=1)

```
#Gini Coefficient - Logistic Regression - Model1

ginilog <- performance(predictionlog, measure="auc") #Area under the curve - higher the value the better than predictive capacity of the results

ginilog <- ginilog@y.values[[1]]

ginilog <- 2*ginilog-1

ginilog   # Gini Coefficient = 0.4572285


#Kolmogorov-Smirnov Statistic - Logistic Regression - Model1

bads1 <- unlist(performancelog@x.values)

goods1 <- unlist(performancelog@y.values)

ks.test(goods1, bads1) #0.33043


#Linear Regression model 2 - ROC Performance

pred2 <- predict(linearmodel2bins, validation2new)

prediction2 <- prediction(pred2, validation2new$Good)

performance2 <- performance(prediction2, measure='tpr', x.measure='fpr')

plot(performance2, xlab='1-Specificity', ylab='Sensitivity')

abline(a=0,b=1)


#Gini Coefficient - Linear Regression - Model2

gini2 <- performance(prediction2, measure="auc") #Area under the curve - higher the value the better than predictive capacity of the results

gini2 <- gini2@y.values[[1]]

gini2 <- 2*gini2-1

gini2              #Gini Coefficient = 0.1873327


#Kolmogorov-Smirnov Statistic - Linear Regression - Model2

bads2 <- unlist(performance2@x.values)

goods2 <- unlist(performance2@y.values)

ks.test(bads2, goods2) # 0.19328


#Logistic Regression Model 2 - ROC Performance
```

```
predlog2 <- predict(logmodel2bins, validation2new)

predictionlog2 <- prediction(predlog2, validation2new$Good)

performancelog2 <- performance(predictionlog2, measure='tpr', x.measure='fpr')

plot(performancelog2, xlab='1-Specificity', ylab='Sensitivity')

abline(a=0,b=1)


#Gini Coefficient - Logistic Regression - Model2

ginilog2 <- performance(predictionlog2, measure="auc") #Area under the curve - higher the value the better than predictive capacity of the results

ginilog2 <- ginilog2@y.values[[1]]

ginilog2 <- 2*ginilog2-1

ginilog2        # Gini Coefficient = 0.2140946


#Kolmogorov-Smirnov Statistic - Logistic Regression - Model2

bads3 <- unlist(performancelog2@x.values)

goods3 <- unlist(performancelog2@y.values)

ks.test(goods3, bads3) #0.21008
```