INTRODUCTION TO CREDIT SCORING & DATA PREPARATION

Dr. Aric LaBarr

Institute for Advanced Analytics

INTRODUCTION TO SCORECARDS

What is a Scorecard?

- Common way of displaying the patterns found in a binary response model.
- Typically, people use logistic regression models.
- The main benefit is that a scorecard provides a clear and intuitive way of presenting the regression coefficients.

Scorecard Usage

these are just scorecards on logistic regression models.

- Credit Scoring
  - Equifax (http://www.equifax.com/home/en_us)
  - Experian (http://www.experian.com)
  - Transunion (http://www.transunion.com)
- Medicine / Healthcare
  - Trauma and Injury Severity Score (http://www.trauma.org/archive/scores/iss.html)
  - Coronary Heart Disease Risk Calculator (http://www.medcalc.com/heartrisk.html)
- Retail, IT and most cases where binary models can be applied.

CREDIT SCORING

Credit Scoring and Scorecards

- "One of the oldest applications of data mining, because it is one of the earliest uses of data to predict consumer behavior."
- David Edelman – Credit Director of Royal Bank of Scotland

Credit Scoring and Scorecards

actual model itself

- **Credit scoring** is a statistical model that assigns a risk value to prospective or existing credit accounts.
- A **credit scorecard** is a statistical risk model that was put into a special format designed for ease of interpretation.
- Scorecards are used to make strategic decisions such as accepting/rejecting applicants and deciding when to raise a credit line, as well as other decisions.

layer put on top of model. interpretation of model. this is what we hand to regulators.

Credit Scoring and Scorecards

- The credit scorecard format is very popular and successful in the consumer credit world for a number of reasons:
  1. People at all levels within an organization generally find it easy to understand and use.
  2. Regulatory agencies are accustomed to credit risk models presented in this fashion.
  3. Credit scorecards are straightforward to implement and monitor over time.

regulators not used to looking at ML models, regulators not trained at statistical model, they trained to make sure no one is underserved.

Example 1 Scorecard

- Cut-off = 500
- New customer:
  - Months Since Last Miss Payment: 32
  - Home: OWN
  - Salary: $30,000

Variable	Level	Scorecard Points
MISS	$x < 24$	100
MISS	$24 \leq x < 36$	120
MISS	$36 \leq x < 48$	185
MISS	$x \geq 48$	200
HOME	OWN	225
HOME	RENT	110
INCOME	$x < 10{,}000$	120
INCOME	$10{,}000 \leq x < 25{,}000$	140
INCOME	$25{,}000 \leq x < 35{,}000$	180
INCOME	$35{,}000 \leq x < 50{,}000$	200
INCOME	$x \geq 50{,}000$	225

Example 1 Scorecard

- Cut-off = 500
- New customer:
  - Months Since Last Miss Payment: 32
  - Home: OWN
  - Salary: $30,000
- Total Points:

$$120 + 225 + 180 = 525$$

- **ACCEPT FOR CREDIT**

Variable	Level	Scorecard Points
MISS	$x < 24$	100
MISS	$24 \leq x < 36$	120
MISS	$36 \leq x < 48$	185
MISS	$x \geq 48$	200
HOME	OWN	225
HOME	RENT	110
INCOME	$x < 10{,}000$	120
INCOME	$10{,}000 \leq x < 25{,}000$	140
INCOME	$25{,}000 \leq x < 35{,}000$	180
INCOME	$35{,}000 \leq x < 50{,}000$	200
INCOME	$x \geq 50{,}000$	225

Example 2 Scorecard

- Cut-off = 500
- New customer:
  - Months Since Last Miss Payment: 22
  - Home: OWN
  - Salary: $8,000

Variable	Level	Scorecard Points
MISS	$x < 24$	100
MISS	$24 \leq x < 36$	120
MISS	$36 \leq x < 48$	185
MISS	$x \geq 48$	200
HOME	OWN	225
HOME	RENT	110
INCOME	$x < 10{,}000$	120
INCOME	$10{,}000 \leq x < 25{,}000$	140
INCOME	$25{,}000 \leq x < 35{,}000$	180
INCOME	$35{,}000 \leq x < 50{,}000$	200
INCOME	$x \geq 50{,}000$	225

Example 2 Scorecard

- Cut-off = 500
- New customer:
  - Months Since Last Miss Payment: 22
  - Home: OWN
  - Salary: $8,000
- Total Points:

$$100 + 225 + 120 = 445$$

- **DO NOT ACCEPT FOR CREDIT**

Variable	Level	Scorecard Points
MISS	$x < 24$	100
MISS	$24 \leq x < 36$	120
MISS	$36 \leq x < 48$	185
MISS	$x \geq 48$	200
HOME	OWN	225
HOME	RENT	110
INCOME	$x < 10{,}000$	120
INCOME	$10{,}000 \leq x < 25{,}000$	140
INCOME	$25{,}000 \leq x < 35{,}000$	180
INCOME	$35{,}000 \leq x < 50{,}000$	200
INCOME	$x \geq 50{,}000$	225

Example 2 Scorecard

- Cut-off = 500
- New customer:
  - Months Since Last Miss Payment: 22
  - Home: OWN
  - Salary: $8,000
- Total Points:

$$100 + 225 + 120 = 445$$

- **DO NOT ACCEPT FOR CREDIT**

Variable	Level	Scorecard Points
MISS	$x < 24$	100
MISS	$24 \leq x < 36$	120
MISS	$36 \leq x < 48$	185
MISS	$x \geq 48$	200
HOME	OWN	225
HOME	RENT	110
INCOME	$x < 10{,}000$	120
INCOME	$10{,}000 \leq x < 25{,}000$	140
INCOME	$25{,}000 \leq x < 35{,}000$	180
INCOME	$35{,}000 \leq x < 50{,}000$	200
INCOME	$x \geq 50{,}000$	225

Example 2 Scorecard

- Cut-off = 500
- New customer:
  - Months Since Last Miss Payment: 22
  - Home: OWN
  - Salary: $8,000
- Total Points:

$$100 + 225 + 120 = 445$$

- **DO NOT ACCEPT FOR CREDIT**

Variable	Level	Scorecard Points
MISS	$x < 24$	100
MISS	$24 \leq x < 36$	120
MISS	$36 \leq x < 48$	185
MISS	$x \geq 48$	200
HOME	OWN	225
HOME	RENT	110
INCOME	$x < 10{,}000$	120
INCOME	$10{,}000 \leq x < 25{,}000$	140
INCOME	$25{,}000 \leq x < 35{,}000$	180
INCOME	$35{,}000 \leq x < 50{,}000$	200
INCOME	$x \geq 50{,}000$	225

Example 2 Scorecard

- Cut-off = 500
- New customer:
  - Months Since Last Miss Payment: 22
  - Home: OWN
  - Salary: $8,000
- Total Points:

$$100 + 225 + 120 = 445$$

- **DO NOT ACCEPT FOR CREDIT**

Variable	Level	Scorecard Points
MISS	$x < 24$	100
MISS	$24 \leq x < 36$	120
MISS	$36 \leq x < 48$	185
MISS	$x \geq 48$	200
HOME	OWN	225
HOME	RENT	110
INCOME	$x < 10{,}000$	120
INCOME	$10{,}000 \leq x < 25{,}000$	140
INCOME	$25{,}000 \leq x < 35{,}000$	180
INCOME	$35{,}000 \leq x < 50{,}000$	200
INCOME	$x \geq 50{,}000$	225

Discrete because you paid it doesnt matter Jan 6 or Jan 23..Jan payment is done and made.

Question when ask are you gonna default is are you gonna pay me next month? I dont care if first or last day not gonna pay you. Point is are you gonna miss pmt?

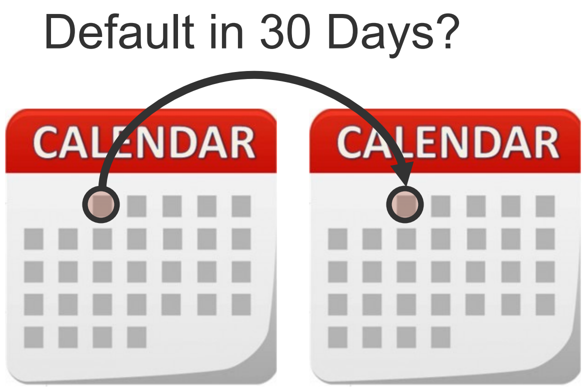
Discrete vs. Continuous Time

- Credit scoring typically tries to understand the probability of default on a customer (or business).
- However, default is also dependent on **time**.
- When will someone default? → **JUST AS IMPORTANT!**

- **Discrete** – Evaluating binary decisions on predetermined intervals of time.
- **Continuous** – Evaluating probability of default as it changes over continuous points in time (survival analysis).

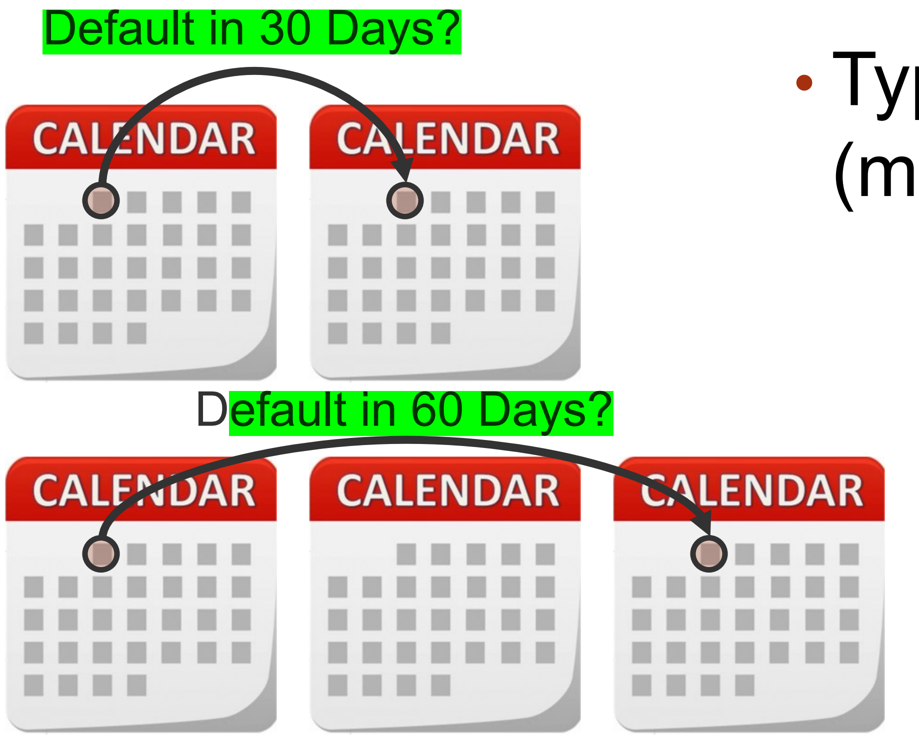
Discrete Time

Discrete Time



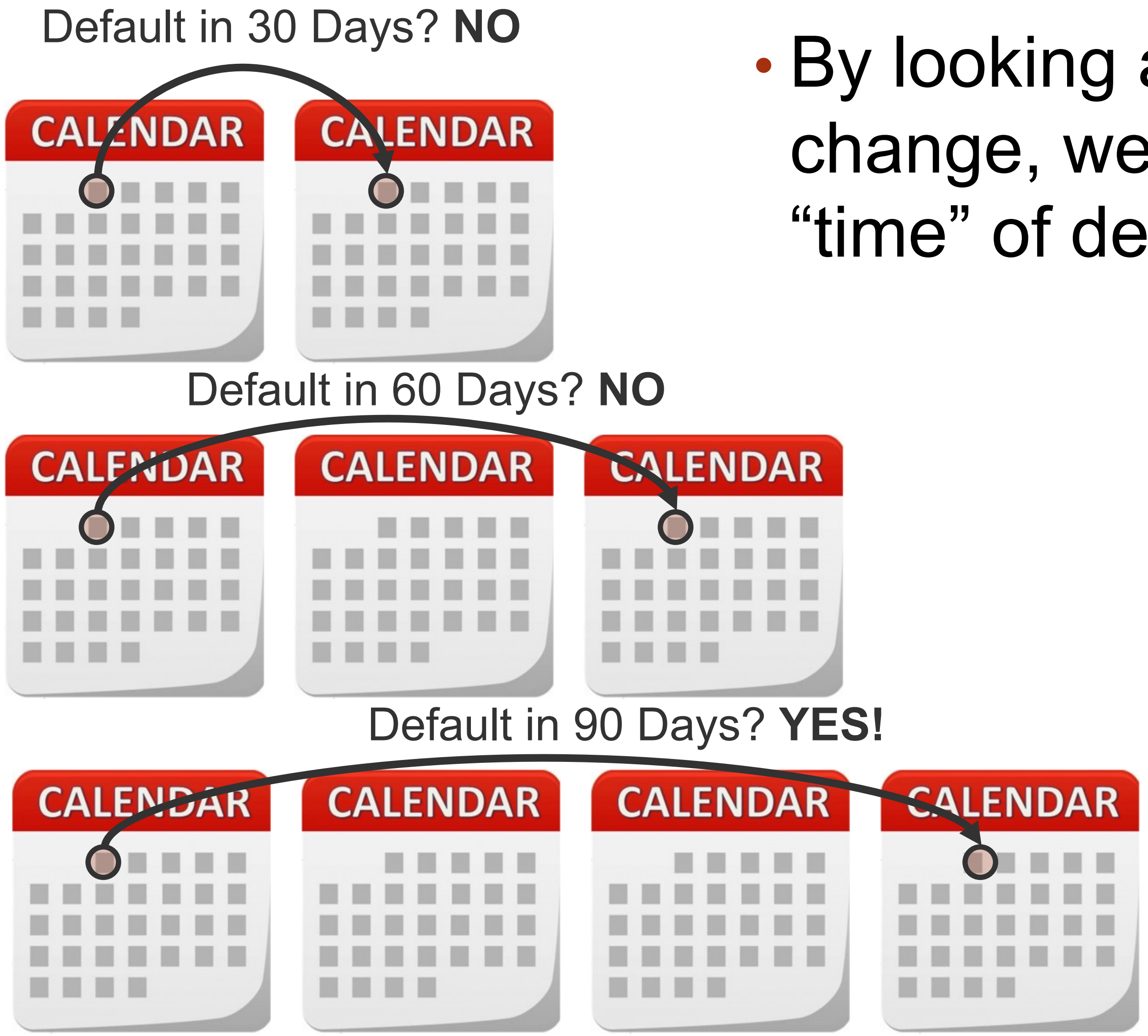
- Discrete time models evaluate the probability of default within a window of time.

Discrete Time



- Typically pick multiple windows (models) to evaluate across.

Discrete Time



- By looking at where decisions change, we can evaluate a “time” of default.

Continuous Time

Default in 42 Days

- Continuous time models provide a probability of default for **every** day.
- From this more exact times of default are possible.

Process Flow

Data Collection
- Variable Selection
- Sample Size
- Sample / Performance Window

Data Cleaning
- Eliminate Duplicates
- Examine / Remove Outliers

Variable Grouping and Selection
- Weights of Evidence (WOE)
- Information Value (IV)
- Gini Criterion

Initial Scorecard Creation
- Logistic Regression
- Accuracy
- Threshold
- Assessment

Reject Inference
- Remove bias resulting from exclusion of rejects

Final Scorecard Creation
- Final Model Assessment

DATA DESCRIPTION

ACCEPTS Data Set

- Type of Product: Auto Loans
- Information available on customers with performing or non-performing loans.
- 5,837 cases of individuals who applied for and were granted an automobile loan.
- 22 variables in all.

Data Dictionary

Variable Name	Description
Age_oldest_tr	Age of oldest trade
App_id	Application ID
Bad	Good/Bad Loan
Bankruptcy	Bankruptcy or Not
Bureau_score	Bureau Score
Down_pyt	Amount of down payment on vehicle
Loan_amt	Amount of Loan
Loan_term	How many months vehicle was financed
Ltv	Loan to Value
MSRP	Manufacturer suggested retail price
Purch_price	Purchase price of vehicle

Variable Name	Description
Purpose	Lease or own
Rev_util	Revolving utilization (balance/credit limit)
Tot_derog	Total number of derogatory trades (go DPD)
Tot_income	Applicant's income
Tot_open_tr	Number of open trades
Tot_rev_debt	Total revolving debt
Tot_rev_line	Total revolving line
Tot_rev_tr	Total revolving trades
Tot_tr	Total number of trades
Used_ind	Used car indicator
Weight	Weight variable

REJECTS Data Set

- Type of Product: Auto Loans
- 4,233 cases of individuals who applied for and were **NOT** granted an automobile loan.
- 21 variables in all – BAD variable not part of data set and should be inferred.
- Used for **reject inference** later in the analysis.

Reject Inference

- **Reject inference** is the process of inferring the status of the rejected applicants based on the accepted applicants model in an attempt to use their information to build a scorecard that is representative of the entire applicant population.
- Reject inference is about solving sample bias so that the development sample is similar to the population to which the scorecard will be applied.

Rejected Inference

- Can we develop a scorecard without rejected applications?

technically can still build scorecard wihtout rejected, legally permissible currently, but legislature in place to push through against it. Should be using entire applicant pool not just good candidate data set other get bias.

Weight assignment not work here cuz weight assigned on what data set seen by model (need model to see it ALL).

Rejected Inference

- Can we develop a scorecard without rejected applications? **YES!**
- Is it **legally permissible** to develop a scorecard without rejected applications?

Rejected Inference

- Can we develop a scorecard without rejected applications? **YES!**
- Is it **legally permissible** to develop a scorecard without rejected applications? **YES!**
- If yes, then how **biased** would the scorecard model be?

Rejected Inference

- Can we develop a scorecard without rejected applications? **YES!**
- Is it **legally permissible** to develop a scorecard without rejected applications? **YES!**
- If yes, then how **biased** would the scorecard model be? **DEPENDS!**

- *"My suggestion is to develop the scorecard using what data you have, but start saving rejected applications ASAP."*
  Raymond Anderson, Head of Scoring at Standard Bank Africa, South Africa

DATA COLLECTION AND CLEANING

Process Flow

Data Collection
- Variable Selection
- Sample Size
- Sample / Performance Window

Data Cleaning
- Eliminate Duplicates
- Examine / Remove Outliers

Variable Grouping and Selection
- Weights of Evidence (WOE)
- Information Value (IV)
- Gini Criterion

Initial Scorecard Creation
- Logistic Regression
- Accuracy
- Threshold
- Assessment

Reject Inference
- Remove bias resulting from exclusion of rejects

Final Scorecard Creation
- Final Model Assessment

Defining Our Target

decide default on loan based on how many payments missed. Miss 1 month out of 2 month loan. But miss 1 month out of 5y loan, doesnt not mean defaulted.

Banking has 90 dpd - days passed origiinal payment date. equates 3 pmts. Nowadays they have varying degrees of past due pmt. Bank write off loan after 6 months past due.

- When does someone actually default?
  - Is it when the loan is charged-off?
  - Probably signs of stopped paying before then

- Need to define target variable
  - 90 days past due (DPD) for everything (old approach)
  - 90-180 DPD based on types of loans, business sector, country regulations, etc. (current approach)
    - For example: US mortgages – 180 DPD

Variable Selection

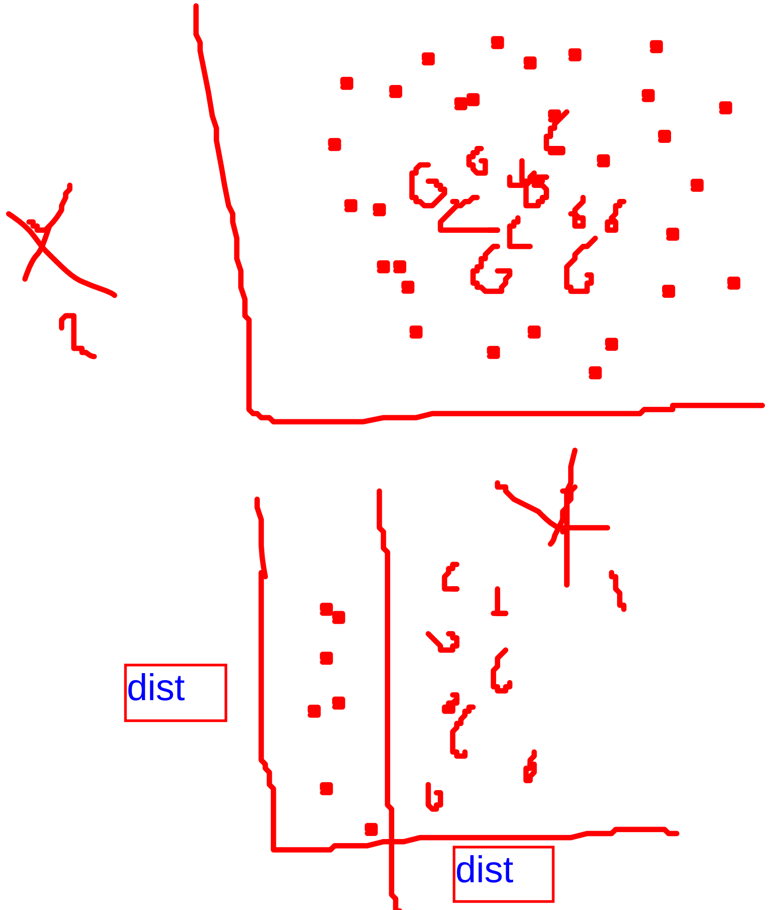
- Criteria for explanatory variables:
  - Expected predictability power
  - Business interpretation
  - Reliability
  - Legal issues
  - Ease in collection
  - Future availability

Feature Engineering

- Variable creation based on business reasoning:
  - Loan to value ratio
  - Number of delinquent accounts
  - Expense to income ratio
  - Credit line utilization
- Omit variables that are highly dependent:
  - Variable clustering!
- Review / remove outlier and abnormal values

Variable features will always make a model better comapred to technique. They beat technique.

logistic regression is a linear separator will be worse for if data was like this. Below can feature engineer distance from centre, then data look like this 2nd graph below:



Rich ppl doesnt mean they will pay it back cuz they take higher amt loan default prob could be same. Look at loan to income ratio, these features engineered make it better.

Still have to do multi coll.

Sample Size

- *"There are no hard and fast rules, but the sample selected normally includes at least 1,000 good, 1,000 bad, and about 750 rejected applicants."* FDIC, Credit Card Activities Manual (https://www.fdic.gov/regulations/examinations/credit_card/index.html)
- No exact answer on the correct sample size.
- Sample size depends on the overall size of the portfolio, the number of explanatory variables you are planning to use, and the number of defaults or claims filled.

Sample and Performance Window

- The sample must be characteristic of the population to which the scorecard will be applied.
- Example:
  - If the scorecard is to be applied in the subprime lending program, then use a sample that captures the characteristics of the subprime population targeted.

Sample and Performance Window

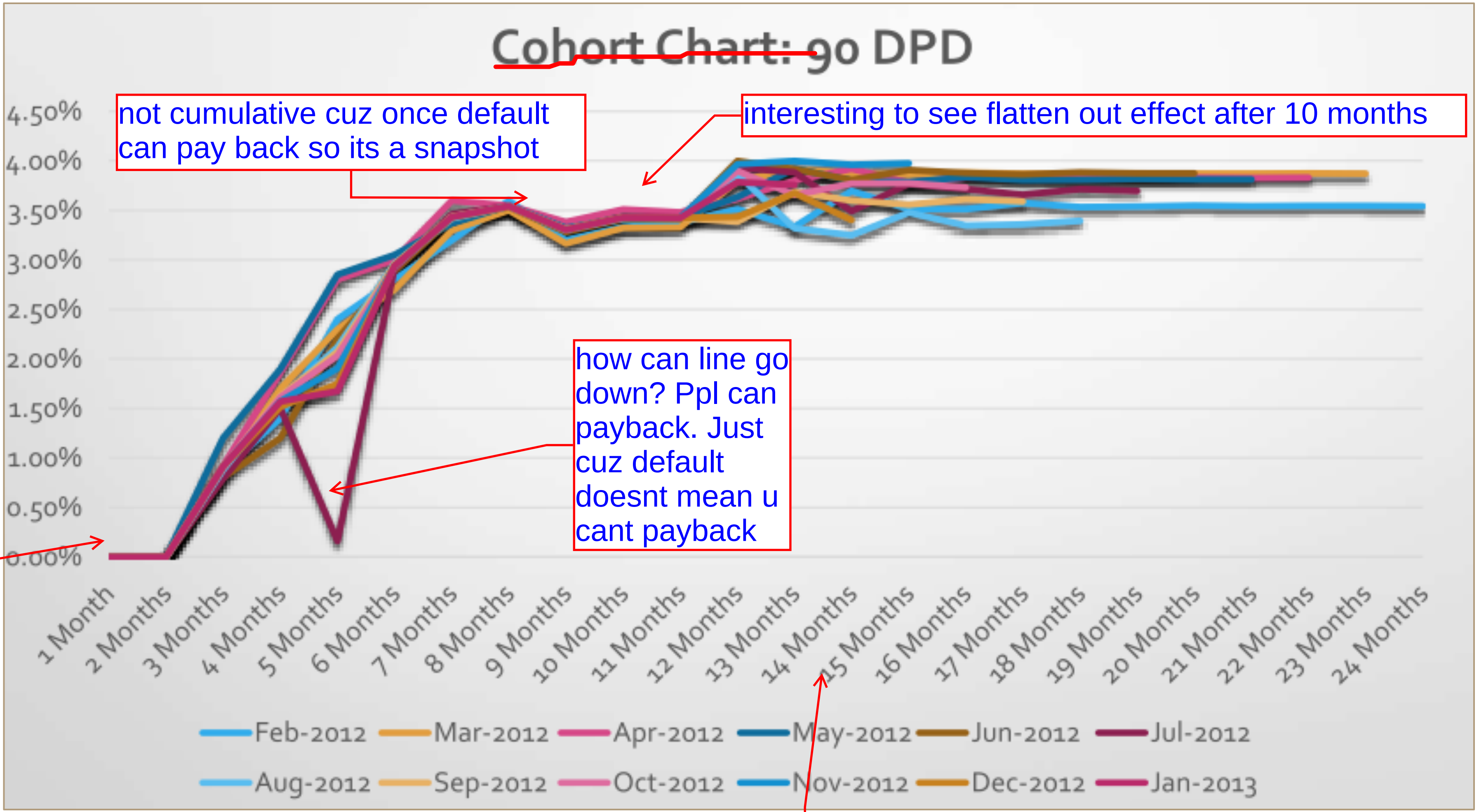
- Objective:
  - Gather data for accounts opened during a specific time frame.
  - Monitor their performance for another specific length of time to determine if they were good or bad.
- Problems:
  - Accounts opened recently are more similar to accounts that will be opened in the near future.
  - Want to minimize the chances of misclassifying performance – accounts must be monitored long enough to not underestimate expected bad rates.

how many ppl in default scneario over TIME

Sample and Performance Window

cohort chart each line represent cohort of people given loan a given month. We are using this to figure how long to give someone before they default ie become problem. "Cut off lenght of time"

Y axis is default rate. What % of ppl defaulted at that month out on x axis. This is defined by 180days



not cumulative cuz once default can pay back so its a snapshot

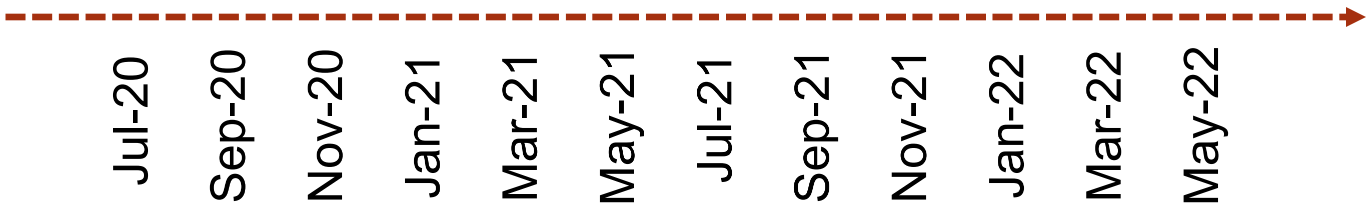
interesting to see flatten out effect after 10 months

how can line go down? Ppl can payback. Just cuz default doesnt mean u cant payback

havent had time to default After 3 months defaulted ie given loan but never pmt

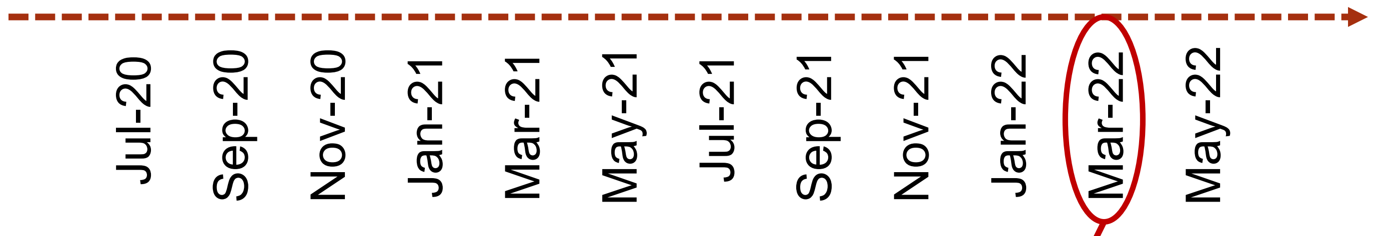
X axis is #of months since they given loan. Think survival analysis. tenure was same between 2 ppl but physical time wasnt same. Same idea here.

Sample and Performance Window



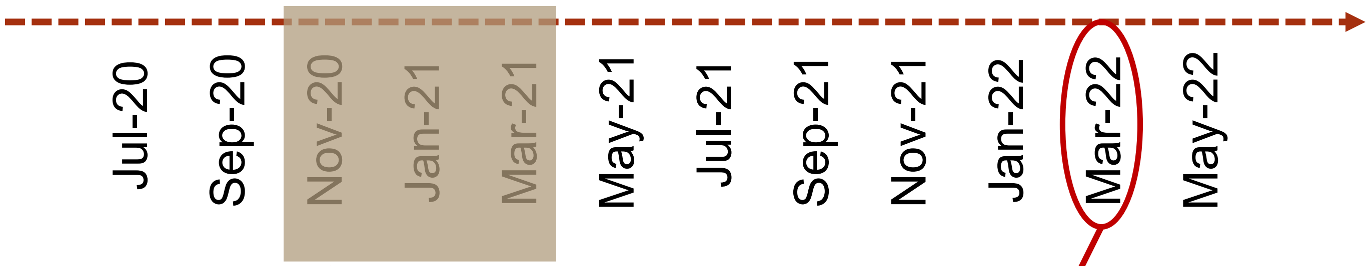
- Based on cohort graph: “Bads” level off about 14 months after loan origination.
- If the analysis is to be performed on March 2022, we select our **sample** from 12-16 months back; this will give an average of 14 months **performance window**.

Sample and Performance Window



- Based on cohort graph: “Bads” level off about 14 months after loan origination.
- If the analysis is to be performed on March 2022, we select our **sample** from 12-16 months back; this will give an average of 14 months **performance window**.

Sample and Performance Window

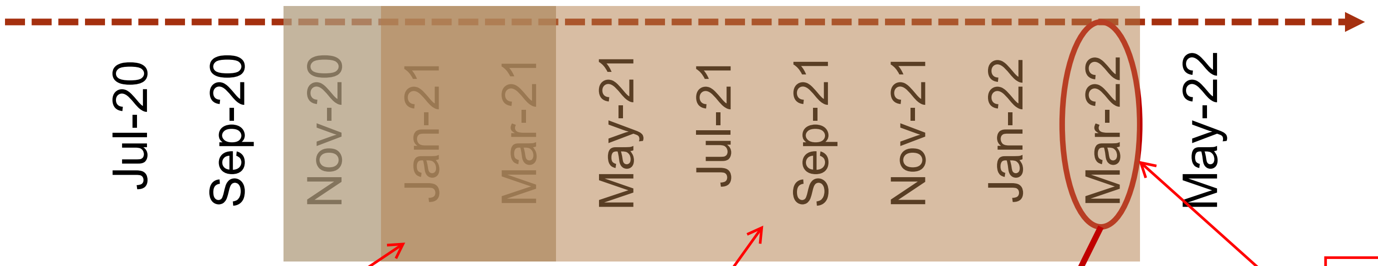


sample window where i get ppl from. Start 12-14 months back

- Based on cohort graph: “Bads” level off about 14 months after loan origination.
- If the analysis is to be performed on March 2022, we select our sample from 12-16 months back; this will give an average of 14 months **performance window**.

Sample and Performance Window

window changes based on recession time etc etc



sample window where get obs from

performance window. Waiting long enough to see default based on

cut off, 1 and 0s at this point

- Based on cohort graph: “Bads” level off about 14 months after loan origination.
- If the analysis is to be performed on March 2022, we select our **sample from 12-16 months back**; this will give an average of 14 months **performance window**.

if u buy sth too expensive for you, you will quickly if you default. Credit card tell 1-2 years if defaulting. 30y mortgage within 5y.

Sample and Performance Window

- The exact length of the performance window depends on the product.
    - Credit Cards: Typically 1 – 2 years
    - Mortgages: Typically 3 – 5 years
- Sample window length can vary based on data availability as well.

SCORECARD VARIABLE GROUPING AND SELECTION

Dr. Aric LaBarr

Institute for Advanced Analytics

Bin Variables cuz bye bye odds ratio. Essentially compare 2 categories. easiest interpretation out there. Mkaes it easy to compare. Can model non linearity

Optbinning package in Python

SMbinning package in R

Proc binning in SAS

Process Flow

Data Collection
- Variable Selection
- Sample Size
- Sample / Performance Window

Data Cleaning
- Eliminate Duplicates
- Examine / Remove Outliers

Variable Grouping and Selection
- Weights of Evidence (WOE)
- Information Value (IV)
- Gini Criterion

Initial Scorecard Creation
- Logistic Regression
- Accuracy
- Threshold
- Assessment

Reject Inference
- Remove bias resulting from exclusion of rejects

Final Scorecard Creation
- Final Model Assessment

VARIABLE GROUPING

Variable Grouping and Selection

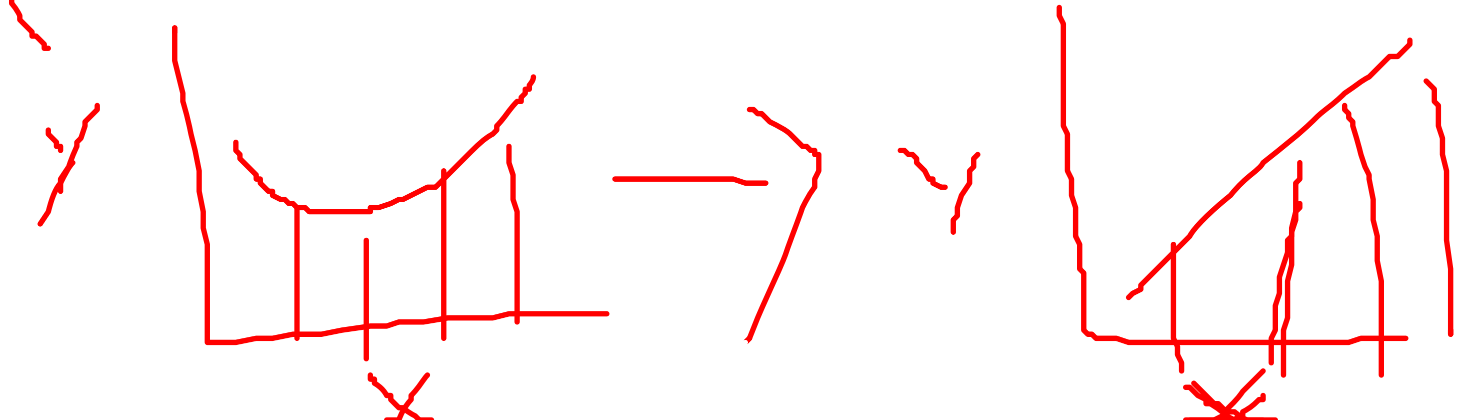
- Scorecards end up with only just groups within a variable.
- Objectives:
  1. Eliminate weak characteristics (variables) or those that do not conform to good business logic.
  2. Group the strongest characteristics' attribute levels in order to produce a model in scorecard format.
- Function/package "smbinning" in R.
- Package "scorecard" or "OptBinning" in Python.
- PROC BINNING in SAS VIYA.

Variable	Level
MISS	$x < 24$
MISS	$24 \leq x < 36$
MISS	$36 \leq x < 48$
MISS	$x \geq 48$
HOME	OWN
HOME	RENT

Why Grouping (Binning)?

- Goal is to help simplify analysis through grouping:
  - Useful for understanding relationships – no worries about explaining coefficients.
  - Modeling nonlinearities – similar to decision trees.
    (NO MORE LOGISTIC REGRESSION LINEARITY ASSUMPTION!)
  - Dealing with outliers – contained in the smallest / largest group.
  - Missing values typically in own group.

get their own category of missing



Initial Characteristic Analysis – SAS

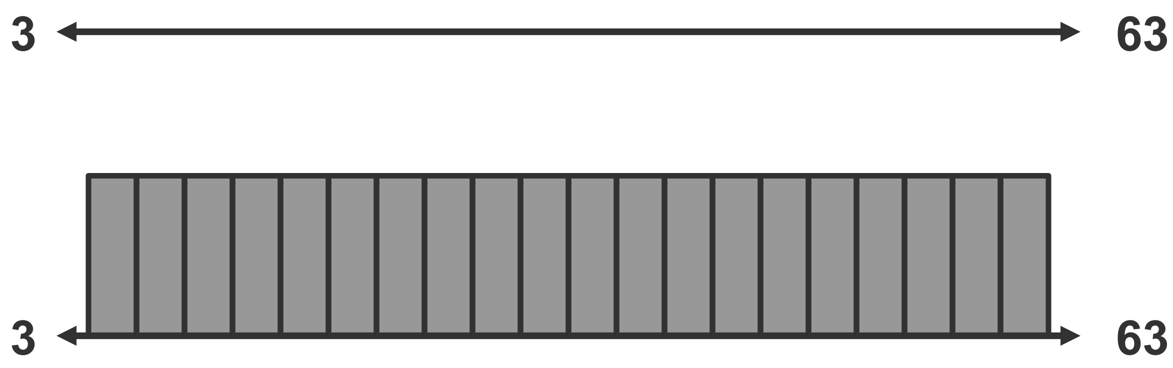
- Need a starting point for the grouping / binning.
  - Quantiles are most popular technique.
- Pre-bin the interval variables into a number of user-specified quantiles / buckets for fine detailed groupings.
- Aggregate the fine detailed groupings into a smaller number to produce coarse groupings.
  - Chi-squared tests to combine groups.

technique is pre bin and combine method. SAS first one to do this. Break into equal groups 20 to 100 groups. then they use chi sq test to combine together. predicting a binary target.

Initial Characteristic Analysis – SAS

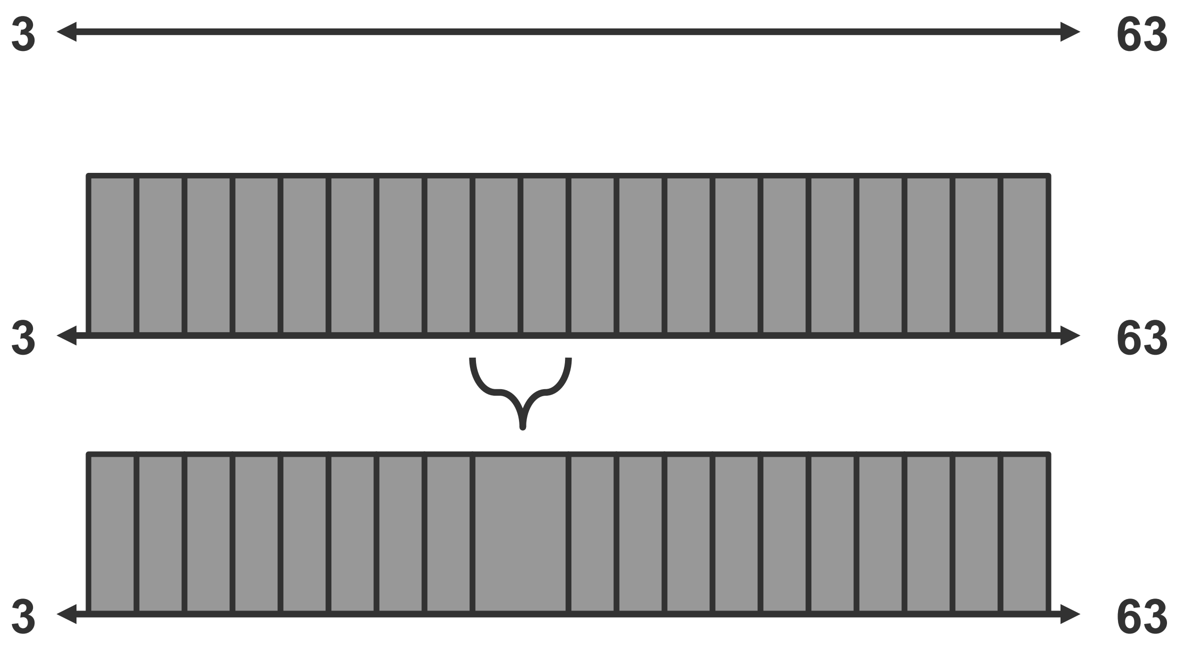
3 ⟷ **63**

Initial Characteristic Analysis – SAS

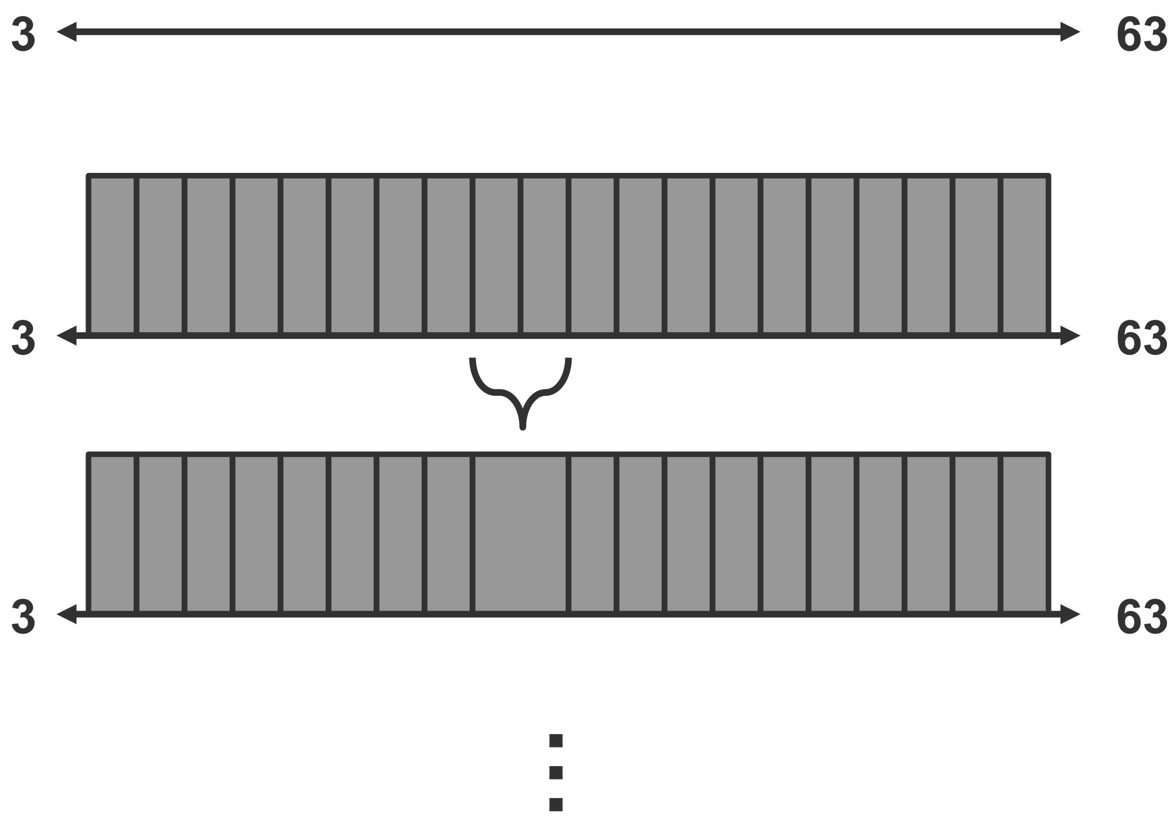


Initial Characteristic Analysis – SAS

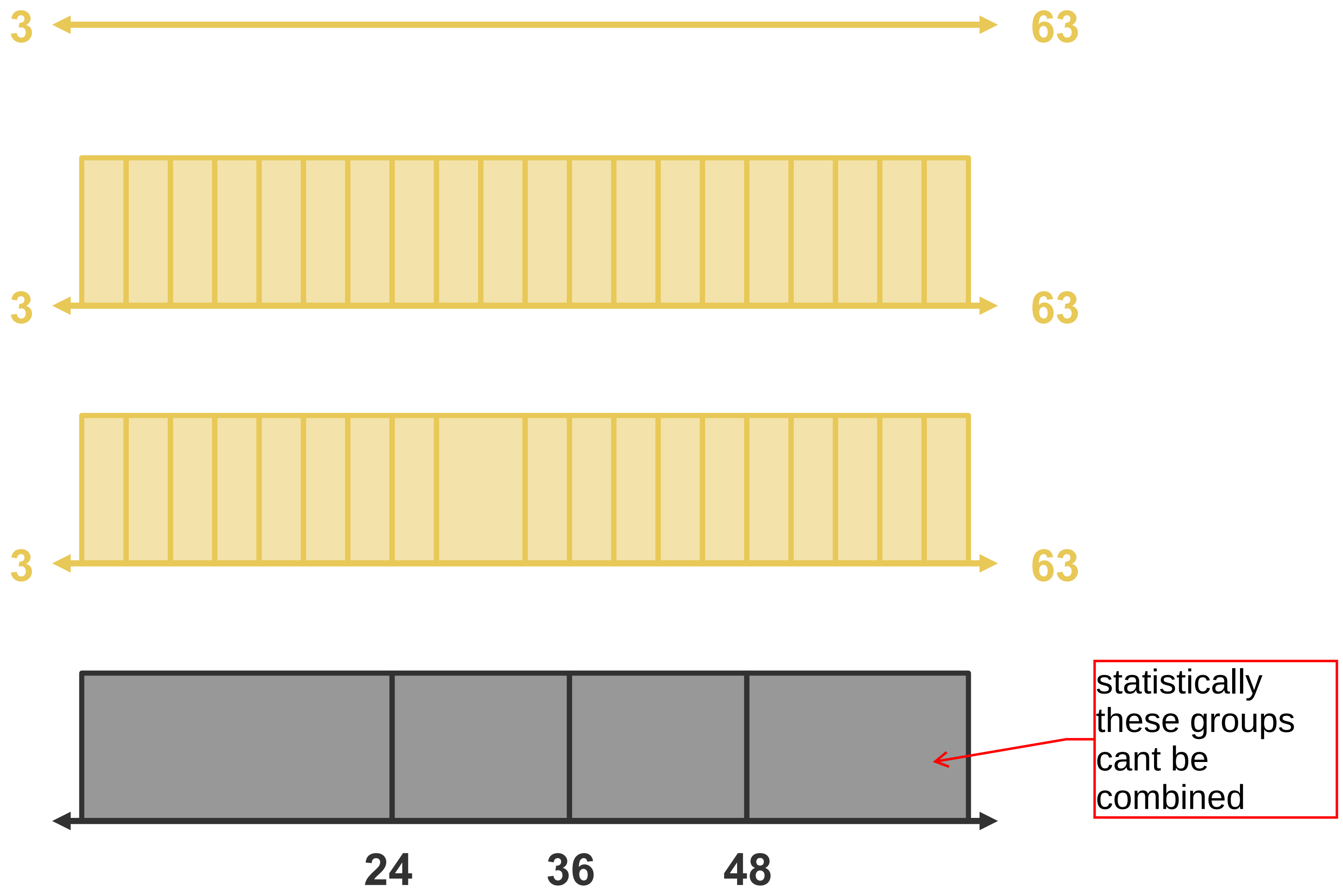
MH test - see if groups can be combined.
of those pairs it combines pairs that are
most significantly similar.



Initial Characteristic Analysis – SAS



Initial Characteristic Analysis – SAS



Initial Characteristic Analysis – R

Splitting not based on Gini. Based on Chi Sq test.

- The package (and function) "smbinning" uses a different approach than SAS.

- Conditional Inference Trees: CIT
  - CART methods have inherent bias – variables with more levels → more likely to be split on if split on Gini and Entropy.
  - CIT method adds extra statistical step before splits occur – statistical tests of significance.
  - What is **MOST** significant variable? → What is the best split (Chi-square) on **THIS** variable? → **REPEAT**.

SAS splits first then try and combine. in R, keep everythign combined then split. its basically a decision tree with one variable. split based on small p value. First do global test are there any splits.

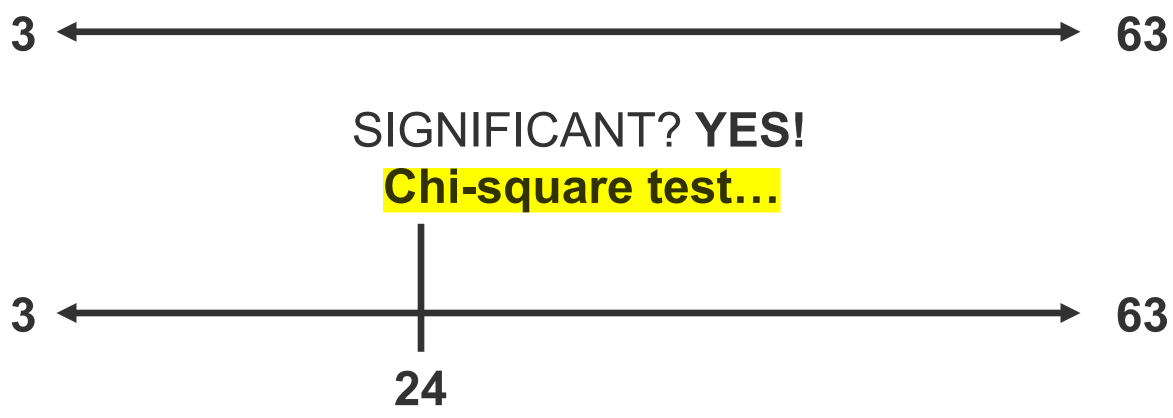
Opt binning package in Python does both methods - you pick technique A or B. Other package in Python does SAS way

Initial Characteristic Analysis – R

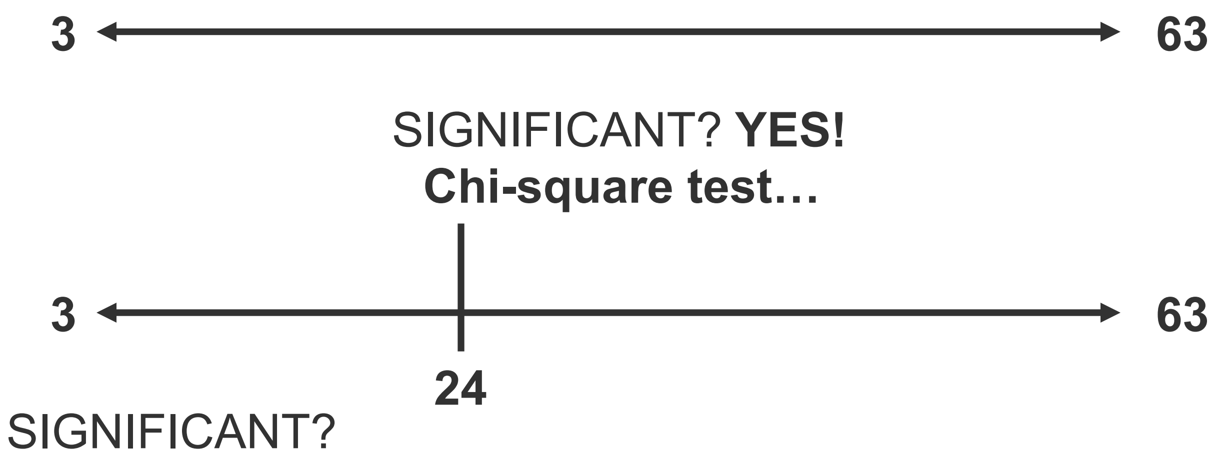
3 $\longleftrightarrow$ **63**

SIGNIFICANT?

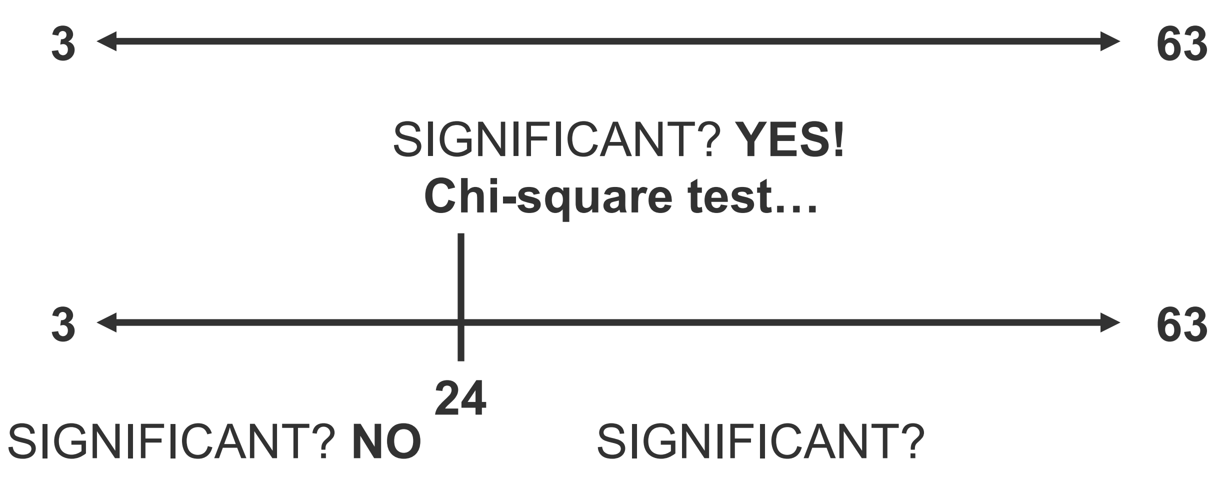
Initial Characteristic Analysis – R



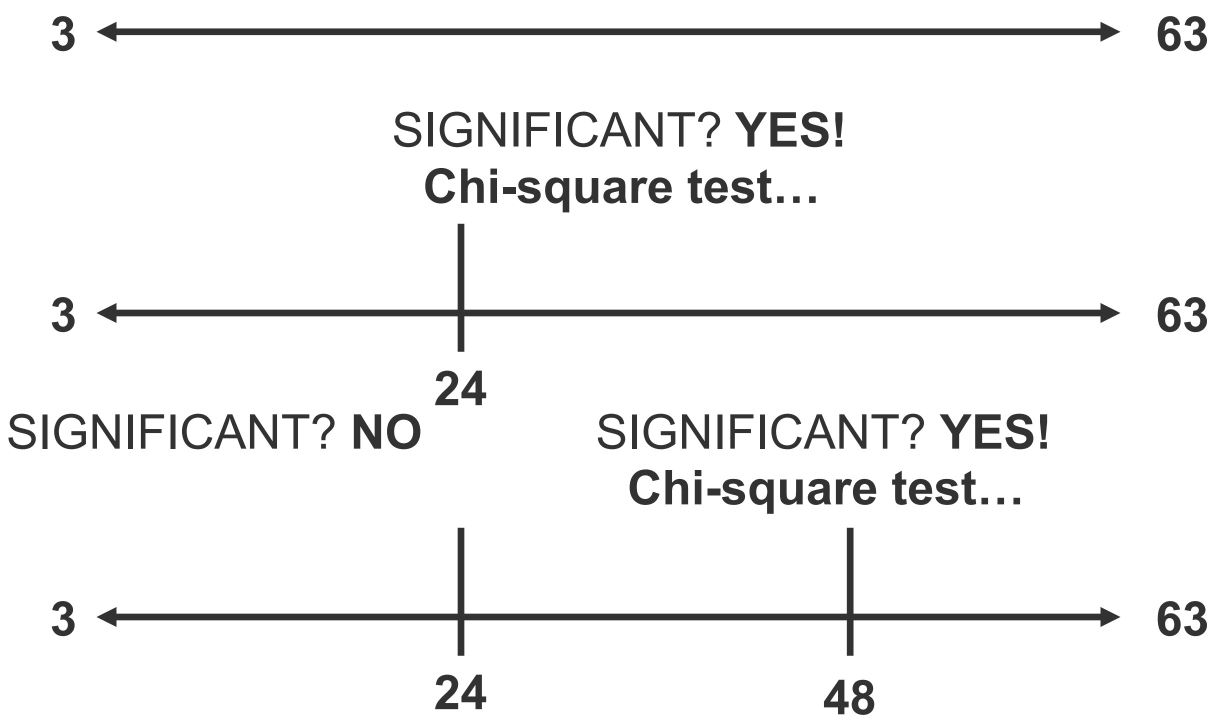
Initial Characteristic Analysis – R



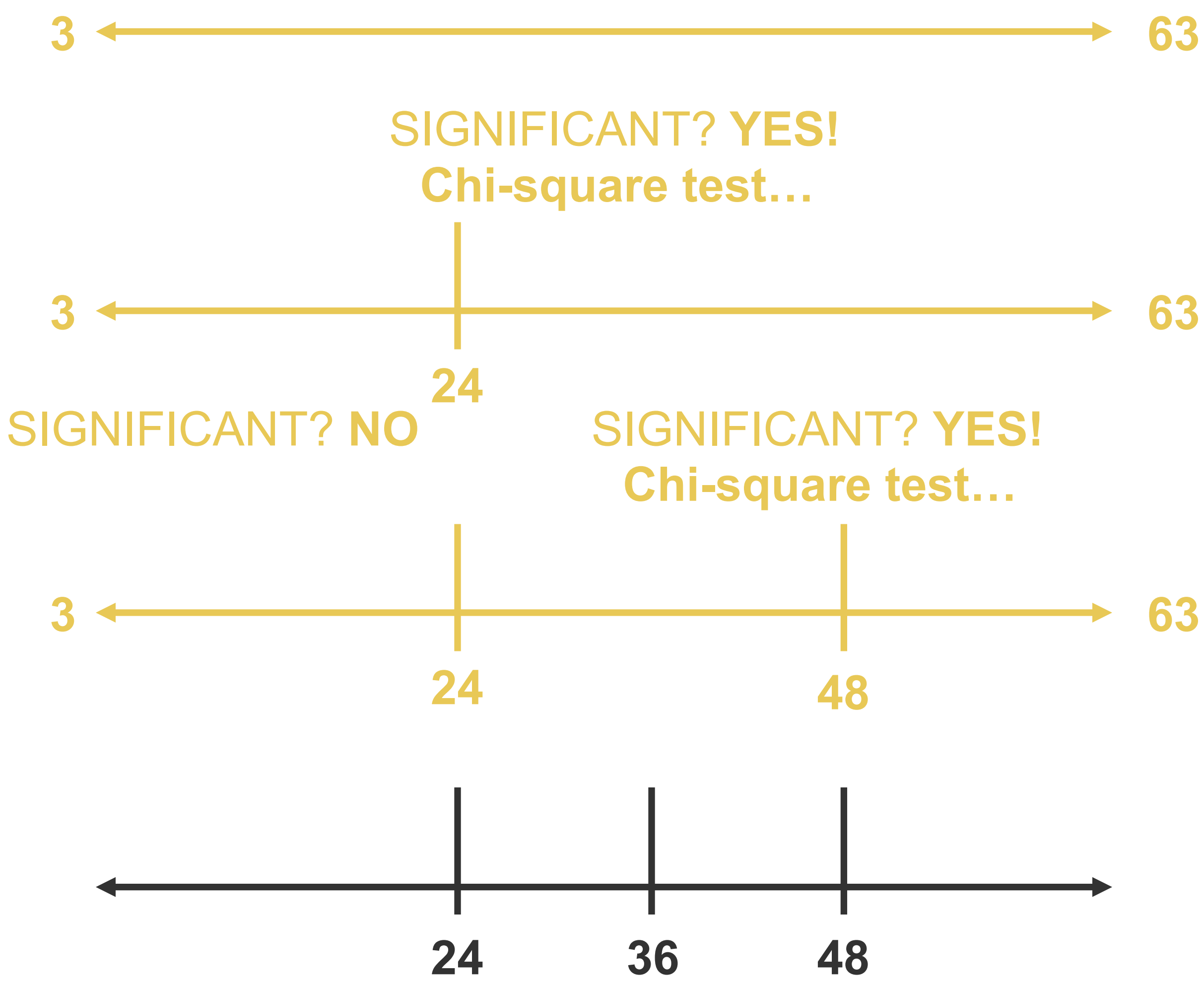
Initial Characteristic Analysis – R



Initial Characteristic Analysis – R



Initial Characteristic Analysis – R



Initial Characteristic Analysis

- Cut-offs may be rough from decision tree combining.
- **Optional to override** automatically generated groups to conform to business rules.
- Overrides may make groups suboptimal.

Group Definition
Missing
< $35,200
$35,200 - $60,000
$60,000 - $85,000
$85,000 - $110,000
$110,000 - $142,530
> $142,530

Initial Characteristic Analysis

- Cut-offs may be rough from decision tree combining.
- **Optional to override** automatically generated groups to conform to business rules.
- Overrides may make groups suboptimal.

Group Definition	Override
Missing	Missing
< $35,200	< $35,000
$35,200 - $60,000	$35,000 - $60,000
$60,000 - $85,000	$60,000 - $85,000
$85,000 - $110,000	$85,000 - $110,000
$110,000 - $142,530	$110,000 - $140,000
> $142,530	> $140,000

Initial Characteristic Analysis

- Cut-offs may be rough from decision tree combining.
- **Optional to override** automatically generated groups to conform to business rules.
- Overrides may make groups suboptimal.

Group Definition	Override
Missing	Missing
< $35,200	**< $35,000**
$35,200 - $60,000	**$35,000** - $60,000
$60,000 - $85,000	$60,000 - $85,000
$85,000 - $110,000	$85,000 - $110,000
$110,000 - $142,530	$110,000 - **$140,000**
> $142,530	**> $140,000**

Initial Characteristic Analysis

- Calculate and examine the key assessment metrics:
  - **Weight of Evidence** (WOE) – how well attributes discriminate for each given characteristic
  - **Information Value** (IV) – evaluate a characteristic's overall predictive power
  - **Gini Statistic** – alternate to IV for selecting characteristics for final model.

how well is each bin separating 1 or 1.
IV is how well variable as a whole is
Gini is rarely used

how good we
at separating
1s and 0s
power

WEIGHT OF EVIDENCE

Weight of Evidence (WOE)

bins

- WOE measures the strength of the attributes of a characteristic in **separating good and bad accounts**.
- WOE is based on comparing the proportion of goods to bads at each attribute level (levels of the predictor variable).

non defaulters

bankers came up with that.

$$WOE_i = \log\left(\frac{Dist.Good_i}{Dist.Bad_i}\right)$$

Weight of Evidence (WOE)

- WOE measures the strength of the attributes of a characteristic in **separating good and bad accounts**.
- WOE is based on comparing the proportion of goods to bads at each attribute level (levels of the predictor variable).

$$WOE_i = \log\left(\frac{Dist.Good_i}{Dist.Bad_i}\right)$$

$$Dist.Good_i = \frac{Number\ Good\ in\ group\ i}{Total\ Number\ Good}$$

divided by total good in all bins

Weight of Evidence (WOE)

- WOE measures the strength of the attributes of a characteristic in **separating good and bad accounts**.
- WOE is based on comparing the proportion of goods to bads at each attribute level (levels of the predictor variable).

$$WOE_i = \log\left(\frac{Dist.Good_i}{Dist.Bad_i}\right)$$

$$Dist.Bad_i = \frac{Number\ Bad\ in\ group\ i}{Total\ Number\ Bad}$$

Weight of Evidence (WOE)

- What are we looking for?
  - Looking for "big" differences in WOE between groups.
  - Monotonic changes within an attribute for **interval variables** (not always required).

ppl like to see monotonic changes. As variable gets bigger, ..

- Why monotonic increases?
  - Oscillation back and forth of positive to negative values of WOE typically sign of variable that has trouble separating good vs. bad.
  - Not always required **if makes business sense** – credit card utilization for example.

WOE – Example

not a target variable, it is a perdictor variable

Good group

WOE for Bureau Score				
Group	Values	Event Count	Non-event Count	WOE
1	< 603	111	112	
2	604 – 662	378	678	
3	663 – 699	185	754	
4	700 – 717	74	440	
5	718 – 765	75	824	
6	> 765	15	498	
7	MISSING	80	153	
Total		**918**	**3,459**	

WOE – Example

WOE for Bureau Score				
Group	Values	Event Count	Non-event Count	WOE
1	< 603	111	112	
2	604 – 662	378	678	
3	663 – 699	185	754	
4	700 – 717	74	440	
5	718 – 765	75	824	
6	> 765	15	498	
7	MISSING	80	153	
Total		**918**	**3,459**	

$$Dist.Good_1 = \frac{112}{3459}$$

$$= 0.032$$

WOE – Example

WOE for Bureau Score				
Group	Values	Event Count	Non-event Count	WOE
1	< 603	111	112	
2	604 – 662	378	678	
3	663 – 699	185	754	
4	700 – 717	74	440	
5	718 – 765	75	824	
6	> 765	15	498	
7	MISSING	80	153	
Total		**918**	**3,459**	

$$Dist.Good_1 = \frac{112}{3459}$$

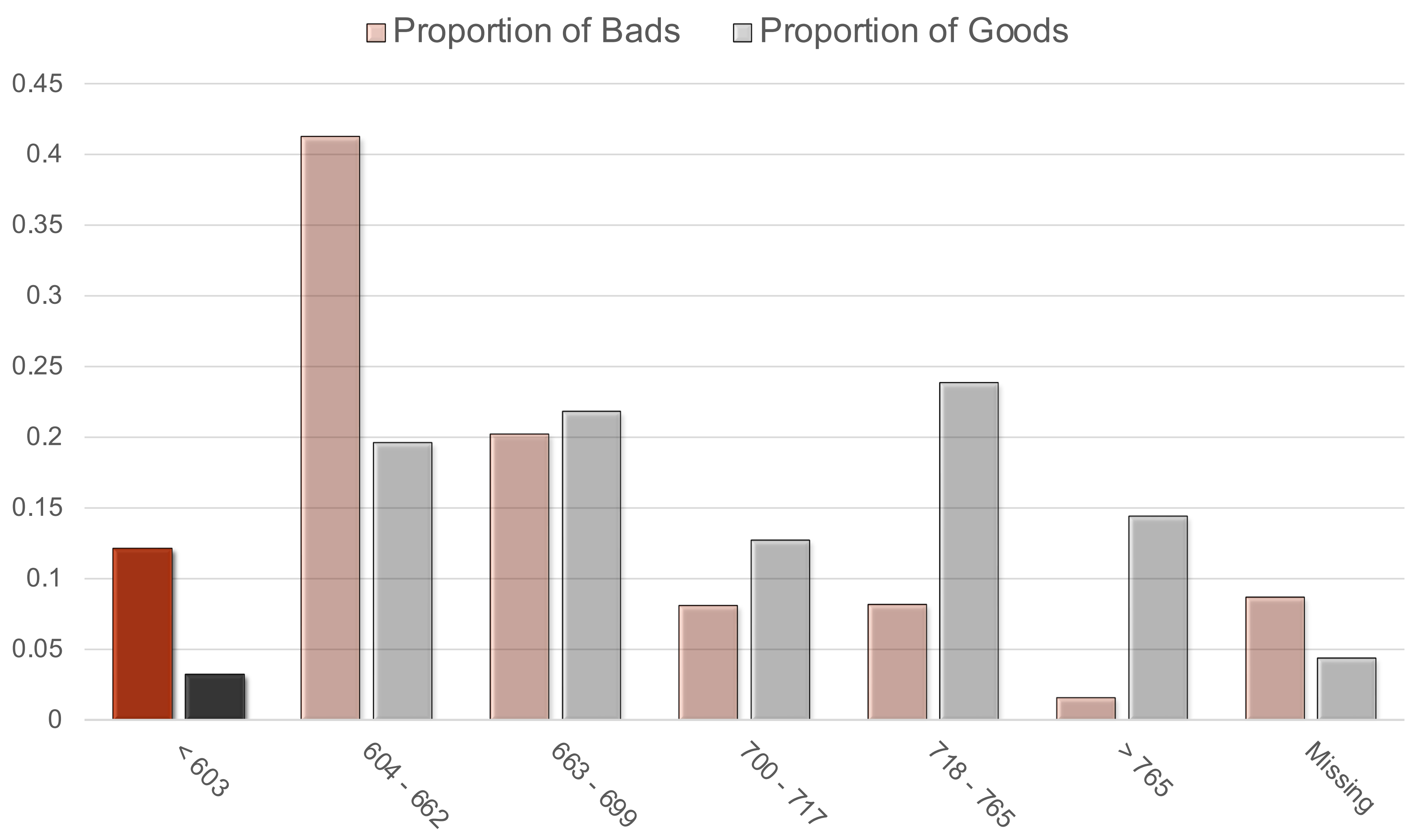
$$= 0.032$$

$$Dist.Bad_1 = \frac{111}{918}$$

$$= 0.121$$

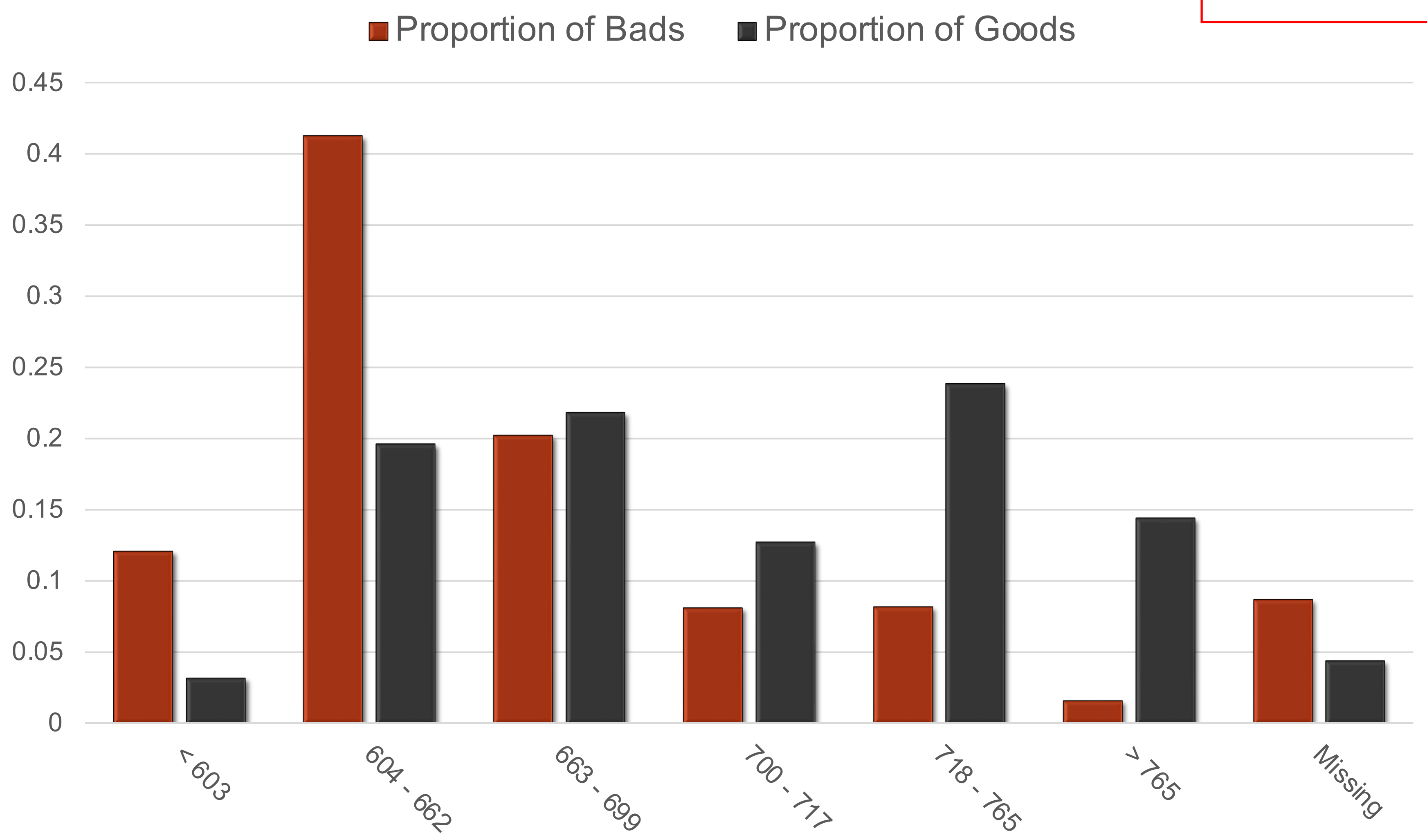
this category leans towards BAD 12 % vs 3%

WOE – Example

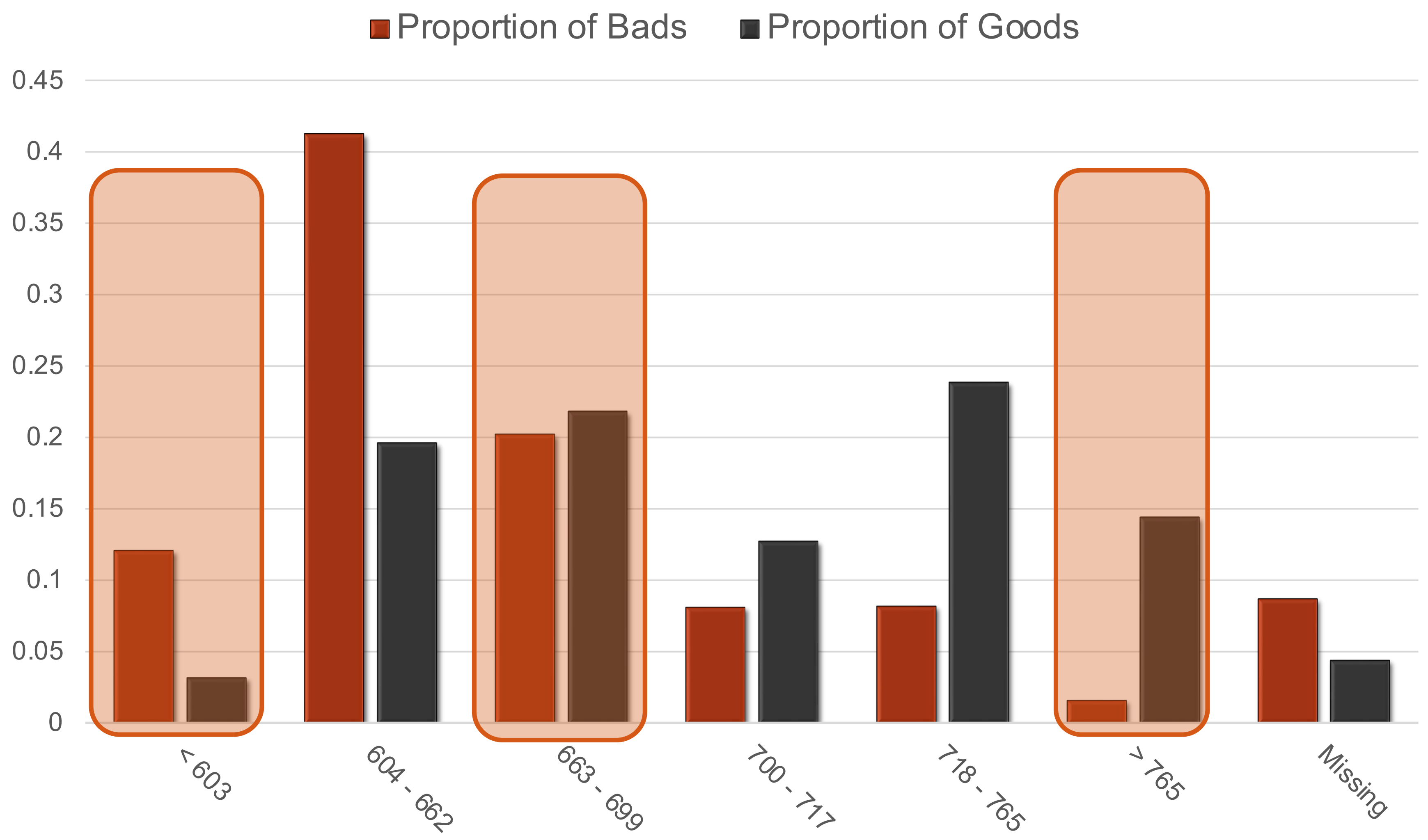

Proportion of Bads
Proportion of Goods
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
< 603
604 - 662
663 - 699
700 - 717
718 - 765
> 765
Missing

WOE – Example

All the left hand groups "Good" will sum to 1. ALL the right hand groups will sum to 1.



WOE – Example


Proportion of Bads
Proportion of Goods
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
< 603
604 - 662
663 - 699
700 - 717
718 - 765
> 765
Missing

WOE – Example

Event is Deafaulted so " Bad"

WOE for Bureau Score				
Group	Values	Event Count	Non-event Count	WOE
1	< 603	111	112	**-1.32**
2	604 – 662	378	678	
3	663 – 699	185	754	
4	700 – 717	74	440	
5	718 – 765	75	824	
6	> 765	15	498	
7	MISSING	80	153	
Total		**918**	**3,459**	

$$Dist.Good_1 = \frac{112}{3459}$$

$$= 0.032$$

$$Dist.Bad_1 = \frac{111}{918}$$

$$= 0.121$$

NOT BASE 10, it is base e

$$WOE_1 = \log\left(\frac{0.032}{0.121}\right)$$

$$= -1.32$$

WOE – Example

WOE for Bureau Score				
Group	Values	Event Count	Non-event Count	WOE
1	< 603	111	112	**-1.32**
2	604 – 662	378	678	**-0.74**
3	663 – 699	185	754	**0.08**
4	700 – 717	74	440	**0.46**
5	718 – 765	75	824	**1.07**
6	> 765	15	498	**2.18**
7	MISSING	80	153	**-0.68**
Total		**918**	**3,459**	

Higher the number. higher the evidence. 0 means both proporiton equal, ratio is 1.

Weight of Evidence (WOE)

- WOE measures the strength of the attributes of a characteristic in **separating good and bad accounts**.

$$WOE_i = \log\left(\frac{Dist.Good_i}{Dist.Bad_i}\right)$$

- WOE approximately zero implies what?

Weight of Evidence (WOE)

- WOE measures the strength of the attributes of a characteristic in **separating good and bad accounts**.

$$WOE_i = \log\left(\frac{Dist.Good_i}{Dist.Bad_i}\right)$$

- WOE approximately zero implies % good approximately equal to % bad so group doesn’t separate good vs. bad well.

WOE 0 means no evidence to separate Good from Bad.

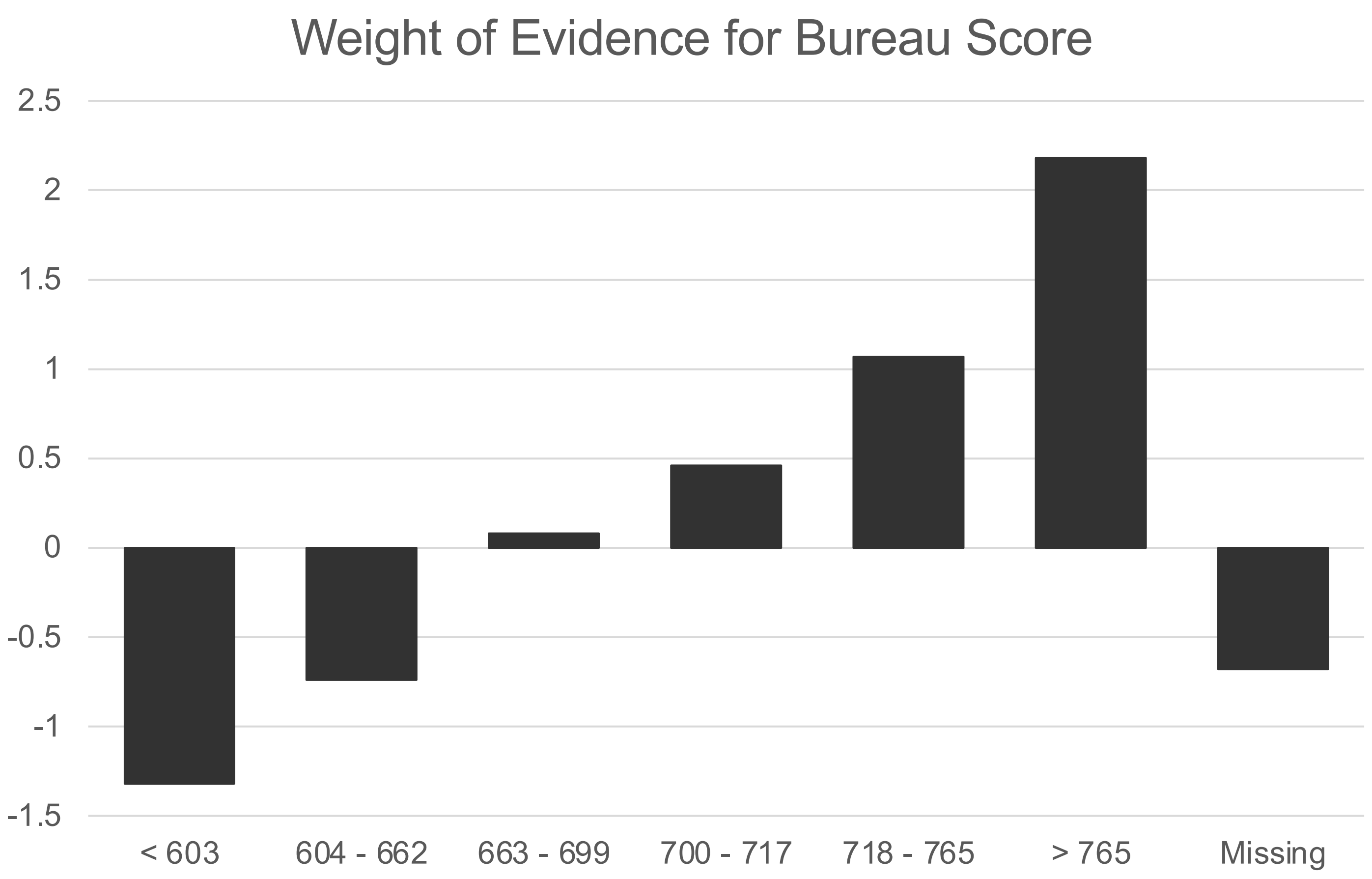
Weight of Evidence (WOE)

- WOE measures the strength of the attributes of a characteristic in **separating good and bad accounts**.

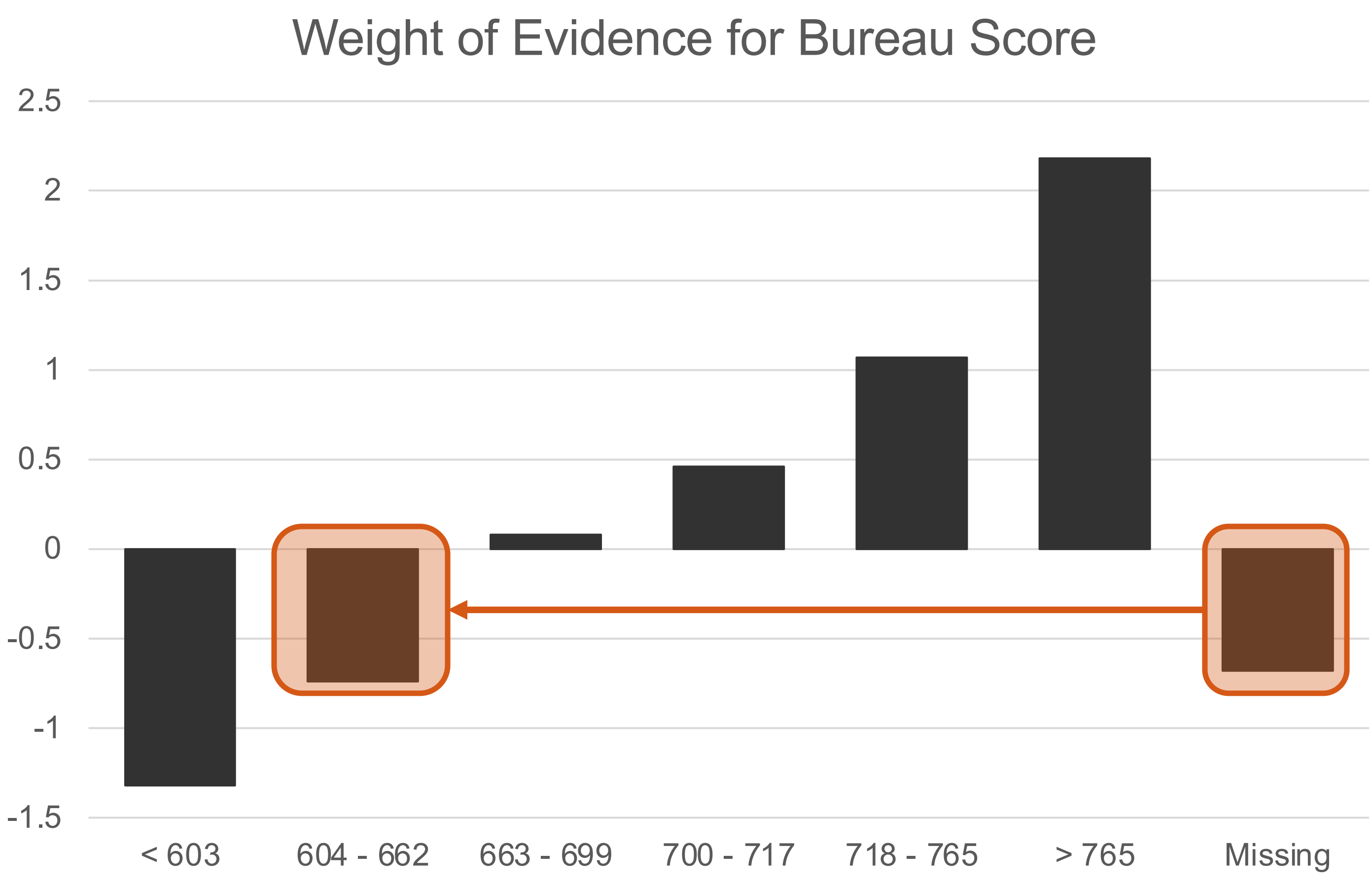
$$WOE_i = \log\left(\frac{Dist.Good_i}{Dist.Bad_i}\right)$$

- WOE approximately zero implies % good approximately equal to % bad so group doesn’t separate good vs. bad well.
- WOE positive implies group identifies people who are good.
- WOE negative implies group identifies people who are bad.

WOE – Example


Weight of Evidence for Bureau Score
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
< 603
604 - 662
663 - 699
700 - 717
718 - 765
> 765
Missing

WOE – Example


Weight of Evidence for Bureau Score
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
< 603
604 - 662
663 - 699
700 - 717
718 - 765
> 765
Missing

df

WOE – R

smbinning is a function and a package. Takes df, y=target variable. One downside is notice how variable is called good 1 on top of weight of eveidence calc and 0 on bottom (1 is numerator). I need variable that flags 1 as bad, 0s as good. Notice it is variable name in quotes as y and x. Then it finds cuts for you and also

numerator column name is good

```r
result <- smbinning(df = train, y = "good", x = "bureau_score")
result$ivtable
```

name of column

```
##    Cutpoint CntRec CntGood CntBad CntCumRec CntCumGood CntCumBad PctRec
## 1    <= 603    223     112    111       223        112       111 0.0509
## 2    <= 662   1056     678    378      1279        790       489 0.2413
## 3    <= 699    939     754    185      2218       1544       674 0.2145
## 4    <= 717    514     440     74      2732       1984       748 0.1174
## 5    <= 765    899     824     75      3631       2808       823 0.2054
## 6     > 765    513     498     15      4144       3306       838 0.1172
## 7   Missing    233     153     80      4377       3459       918 0.0532
## 8     Total   4377    3459    918        NA         NA        NA 1.0000

##    GoodRate BadRate    Odds LnOdds     WoE     IV
## 1    0.5022  0.4978  1.0090 0.0090 -1.3176 0.1167
## 2    0.6420  0.3580  1.7937 0.5843 -0.7423 0.1602
## 3    0.8030  0.1970  4.0757 1.4050  0.0785 0.0013
## 4    0.8560  0.1440  5.9459 1.7827  0.4562 0.0213
## 5    0.9166  0.0834 10.9867 2.3967  1.0701 0.1675
## 6    0.9708  0.0292 33.2000 3.5025  2.1760 0.2777
## 7    0.6567  0.3433  1.9125 0.6484 -0.6781 0.0291
## 8    0.7903  0.2097  3.7680 1.3265  0.0000 0.7738
```

only thing you care about is cut points and weight of evidence.

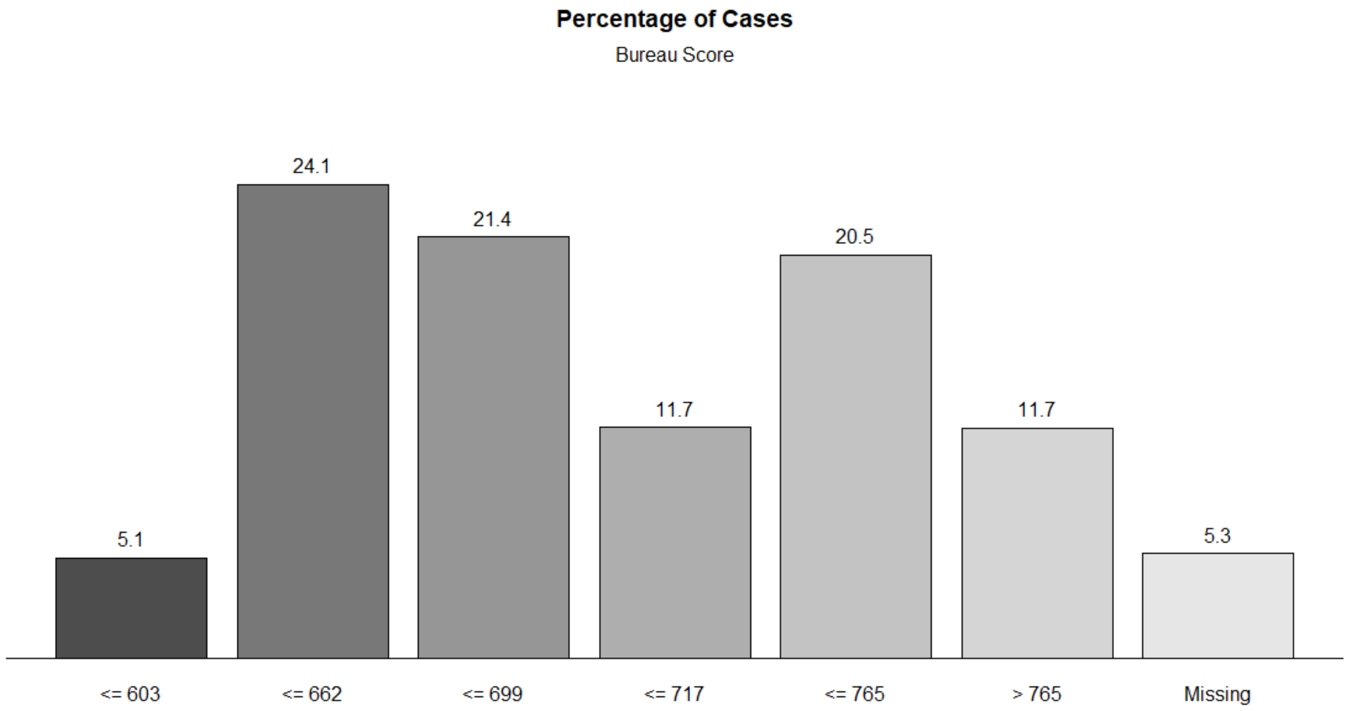
WOE – R

```
result$cut

## [1] 603 662 699 717 765

result$iv

## [1] 0.7738
```

WOE – R

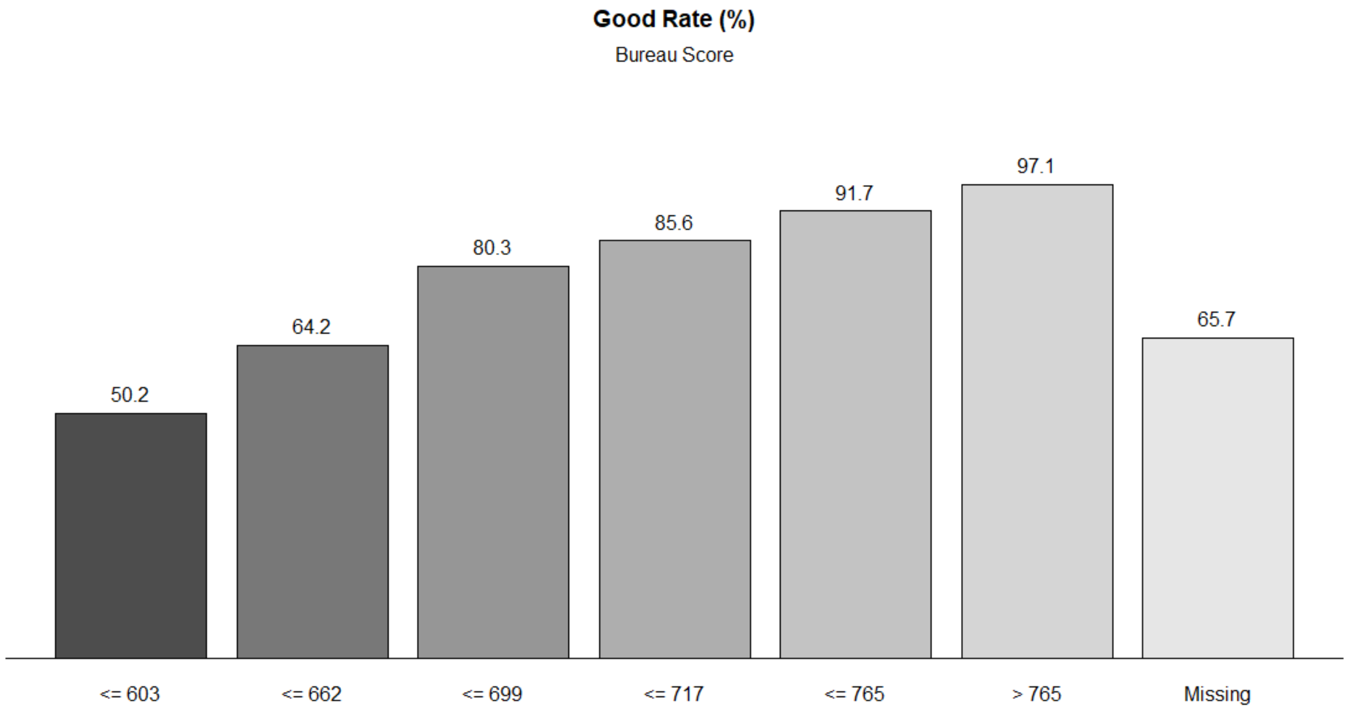
distributions

```
smbinning.plot(result, option = "dist", sub = "Bureau Score")
```



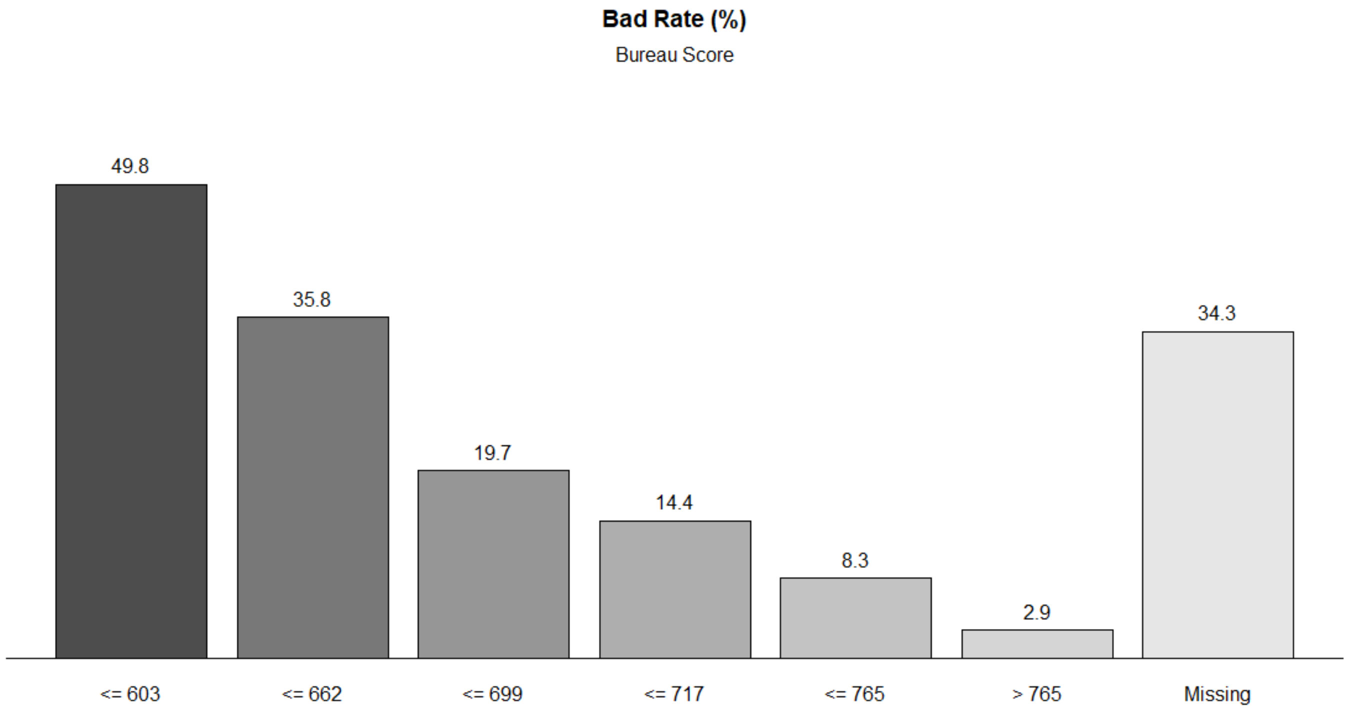
WOE – R

```
smbinning.plot(result, option = "goodrate", sub = "Bureau Score")
```



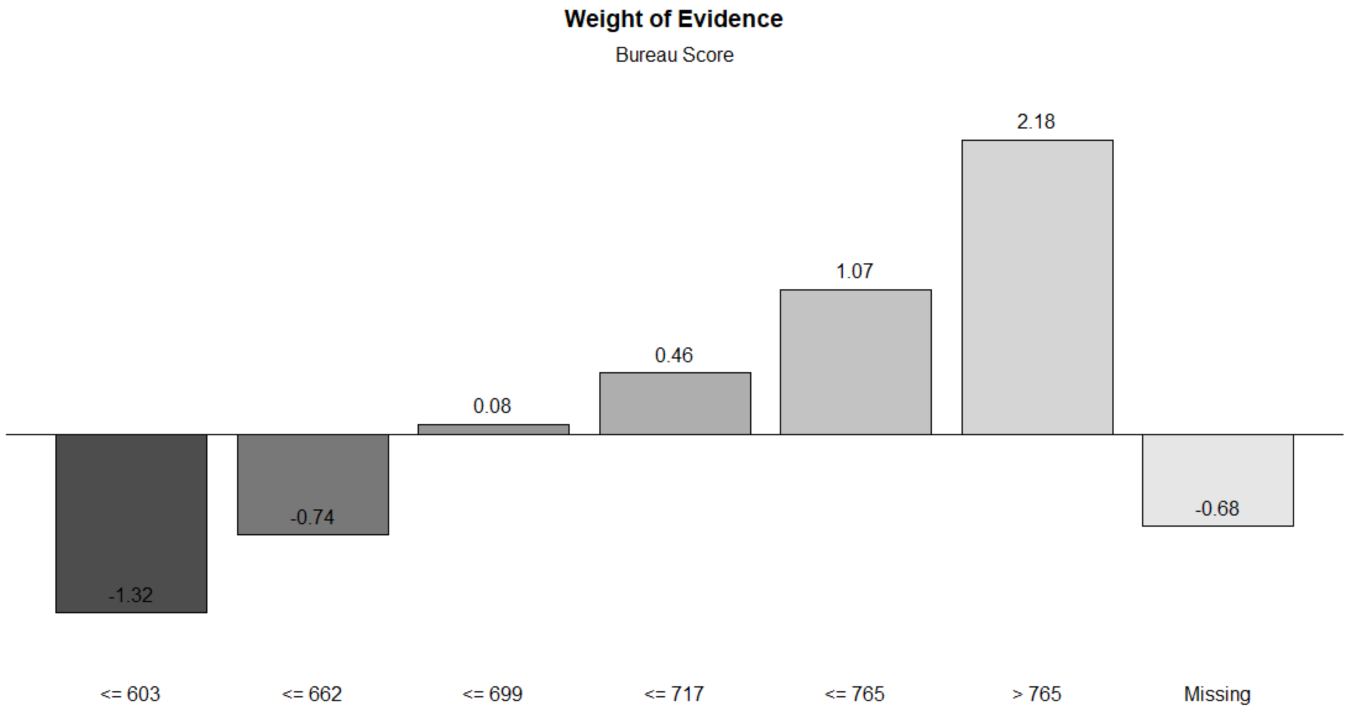
WOE – R

```
smbinning.plot(result, option = "badrate", sub = "Bureau Score")
```



WOE – R

```
smbinning.plot(result, option = "WoE", sub = "Bureau Score")
```



WOE – R

sm binning will not ocmbine existing categories inside a variable. Like it thinks not to touch categorical variable that already has bins. It is always looking for missing.

```
result <- smbinning.factor(df = train, y = "good", x = "purpose")
result$ivtable
```

numerator, good is column name

col name

```
##     Cutpoint CntRec CntGood CntBad CntCumRec CntCumGood CntCumBad PctRec
## 1 = 'LEASE'    1466    1149    317      1466       1149       317 0.3349
## 2  = 'LOAN'    2911    2310    601      4377       3459       918 0.6651
## 3    Missing      0       0      0      4377       3459       918 0.0000
## 4      Total   4377    3459    918        NA         NA        NA 1.0000

##   GoodRate BadRate   Odds LnOdds     WoE     IV
## 1   0.7838  0.2162 3.6246 1.2877 -0.0388 0.0005
## 2   0.7935  0.2065 3.8436 1.3464  0.0199 0.0003
## 3      NaN     NaN    NaN    NaN     NaN    NaN
## 4   0.7903  0.2097 3.7680 1.3265  0.0000 0.0008
```

Separation Issues Remain

This should be found in exploration phase

- Quasi-complete separation still a problem:

	Non-Event	Event	WOE
A	28	7	**-0.032**
B	16	0	∞
C	94	11	**0.728**
D	23	21	**-1.327**
Total	161	39	

happens because denom is 0, so to fix just add a small value

Adjusted WOE

- Adjust the WOE calculation to account for possible quasi-complete separation:

$$Adjusted\ WOE_i = \log\left(\frac{Dist.Good_i + \eta_1}{Dist.Bad_i + \eta_2}\right)$$

- The η_1 and η_2 parameters are smoothing parameters that correct for potential overfitting and also protect against quasi-complete separation.
- Most software just sets $\eta_1 = \eta_2$ and has one parameter.

Adjusted WOE $(\eta_1 = \eta_2 = 0.005)$

- Quasi-complete separation no longer a problem:

	Non-Event	Event	WOE
A	28	7	**-0.031**
B	16	0	**3.039**
C	94	11	**0.719**
D	23	21	**-1.302**
Total	**161**	**39**	

Smoothed WOE (SWOE)

- SAS has recently proposed a slightly different smoothed version of the WOE calculation to account for possible quasi-complete separation:

$$SWOE_i = \log\left(\frac{\#Bad_i + (Overall\ Prop.Bad)\times c}{\#Good_i + (Overall\ Prop.Good)\times c}\right)$$

- This is just a smoothing parameter put in a slightly different place in the WOE calculation based on more Bayesian inference techniques.
- Haven’t seen it really used elsewhere.

INFORMATION VALUE

how good ALL variables are at predicting, used for ranking imp variable. Get 1 # for each varibale. then put them all in 1 table.

Higher the # better at predicting Good/bad

Information Value (IV)

Uses WOE

looks at all 20 variables

- How big is a "big" difference when looking across groups for WOE?
- IV measures the ability of the characteristic to separate goods vs. bads.

entire variable

$$IV = \sum_{i=1}^{L} (Dist.Good_i - Dist.Bad_i) \times \log\left(\frac{Dist.Good_i}{Dist.Bad_i}\right)$$

IC is the sum of all groups, diff bet Good/Bad distr times WOE

Weight of Evidence

multiplying to make it positive

Information Value (IV)

- How big is a “big” difference when looking across groups for WOE?
- IV measures the ability of the characteristic to separate goods vs. bads.

$$IV = \sum_{i=1}^{L} (Dist.Good_i - Dist.Bad_i) \times \boxed{\log\left(\frac{Dist.Good_i}{Dist.Bad_i}\right)}$$

Weight of Evidence!

Information Value (IV)

- How big is a "big" difference when looking across groups for WOE?
- IV measures the ability of the characteristic to separate goods vs. bads.

$$IV = \sum_{i=1}^{L} (Dist.Good_i - Dist.Bad_i) \times \log\left(\frac{Dist.Good_i}{Dist.Bad_i}\right)$$

- Used to select characteristics with strong predictive value.

Information Value (IV)

banks use IV to include or exclude variables in model

- Characteristics of IV:
  - $IV \geq 0$
  - Bigger is Better!

- Rules of Thumb:
  - IV < 0.02 – Not predictive
  - 0.02 < IV < 0.1 – Weak predictor
  - 0.1 < IV < 0.25 – Medium predictor
  - 0.25 < IV – Strong predictor

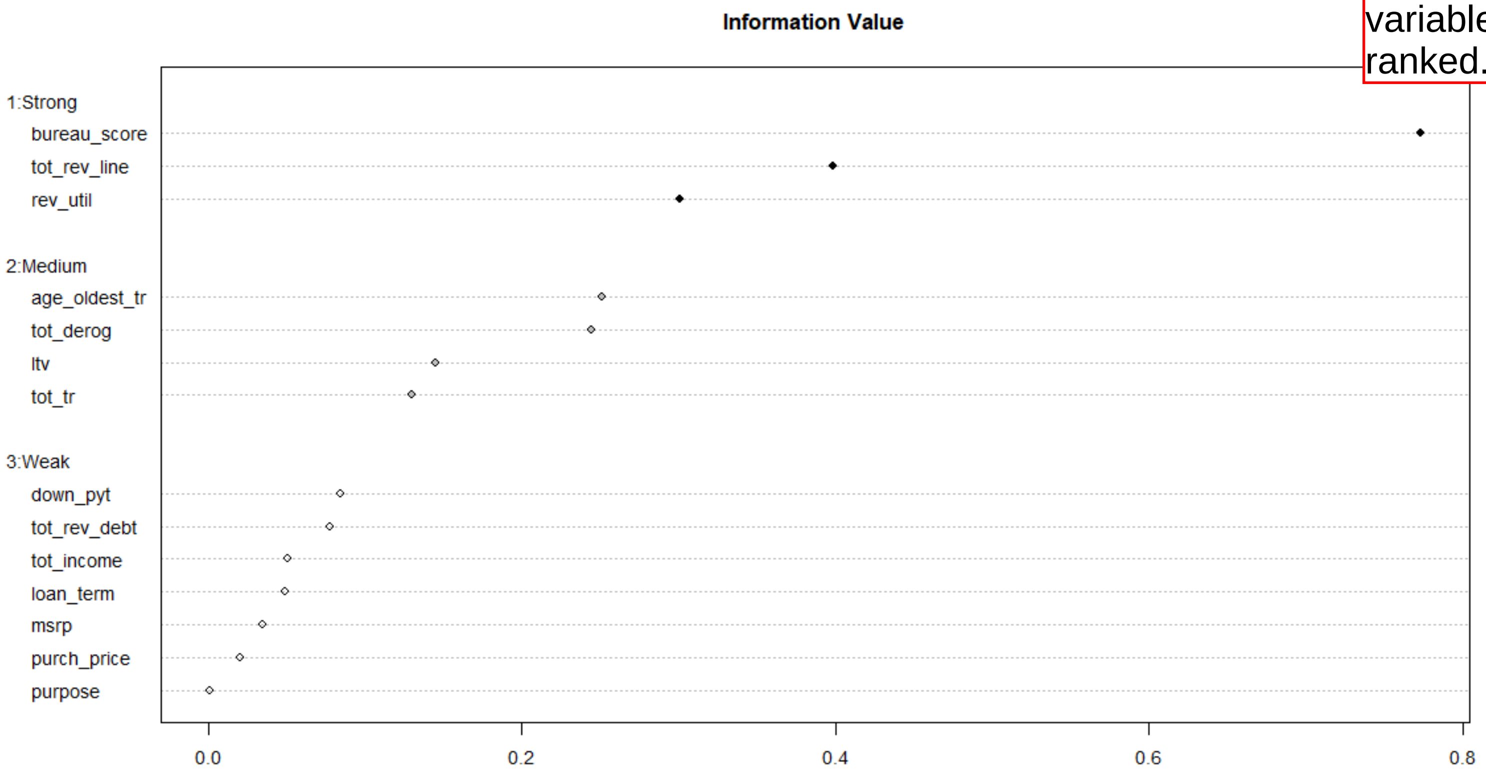
These are banking rules of thumbs, in banking dont like seeing above 0.5. it implies over predicting. Bureau scores already have greater than 0.5. I have already given them loans, how do you think i gave them loans? I used credit scores to give them loan so variable credit scores will be good already - looping kinda variable..HOW DID YOU MKAE THAT ORIGINAL DECISION?

IN scenarios like this, we build 2 models, 1 with bureau score and 1 without. Then we ensemble together. This is a banking construct.

Information Value (IV) – R

```
iv_summary <- smbinning.sumiv(df = train, y = "good")

smbinning.sumiv.plot(iv_summary)
```

everything else it assumes predictor variables. Give you imp variables ranked.



Information Value (IV) – R

dont like >0.5 cuz over corelate bureau score

smbinning by default looks at all variables , anything numeric it tries to split with CIT ie chi square test to split/bin numeric variable. So ensure variables coded 0 and 1 are factors, that way doesnt cause issues.

```
iv_summary

##               Char     IV               Process
## 12  bureau_score 0.7738    Numeric binning OK
## 10  tot_rev_line 0.3987    Numeric binning OK
## 11      rev_util 0.3007    Numeric binning OK
## 6  age_oldest_tr 0.2512    Numeric binning OK
## 4      tot_derog 0.2443    Numeric binning OK
## 19           ltv 0.1454    Numeric binning OK
## 5         tot_tr 0.1304    Numeric binning OK
## 15      down_pyt 0.0848    Numeric binning OK
## 9   tot_rev_debt 0.0782    Numeric binning OK
## 20    tot_income 0.0512    Numeric binning OK
## 17     loan_term 0.0496    Numeric binning OK
## 14          msrp 0.0353    Numeric binning OK
## 13   purch_price 0.0204    Numeric binning OK
## 16       purpose 0.0008     Factor binning OK
## 1     bankruptcy     NA    Uniques values < 5
## 2            bad     NA    Uniques values < 5
## 3         app_id     NA No significant splits
## 7    tot_open_tr     NA No significant splits
## 8     tot_rev_tr     NA No significant splits
## 18      loan_amt     NA No significant splits
## 21      used_ind     NA    Uniques values < 5
## 22        weight     NA    Uniques values < 5
```

no probs with this variable.

will not bin if numeric variables if less than 5 unique value.s

dont worru about this one <5.

no sig splits means numeric variable that it could not find statistic relationship with your target variable statistically. Loan amt had 0 predictive power.

Information Value (IV) – R

```
iv_summary

##                Char     IV                  Process
## 12   bureau_score 0.7738     Numeric binning OK
## 10   tot_rev_line 0.3987     Numeric binning OK
## 11       rev_util 0.3007     Numeric binning OK
## 6   age_oldest_tr 0.2512     Numeric binning OK
## 4       tot_derog 0.2443     Numeric binning OK
## 19            ltv 0.1454     Numeric binning OK
## 5          tot_tr 0.1304     Numeric binning OK
## 15       down_pyt 0.0848     Numeric binning OK
## 9    tot_rev_debt 0.0782     Numeric binning OK
## 20     tot_income 0.0512     Numeric binning OK
## 17      loan_term 0.0496     Numeric binning OK
## 14           msrp 0.0353     Numeric binning OK
## 13     purch_price 0.0204     Numeric binning OK
## 16        purpose 0.0008      Factor binning OK
## 1      bankruptcy     NA     Uniques values < 5
## 2             bad     NA     Uniques values < 5
## 3          app_id     NA No significant splits
## 7     tot_open_tr     NA No significant splits
## 8      tot_rev_tr     NA No significant splits
## 18       loan_amt     NA No significant splits
## 21       used_ind     NA     Uniques values < 5
## 22         weight     NA     Uniques values < 5
```

Information Value (IV) – R

```
iv_summary

##               Char     IV               Process
## 12  bureau_score 0.7738    Numeric binning OK
## 10  tot_rev_line 0.3987    Numeric binning OK
## 11      rev_util 0.3007    Numeric binning OK
## 6  age_oldest_tr 0.2512    Numeric binning OK
## 4      tot_derog 0.2443    Numeric binning OK
## 19           ltv 0.1454    Numeric binning OK
## 5         tot_tr 0.1304    Numeric binning OK
## 15      down_pyt 0.0848    Numeric binning OK
## 9   tot_rev_debt 0.0782    Numeric binning OK
## 20    tot_income 0.0512    Numeric binning OK
## 17     loan_term 0.0496    Numeric binning OK
## 14          msrp 0.0353    Numeric binning OK
## 13    purch_price 0.0204   Numeric binning OK
## 16       purpose 0.0008     Factor binning OK
## 1     bankruptcy     NA    Uniques values < 5
## 2            bad     NA    Uniques values < 5
## 3         app_id     NA No significant splits
## 7    tot_open_tr     NA No significant splits
## 8     tot_rev_tr     NA No significant splits
## 18      loan_amt     NA No significant splits
## 21      used_ind     NA    Uniques values < 5
## 22        weight     NA    Uniques values < 5
```

sm binning will not bin numerical variable if it has less than 5 variables. less than 5 is basically a categorical variable

Information Value (IV) – R

```
iv_summary

##                Char     IV               Process
## 12  bureau_score 0.7738    Numeric binning OK
## 10  tot_rev_line 0.3987    Numeric binning OK
## 11      rev_util 0.3007    Numeric binning OK
## 6  age_oldest_tr 0.2512    Numeric binning OK
## 4      tot_derog 0.2443    Numeric binning OK
## 19           ltv 0.1454    Numeric binning OK
## 5         tot_tr 0.1304    Numeric binning OK
## 15      down_pyt 0.0848    Numeric binning OK
## 9   tot_rev_debt 0.0782    Numeric binning OK
## 20    tot_income 0.0512    Numeric binning OK
## 17     loan_term 0.0496    Numeric binning OK
## 14          msrp 0.0353    Numeric binning OK
## 13    purch_price 0.0204    Numeric binning OK
## 16       purpose 0.0008     Factor binning OK
## 1     bankruptcy     NA    Uniques values < 5
## 2            bad     NA    Uniques values < 5
## 3         app_id     NA No significant splits
## 7    tot_open_tr     NA No significant splits
## 8     tot_rev_tr     NA No significant splits
## 18      loan_amt     NA No significant splits
## 21      used_ind     NA    Uniques values < 5
## 22        weight     NA    Uniques values < 5
```

this means this is numeric variable that could not find any statistical relationship with target variable.

sm binning have a shot at variable selection if you have cont variables

Information Value (IV)

- Characteristics of IV:
  - $IV \geq 0$
  - Bigger is Better!

- Rules of Thumb:
  - IV < 0.02 – Not predictive
  - 0.02 < IV < 0.1 – Weak predictor
  - 0.1 < IV < 0.25 – Medium predictor
  - 0.25 < IV < 0.5 – Strong predictor
  - IV > 0.5 – Over-predicting?

Information Value (IV)

- Rules of Thumb:
  - IV < 0.02 – Not predictive
  - 0.02 < IV < 0.1 – Weak predictor
  - 0.1 < IV < 0.25 – Medium predictor
  - 0.25 < IV < 0.5 – Strong predictor
  - IV > 0.5 – Over-predicting?
- Over-predicting Example:
  - All previous mortgage decisions have been made only on bureau score so of course bureau score is highly predictive – becomes only significant variable!
  - Create two models – one with bureau score, one without bureau score and **ensemble**.

?

GINI STATISTIC

Gini Statistic

- **Gini statistic** is optional technique that tries to answer the same question as Information Value – which variables are strong enough to enter the scorecard model?
- IV is more in line with WOE calculation and used more often.
- Characteristics:
  - Range is 0 to 100.
  - Bigger is Better.

Gini Statistic Calculation

- More complicated technique for trying to evaluate how characteristics separate good from bad.
- Majority of the time Gini and IV agree, but could be different on the borderline cases.
- Calculation:
  - Sort *L* groups of variable by descending order of the proportion of all events.

$$Gini = \left(1 - \frac{\left(2\sum_{i=2}^{L}\left(n_{i,E} \times \sum_{i=1}^{i-1} n_{i,NE}\right) + \sum_{i=1}^{L}\left(n_{i,E} \times n_{i,NE}\right)\right)}{N_E \times N_{NE}}\right) \times 100$$

Gini Statistic Calculation

- More complicated technique for trying to evaluate how characteristics separate good from bad.
- Majority of the time Gini and IV agree, but could be different on the borderline cases.
- Calculation:
  - Sort *L* groups of variable by descending order of the proportion of all events.

$$Gini = \left(1 - \frac{\left(2\sum_{i=2}^{L}\left(n_{i,E} \times \sum_{i=1}^{i-1} n_{i,NE}\right) + \sum_{i=1}^{L}\left(n_{i,E} \times n_{i,NE}\right)\right)}{N_E \times N_{NE}}\right) \times 100$$

Number of events in group *i*

Gini Statistic Calculation

- More complicated technique for trying to evaluate how characteristics separate good from bad.
- Majority of the time Gini and IV agree, but could be different on the borderline cases.
- Calculation:
  - Sort *L* groups of variable by descending order of the proportion of all events.

$$Gini = \left(1 - \frac{\left(2\sum_{i=2}^{L}\left(n_{i,E} \times \sum_{i=1}^{i-1} n_{i,NE}\right) + \sum_{i=1}^{L}\left(n_{i,E} \times n_{i,NE}\right)\right)}{N_E \times N_{NE}}\right) \times 100$$

Number of non-events in group *i*

Gini Statistic Calculation

- More complicated technique for trying to evaluate how characteristics separate good from bad.
- Majority of the time Gini and IV agree, but could be different on the borderline cases.
- Calculation:
  - Sort *L* groups of variable by descending order of the proportion of all events.

$$Gini = \left(1 - \frac{\left(2\sum_{i=2}^{L}\left(n_{i,E} \times \sum_{i=1}^{i-1} n_{i,NE}\right) + \sum_{i=1}^{L}\left(n_{i,E} \times n_{i,NE}\right)\right)}{N_E \times N_{NE}}\right) \times 100$$

Total number of events and non-events

PROC BINNING IN SAS VIYA

WOE – SAS Viya

```
proc binning data = public.train method = tree(initbin = 100
                maxnbins = 10);
    target bad / level = int;
    input bureau_score / level = int;
    ods output BinDetails = bincuts VarTransInfo = bincount;
run;
```

WOE – SAS Viya

Bin Details										
Variable	Bin ID	Lower Bound	Upper Bound	Bin Width	N Levels	Number of Observations	Mean	Standard Deviation	Minimum	Maximum
bureau_score										

WOE – SAS Viya

Transformation Information					
Variable	N	N Miss	N Bins	Importance	Relative Importance
bureau_score					

WOE – SAS Viya

```
data _null_;
   set bincount;
   call symput('numbin', Nbins - 1);
run;

proc sql;
   select Max
      into :cuts separated by ' '
      from bincuts(firstobs = 2 obs = &numbin);
quit;

proc binning data = public.train numbin = &numbin
             method=cutpts(&cuts) woe;
   target bad / event = '1';
   input bureau_score / level = int;
run;
```

WOE – SAS Viya

Bin Details							
Variable	Bin ID	Lower Bound	Upper Bound	Bin Width	Number of Observations	Mean	Standard Deviation
bureau_score							

WOE – SAS Viya

Bin Details						
Variable	Bin ID	Minimum	Maximum	Event Count	Weight of Evidence	Information Value
bureau_score						

WOE – SAS Viya

Variable Information Value	
Variable	**Information Value**
bureau_score	

WOE – SAS Viya

```
proc tabulate data=public.train out=facwoe;
   class bad purpose;
   table purpose, bad*colpctn / rts=10;
run;

proc transpose data = facwoe out = facwoe2(rename=
                                   (col1 = bad0 col2 = bad1));
   var PctN_10;
   by purpose;
run;

data facwoe2;
   set facwoe2;
   WOE = log(bad1/bad0);
run;
```

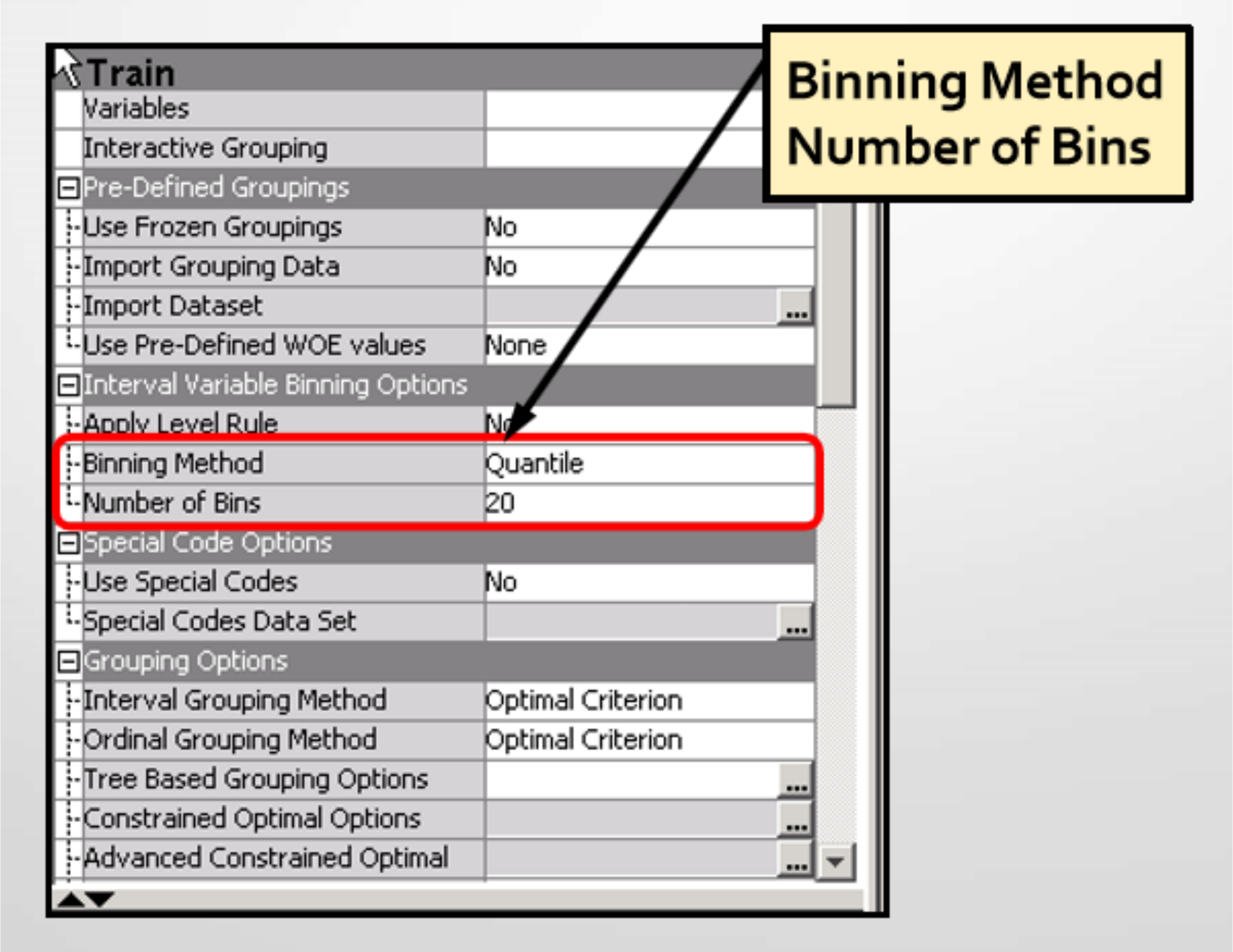
WOE – SAS Viya

	bad	
	0	1
	ColPctN	ColPctN
purpose		
LEASE		
LOAN		

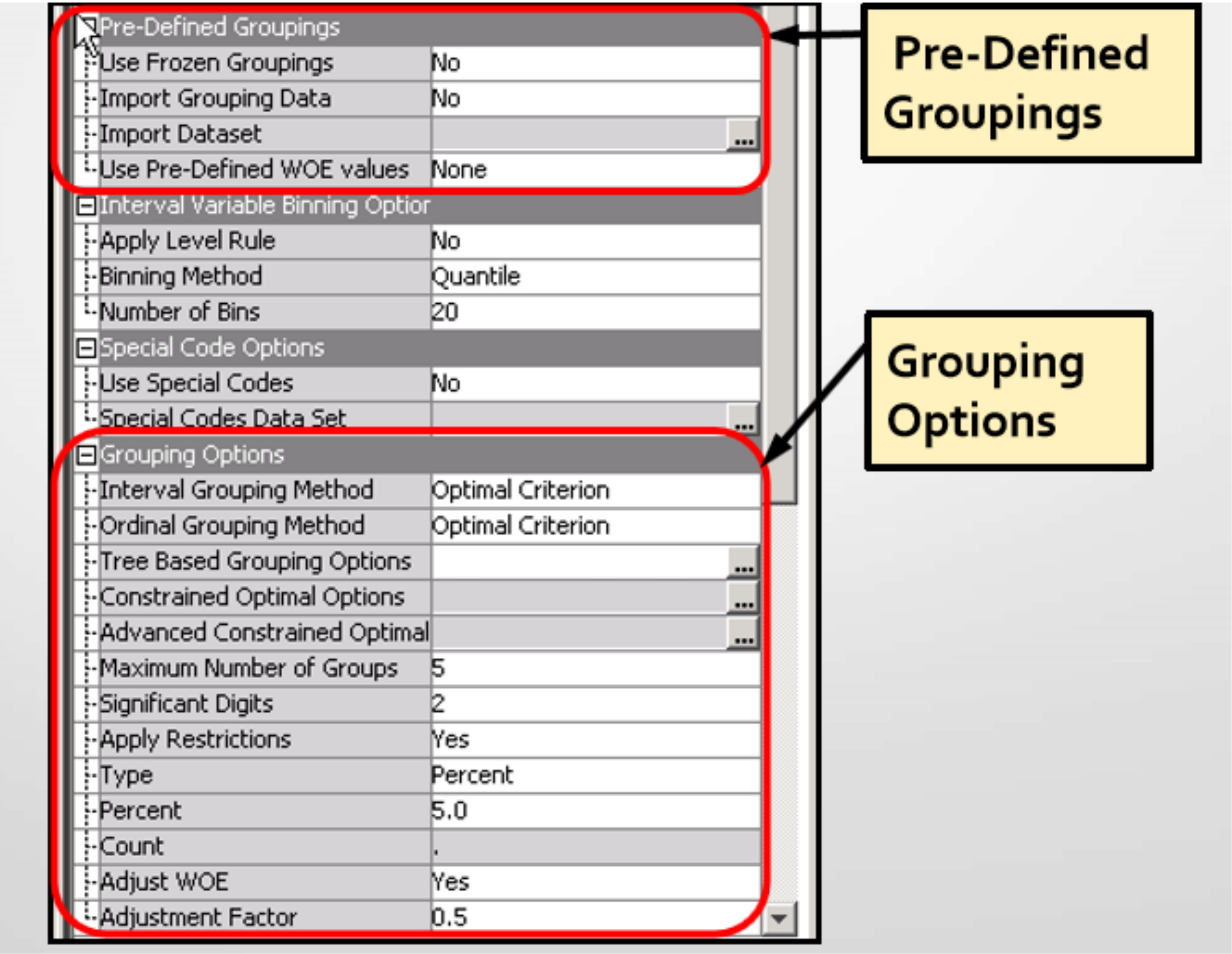
Obs	purpose	_NAME_	bad0	bad1	WOE
1					
2					

INTERACTIVE GROUPING NODE IN SAS EM

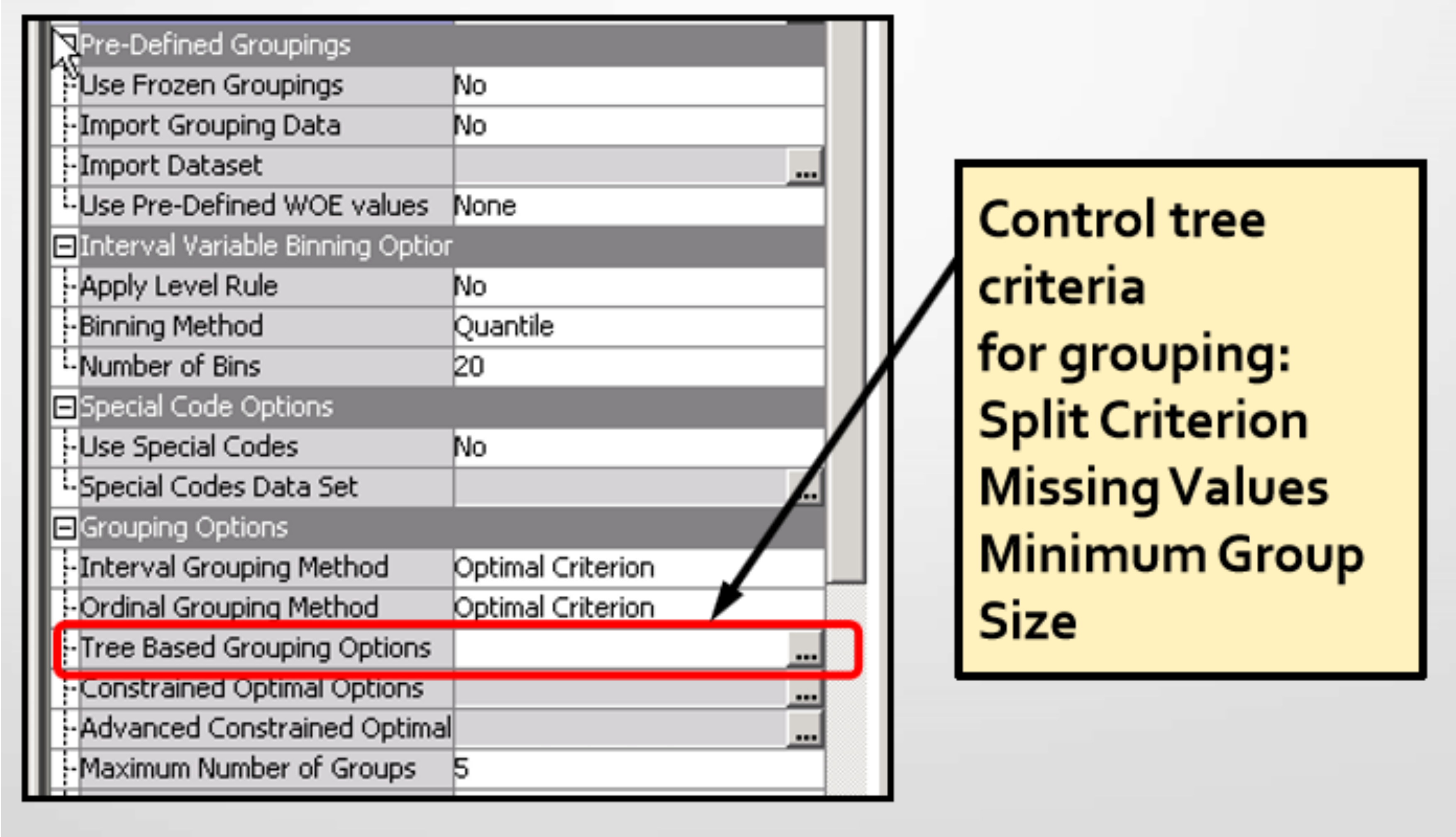
Pre-Binning of the Interval Variables



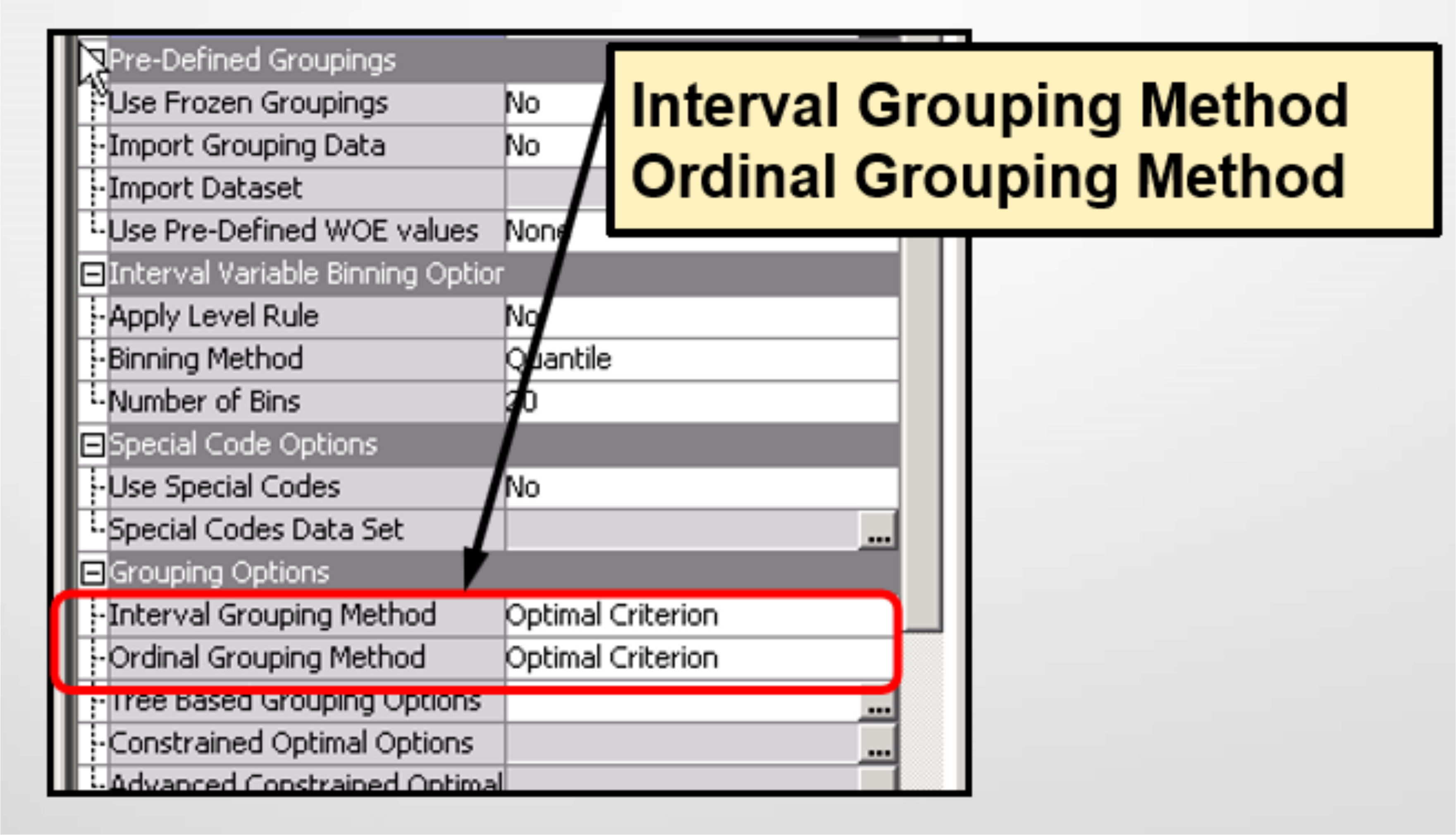
Grouping Options



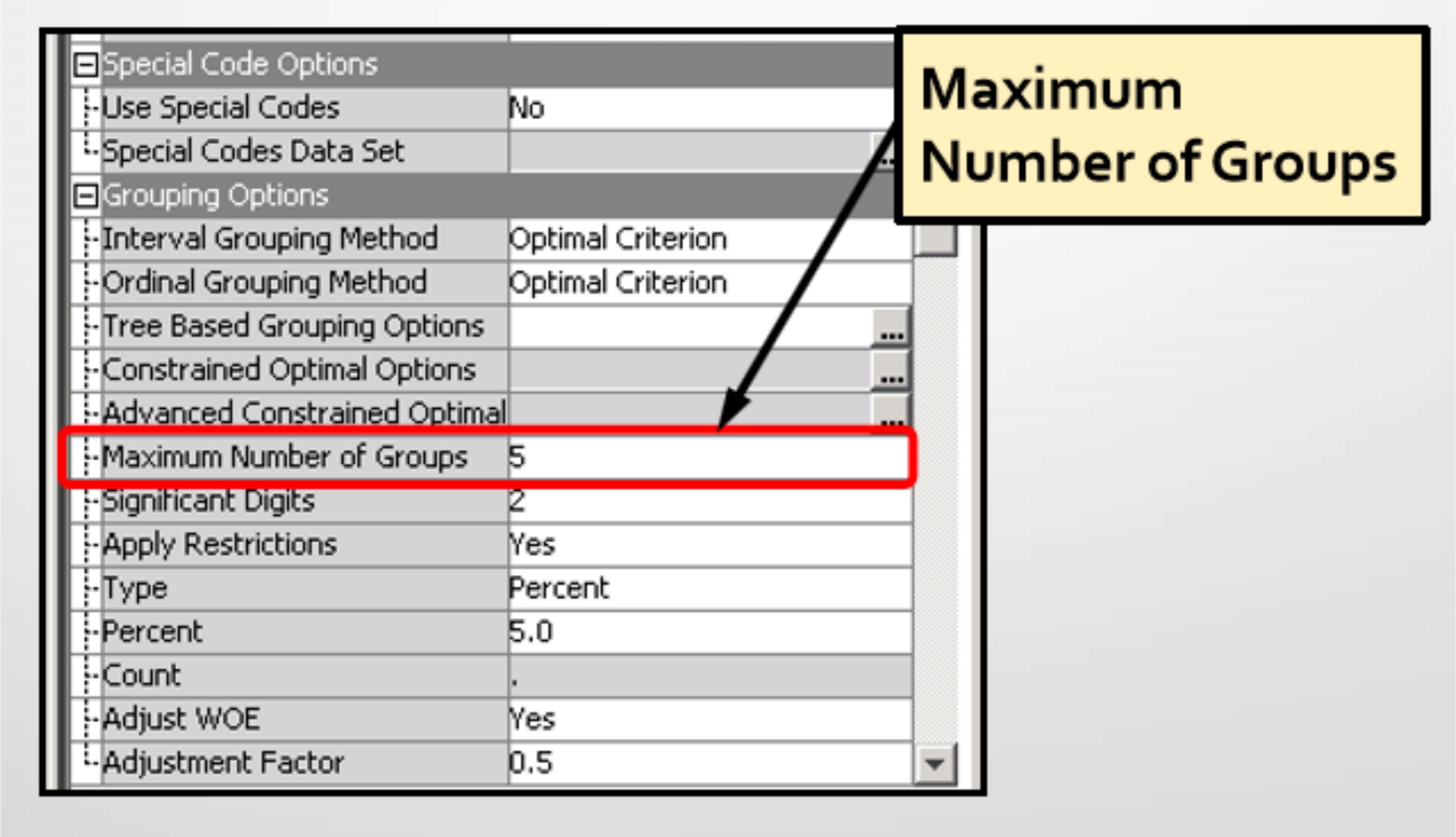
Grouping Options: Tree Criteria



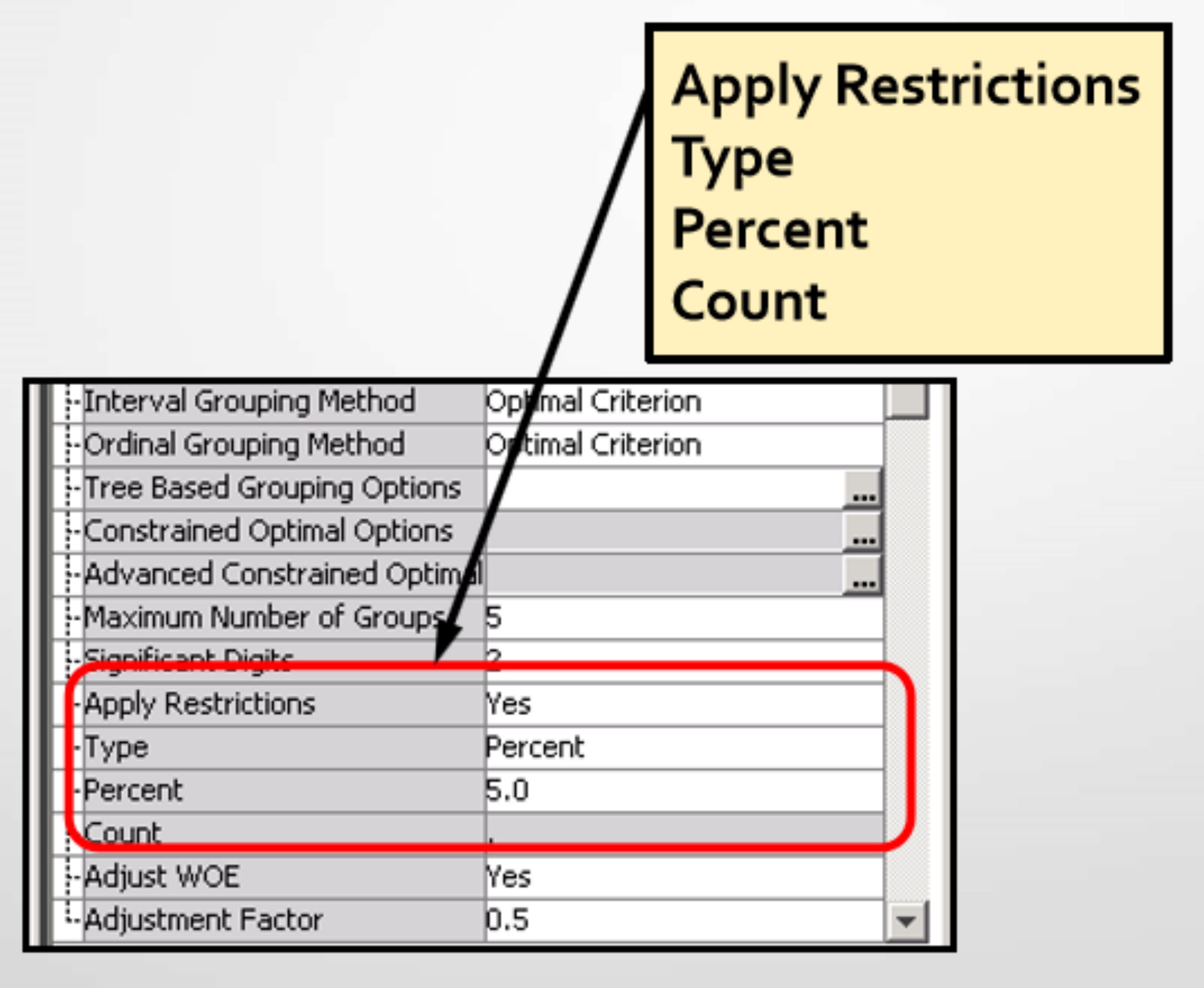
Grouping Options: Interval vs. Ordinal



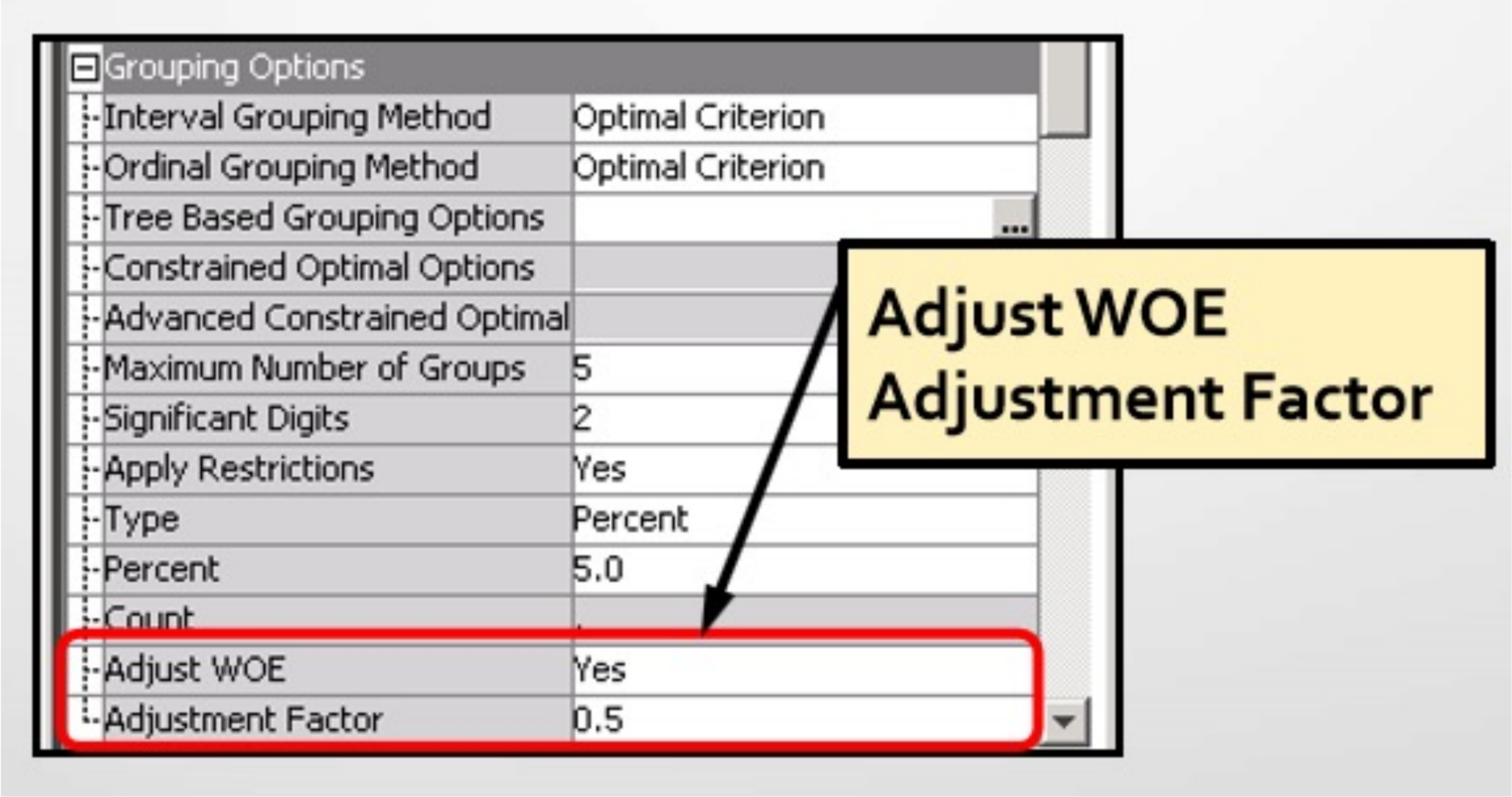
Grouping Options: Number of Groups



Grouping Options: Stopping Rules



Grouping Options: WOE Adjustments



SCORECARD CREATION

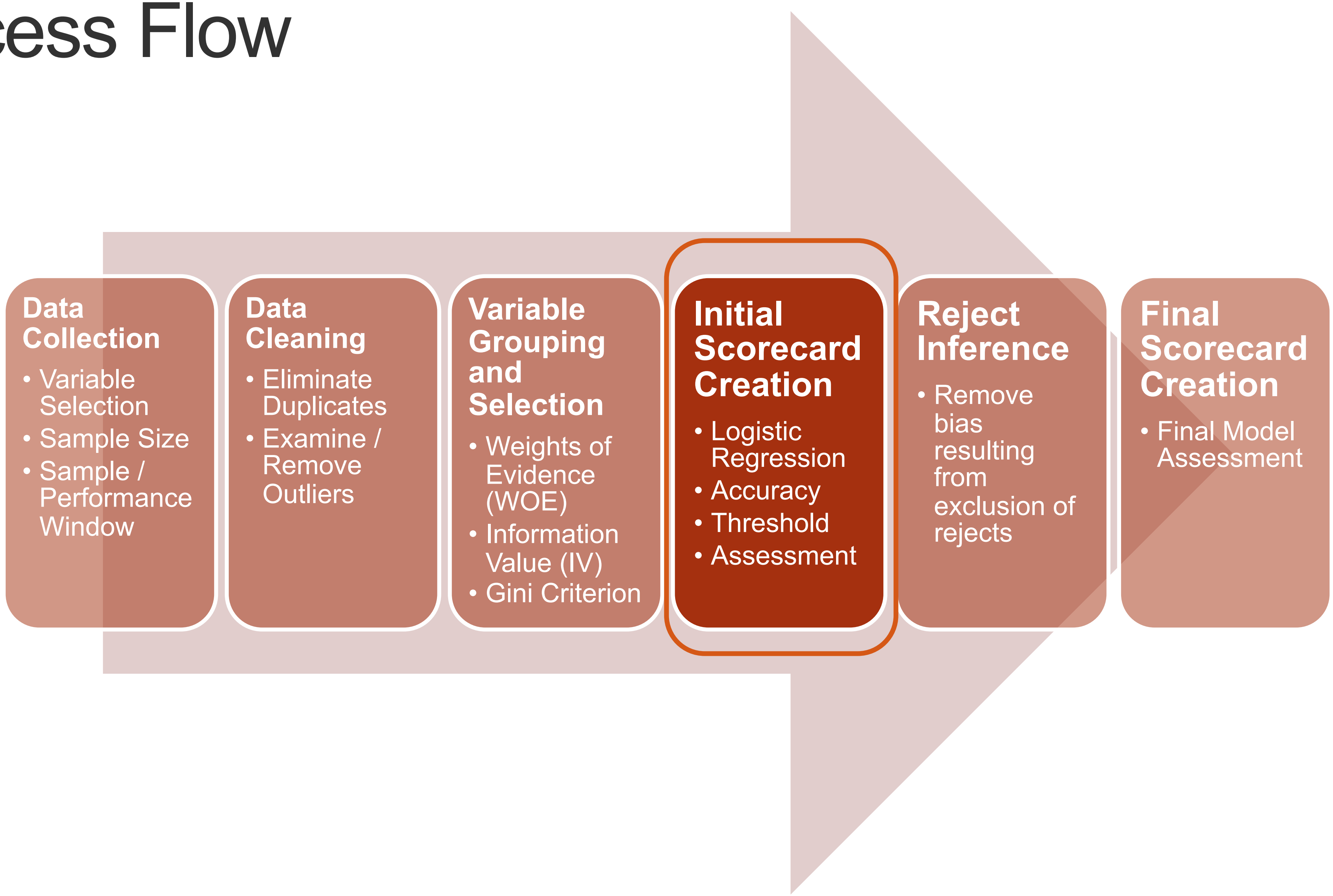
Dr. Aric LaBarr

Institute for Advanced Analytics

can do some variable selection with IVs,

Binning individidual variables

Process Flow



INITIAL SCORECARD CREATION

Initial Scorecard Model

- The scorecard is (typically) based on a logistic regression model:

$$logit(p) = \log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k$$

- *p* is the posterior probability of default (PD) given the inputs.

Blasphemy!!!

- Wait, so I'm going through all that math just to throw things back into the logistic regression I was trying to avoid in the first place?!?!?!?!

Different Inputs

- Instead of using the original variables for the model, scorecard models have the binned variables as their foundation.

1 defaulted

predictor

put woe instead of binned variable in logistic reg

Observation	Target	Bureau Score	Bureau Score Bin (R)	Bureau Score WOE (R)
1	0	757	716 – 765	1.0914
2	1	NA	Missing	-0.6972
3	0	626	605 – 629	-0.9586
4	0	693	665 – 716	0.1776
5	0	706	665 – 716	0.1776
6	0	673	665 – 716	0.1776
7	0	730	716 – 765	1.0914

Diff obs so they re diff ppl with diff score but based on what you did it is the same bin so model treat them same. How strong is this bin at predicting

Odds ratio gone now cuz we have coeff of WOE now not coeff of bureau score (inrease by 1)

Different Inputs

- Instead of using the original variables for the model, scorecard models have the binned variables as their foundation.

needed this categorical rep to get WOE values

Observation	Target	Bureau Score	Bureau Score Bin (R)	Bureau Score WOE (R)
1	0	757	716 – 765	1.0914
2	1	NA	Missing	-0.6972
3	0	626	605 – 629	-0.9586
4	0	693	665 – 716	0.1776
5	0	706	665 – 716	0.1776
6	0	673	665 – 716	0.1776
7	0	730	716 – 765	1.0914

Different Inputs

This numerical variable that was once just tell me the value of your credit score is now a numerical representation of the strength of the bean,
and those bins were created to best predict the target.
So those numerical values are now directly tied to the target variable in a way that best predicts them.
So we no longer have the original credit score or bureau score values. Have instead best represetnation of score. Variable transformation.

- Instead of using the original variables for the model, scorecard models have the binned variables as their foundation.

cont to categorical to cont. We dont do this always cuz otherwise lose interpretabiliy.

- Inputs are still treated as continuous.
- All variables now on the same scale.
- Model **coefficients** are desired output for the scorecard.
- **Coefficients** now serve as measures of variable importance.

Bureau Score WOE (R)
1.0914
-0.6972
-0.9586
0.1776
0.1776
0.1776
1.0914

So now not only can I compare every continuous variable because every continuous variables on the same scale, every categorical variables on the same scale as my continuous variables.
And so all of those betas truly represent a notion of variable importance and variable strength.
I can literally compare any variable I'd like, any variable I'd like, and that's the benefit of this.
This is the underlying piece of putting everything into a point system, right
Because that's what a scorecard is. Everything's on a point system in a scorecard.

Different Inputs

give me result from smbinning funciton

this is a list that calling from. list has result of smbinning.

downside of smbinning is cant put all variables at a time. Have to put in 1 at a time. So looped and put result in a list

```
smbinning.gen(df = train, ivout = result_all_sig$bureau_score,
              chrname = "bureau_score_bin")
```

STeps taken: 1) Did sm binning on bureau score to figure values of categories, 2) smbinning.generate will create those categorical variables for u to put inside data.
 if you do not want to build a scorecard,but you want to be able to just put all categorical variables into your model, whatever your model is. if you want to just look at only categorical variables into a logistic regression model, completely fine (Fall 1 binned set of variables, labarr gave us binn and we inputted it)

Different Inputs

```
smbinning.gen(df = train, ivout = result_all_sig$bureau_score,
              chrname = “bureau_score_bin")
```

Observation	Target	Bureau Score	Bureau Score Bin (R)	Bureau Score WOE (R)
1	0	757	716 – 765	1.0914
2	1	NA	Missing	-0.6972
3	0	626	605 – 629	-0.9586
4	0	693	665 – 716	0.1776
5	0	706	665 – 716	0.1776
6	0	673	665 – 716	0.1776
7	0	730	716 – 765	1.0914

cross references which bin observation falls in and uses that value of WOE variable to model

Different Inputs

Every obs you are extracting out what bin you are in, once i know what bin i look up row in my dictorionary sort of and i look at value of weight of eveidence column.

sm binning labels missing category as 00 ie 'o'. c
Basically you have dictionary with values, you need to compare your categorical values to that dictionary and just replace with numbers of WOE

```
for (i in 1:nrow(train)) {
    bin_name <- "bureau_score_bin"
    bin <- substr(train[[bin_name]][i], 2, 2)
    woe_name <- "bureau_score_WOE"

    if(bin == 0) {
      bin <- dim(result_all_sig$bureau_score$ivtable)[1] - 1
      train[[woe_name]][i] <- result_all_sig$bureau_score$ivtable[bin, "WoE"]
    } else {
      train[[woe_name]][i] <- result_all_sig$bureau_score$ivtable[bin, "WoE"]
    }
}
```

Different Inputs

Observation	Target	Bureau Score	Bureau Score Bin (R)	Bureau Score WOE (R)
1	0	757	716 – 765	1.0914
2	1	NA	Missing	-0.6972
3	0	626	605 – 629	-0.9586
4	0	693	665 – 716	0.1776
5	0	706	665 – 716	0.1776
6	0	673	665 – 716	0.1776
7	0	730	716 – 765	1.0914

this is the goal to get it. Now just do logisitic regression Fall 1. Whether you had categorical or cont variables originally, all of them represnted by WOE column

Initial Scorecard

smbinning requires non defaulters as flagged as 1 (Good). But we are modelling Default in actually modelling.

8 variable WOE value sonly - 8 cuz got it from IV value. used IV for vairable selection.

```
initial_score <- glm(data = train, bad ~ tot_derog_WOE +
                                        tot_tr_WOE +
                                        age_oldest_tr_WOE +
                                        tot_rev_line_WOE +
                                        rev_util_WOE +
                                        bureau_score_WOE +
                                        down_pyt_WOE +
                                        ltv_WOE,
                     weights = train$weight, family = "binomial")
summary(initial_score)
```

INformamtion value tells you for all the levels of variable how does it predict Y vs WOE is level by level

cuz we have rare event problem. weights assigned during modelling

There are two ways of being able to undo the bias inside of the actual under sampling slash oversampling technique.
The first way is to be able to just adjust the intercept. The second ways to do weighted observations.
If you remember, adjusting the intercept only works if you're completely, 100% sure that those variables are right.
Okay. Weighted observation works better when you do not know if the variables you have are right.
And there was an observer, there was a number of observations limit in there as well.
We have over a thousand observations. We don't know if these variables are 100% right.
We're going to use weighted observations.

in world outside banking world, can remove variable that have high p value. IN banking world, use IV to variable select not p value.

Initial Scorecard

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6969  -0.7432  -0.4273  -0.1679   3.3704

Coefficients:
                  Estimate Std. Error z value             Pr(>|z|)
(Intercept)       -2.98190    0.04101 -72.706 < 0.0000000000000002 ***
tot_derog_WOE     -0.14478    0.08285  -1.747              0.08055 .
tot_tr_WOE        -0.04041    0.12726  -0.318              0.75084
age_oldest_tr_WOE -0.28207    0.09501  -2.969              0.00299 **
tot_rev_line_WOE  -0.38840    0.07963  -4.878           0.00000107 ***
bureau_score_WOE  -0.77495    0.05833 -13.286 < 0.0000000000000002 ***
rev_util_WOE      -0.23923    0.07643  -3.130              0.00175 **
down_pyt_WOE      -0.39379    0.14828  -2.656              0.00791 **
ltv_WOE           -0.86395    0.10116  -8.541 < 0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 6910.4  on 4376  degrees of freedom
Residual deviance: 6080.4  on 4368  degrees of freedom
AIC: 6185.1
```

How come variabel high p value but still got selected based on IV? So all by itself, that variable has predictive power on Y. But like every model, once we account for everything else in the model, that's what information each variable provides. So remember all of these p values are relative. what we refer to as type three tests - assuming every other variables in the model. What is the significance of this variable

So again, you can do variable selection like we talked about in the fall. Not interpretating coeff values here cuz its of WOE.

You can easily do that. There's no problem here. You can throw this into a lasso, you can do forward, you can do backward, you can do stepwise. But in banking you would not do this all. just output below and done.
In banking treat variables individually not holistically.

Model Evaluation

same as before

Now alll that is left is points column

- Variable significance – review using “standard” output of logistic regression, but don’t forget **business logic**.
- Overall performance of model – AUC (area under ROC curve, also called c) is the most popular criterion.
- This is only a **preliminary scorecard**.
- Final scorecard is created after reject inference is performed.

Model Evaluation

gives all metric results

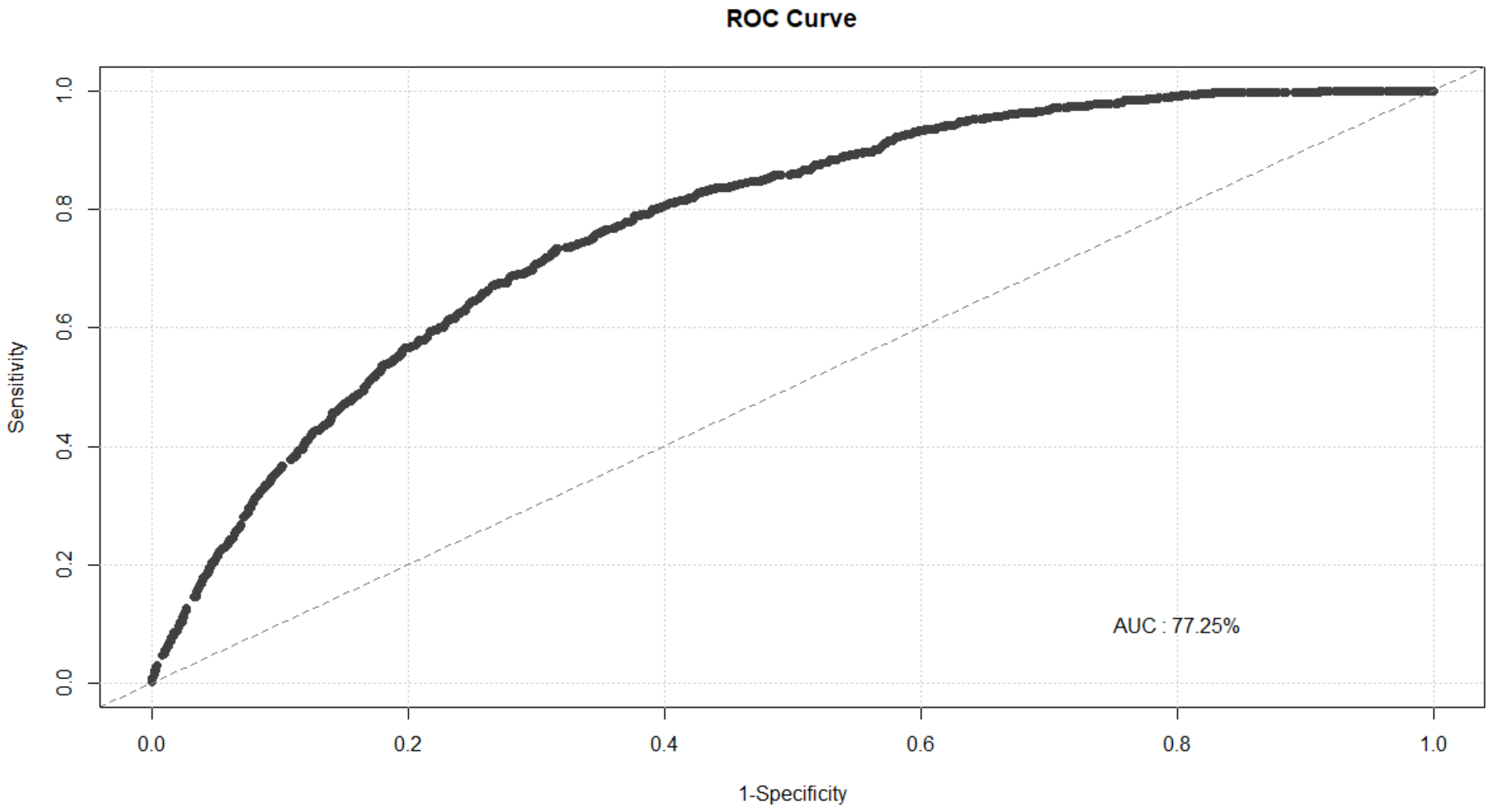
target in my model

get your predicted values prob

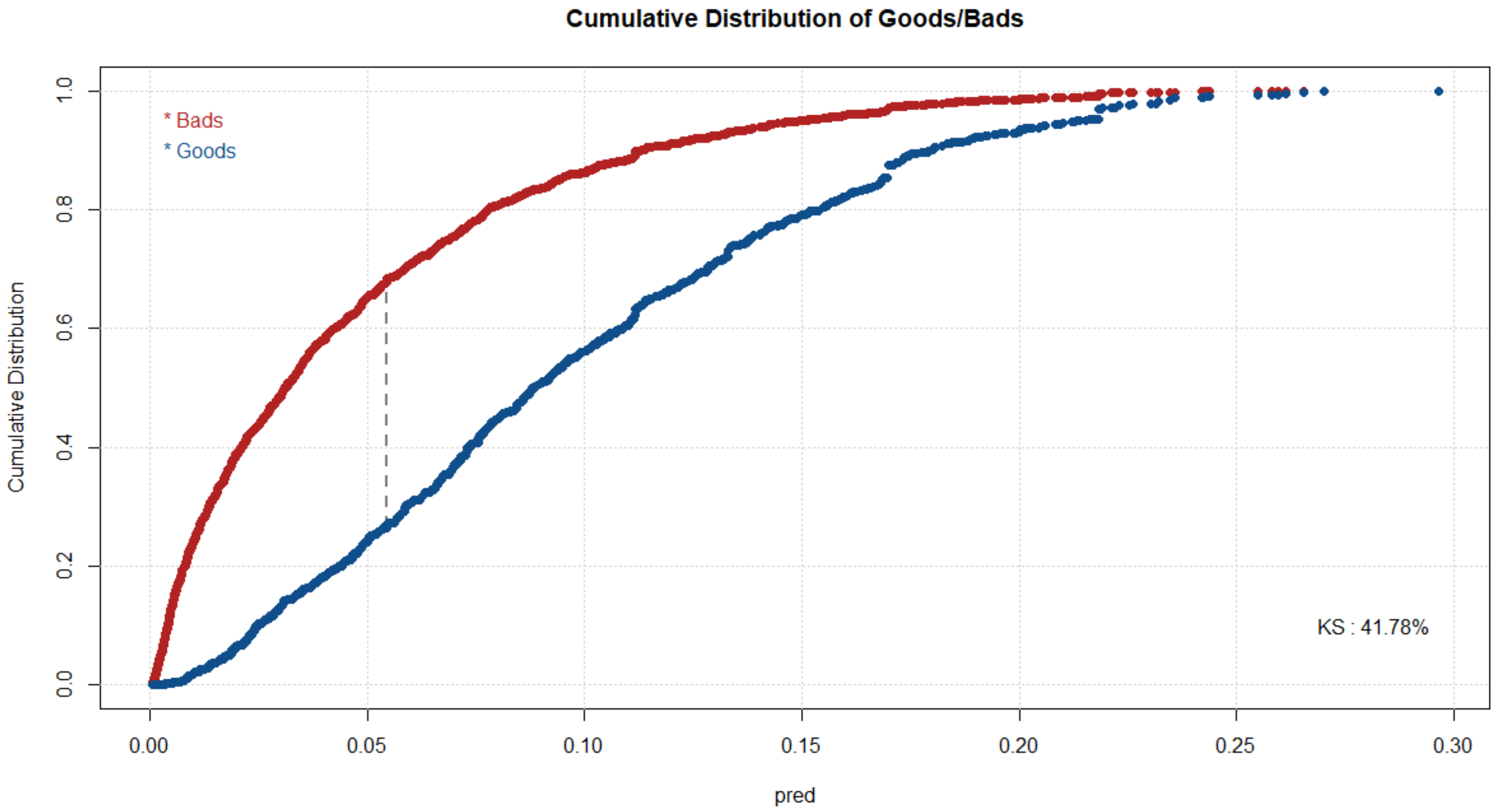
```
smbinning.metrics(dataset = train, prediction = "pred",
                  actualclass = "bad", report = 1)
smbinning.metrics(dataset = train, prediction = "pred",
                  actualclass = "bad", plot = "ks")
smbinning.metrics(dataset = train, prediction = "pred",
                  actualclass = "bad", plot = "auc")
```

1 means everything - AUC, KS, Youden Index, Optimal Cutoff

Model Evaluation



Model Evaluation



To summarize, to build model, i am not using original variables. I am using WOE of variables. If variables are cont, made them categorical and calculated WOE. If categorical i just calculated WOE with dictionary lookup. At end every variable represented by WOE correspondence. That goes into logistic reg, every variable is on same scale. Now all variables cont and cat are on same scale think standardization (where you std both categorical same scale as cont).

So now everything can be compared on a single metric.
That single metric are those beta coefficients from the logistic regression model.
And those beta coefficients are going to underlie the points that we're about to calculate. Thats why we did what we did.

Now, some of the other machine learning techniques we talked about, we have other things other than betas to summarize variable importance. eg Random Forest will give you that. But in logistic regression, we never had that. We never had one number that we could just draw upon to compare cuz things on diff scale.

SCALING THE SCORECARD

score card pt range all depends on you. Why they picked 350 to 850 is arbitraty.

You are gonna pick 2 points and entire scorecard built off of that. Algebra eqn. What are those two points? Essentially what we do is we call them factor and offset.
But that's really what you're looking at. You're looking at a slope and you're looking at an intercept for your point system.

Scaling the Scorecard

- The relationship between odds and scores is represented by a linear function:

$$Score = Offset + Factor \times \log(odds)$$

- If the scorecard is developed using “odds at a certain score” and “points to double the odds” (PDO), $Factor$ and $Offset$ can be calculated using the simultaneous equations:

$$\left.\begin{aligned} Score &= Offset + Factor \times \log(odds) \\ Score + PDO &= Offset + Factor \times \log(2 \times odds) \end{aligned}\right\}$$

2 eqn 2 unknowsn

So how do we do something like that instead of asking someone for an offset and the factor (2 points)? What we do is we ask them for two things. Give me the odds at a specific score. And then give me the points to double the odds (PDO)

Scaling the Scorecard

- Solving the equations for PDO, you get the following results:

$$PDO = Factor \times \log(2)$$

- Therefore,

$$Factor = \frac{PDO}{\log(2)}$$

$$Offset = Score - Factor \times \log(odds)$$

Scaling the Scorecard – Example

- If a scorecard were scaled where the developer wanted odds of 50:1 at 600 points and wanted the odds to double every 20 points (PDO = 20), *Factor* and *Offset* would be:

$$Factor = \frac{20}{\log(2)} = 28.8539$$

$$Offset = 600 - (28.8539 \times \log(50)) = 487.123$$

- Therefore, each score corresponding to each set of odds can be calculated as follows:

$$Score = 487.123 + 28.8539 \times \log(odds)$$

Scaling the Scorecard – Example

Score eqn corresponds to odds.
Get odds from model.
Scorecards can be easily built into any database cuz its just an eqn, once you have eqn you are done.

- If a scorecard were scaled where the developer wanted odds of 50:1 at 600 points and wanted the odds to double every 20 points (PDO = 20), *Factor* and *Offset* would be:

$$Factor = \frac{20}{\log(2)} = 28.8539$$

$$Offset = 600 - (28.8539 \times \log(50)) = 487.123$$

- Therefore, each score corresponding to each set of odds can be calculated as follows:

$$Score = 487.123 + 28.8539 \boxed{\times \log(odds)}$$

this is LHS of logistic reg eqn

Why people still love logistic regression!

Scaling the Scorecard – Example

Domain knowledge picked. ie if their odds were 50:1 , then their score was 600. If their odds were 20:1, their score is 620. Then all find is worst and best person obs, and that gives you your bounds on what you got thats how you ended up with 350 and 800. No One knew going in.

- If a scorecard were scaled where the developer wanted odds of 50:1 at 600 points and wanted the odds to double every 20 points (PDO = 20), *Factor* and *Offset* would be:

$$Factor = \frac{20}{\log(2)} = 28.8539$$

$$Offset = 600 - (28.8539 \times \log(50)) = 487.123$$

- Therefore, each score corresponding to each set of odds can be calculated as follows:

$$Score = 487.123 + 28.8539 \times \boxed{\log(odds)}$$

This is predicted value from logit function.

Scaling the Scorecard – Example

Score	Odds
600	50.0
601	51.8
604	57.4
.	
.	
.	
.	
620	100.0

probability of default. Each score has assocaited probability of default.

Points Allocation

each variable each category. How well that category does in separating 1s and 0s. Diff variables stronger than others in predicting Y. So incl Beta j. So those 2 things tell you your points for that category level in that variable. last term is scale things up to add up together.

- The points allocated to attribute *i* of characteristic *j* are computed as follows:

$$Points_{i,j} = -\left(WOE_{i,j} \times \hat{\beta}_j + \frac{\hat{\beta}_0}{L}\right) \times Factor + \frac{Offset}{L}$$

Once we have WOE multiplied by beta, the strenght of category multiplied by strenght of variable, then we need to scale it with times by Factor. Then we add in the OFfset, divide intercept by all points.

- $WOE_{i,j}$: Weight of evidence for attribute *i* of characteristic *j*
- $\hat{\beta}_j$: Regression coefficient for characteristic *j*
- $\hat{\beta}_0$: Intercept term from model
- L: Total number of characteristics
- Points typically rounded to nearest integer.

intercept over L. L is total variables in model. Every variable gets small piece of intercept. There are no intercepts in ML algos. All you need is sth that representeg is beta not, the strenght of variable - eg in RF get variable imp. Thats all u need instead of betas. So again, you can easily put this on top of a machine learning model. You just have to convert this equation to something more machine learning-esq. Instead of beta put variable imp, instead of intercept dont put anything.

Points Allocation

```
pdo <- 20
score <- 600
odds <- 50
fact <- pdo/log(2)
os <- score - fact*log(odds)
var_names <- names(initial_score$coefficients[-1])

for(i in var_names) {

    beta <- initial_score$coefficients[i]
    beta0 <- initial_score$coefficients["(Intercept)"]
    nvar <- length(var_names)
    WOE_var <- train[[i]]
    points_name <- paste(str_sub(i, end = -4), "points", sep = "")

    train[[points_name]] <- -(WOE_var*(beta) + (beta0/nvar))*fact + os/nvar
}
```

Points Allocation

This table just gives you observation and predicted score. But you want scorecard itself, what variable has what points ? Next slide takes everyone's score and breaks into separate pieces.

Observation	Target	Variables…	Observation Score
1	0	…	599
2	1	…	524
3	0	…	537
4	0	…	561
5	0	…	578
6	0	…	583
7	0	…	672

Points Allocation

You do this for every variable and you can hand over someone a score card. Pick a category for every variable and add the points together to get final.

WOE for Bureau Score					
Group	Values	Event Count	Non-event Count	WOE	**Scorecard Points**
1	< 603	111	112	-1.32	**50.4**
2	604 – 662	378	678	-0.74	**64.1**
3	663 – 699	185	754	0.08	**83.5**
4	700 – 717	74	440	0.46	**92.4**
5	718 – 765	75	824	1.07	**106.9**
6	> 765	15	498	2.18	**133.1**
7	MISSING	80	153	-0.68	**65.5**
Total		**918**	**3,459**		

get these points for every variable (broken out)

everyone below score of 603, get 50.3 points

Now, the latest research is to take SHAPLEY values and use those as scorecards.
So if you think about it, every individual has their own scorecard as compared to a scorecard for all.
Regulator issue cuz you bias ppl out potential.

3 techniques just tell us how to infer the target. Step 1build model, score oyur rejects, make sure everythig is balanced, step 4 is target

At this point you have a model, a scoreacrd on top of mode. Now you can analyze your current customers anyway you like. You have model you have given loans to ppl. What you should not do is apply model to ppl applying to get new loan. (model bias) but we have reject bias. Done here if banks just want to understand current customers.

REJECT INFERENCE

Process Flow

Data Collection
- Variable Selection
- Sample Size
- Sample / Performance Window

Data Cleaning
- Eliminate Duplicates
- Examine / Remove Outliers

Variable Grouping and Selection
- Weights of Evidence (WOE)
- Information Value (IV)
- Gini Criterion

Initial Scorecard Creation
- Logistic Regression
- Accuracy
- Threshold
- Assessment

Reject Inference
- Remove bias resulting from exclusion of rejects

Final Scorecard Creation
- Final Model Assessment

Reject Inference

- **Reject inference** is the process of inferring the status of the rejected applicants based on the accepted applicants model in an attempt to use their information to build a scorecard that is representative of the entire applicant population.
- Reject inference is about solving sample bias so that the development sample is similar to the population to which the scorecard will be applied.

problem is rejected dataset
doesnt have target

Rejected Inference

- Can we develop a scorecard without rejected applications? **YES!**
- Is it **legally permissible** to develop a scorecard without rejected applications? **YES!**
- If yes, then how **biased** would the scorecard model be? **DEPENDS!**

- *"My suggestion is to develop the scorecard using what data you have, but start saving rejected applications ASAP."*
  Raymond Anderson, Head of Scoring at Standard Bank Africa, South Africa

But I can't tell you. Of the people you didn't give loans to, did you make the right decision?

Why Reject Inference?

model is right but underlying data is missing reject inference.

- Initial scorecard used only **known** good and bad loans (accepted applicants only) – also called “behavioral scoring”
- Reduce bias in model and provide risk estimates for the “through-the-door” population – also called “application scoring”
- Comply with regulatory requirements (FDIC, Basel)
- Provide a scorecard that is able to generalize better to the entire credit application population.

bankers get together in Swiss. US pays attention but not follow

Reject Inference Techniques

- Three common techniques for reject inference:
  1. Hard Cutoff Augmentation
  2. Parceling Augmentation
  3. Fuzzy Augmentation (DEFAULT in SAS)

draw a line in sand, like youden index. Boave and below 1 and 0

Take your original model that you built on the accepts.
Yes, it's bias, but it's the only model we've got score the rejects with that model, basically create a target variable,
create a target variable for the rejects,
then combine the rejects and the accepts together into one huge data set and rebuild the entire modeling process.
Go all the way back to binning, rebuild your variables, rebuild your model, do all of that again,
and you will notice that the variables start changing y because now you have new information.

For your final model, cant have 50% rejects and 50% accepts. Have to reflect population 75-25.

Hard Cutoff Augmentation

1. Build a scorecard model using the known good/bad population (accepted applications)
2. Score the rejected applications with this model to obtain each rejected applicant's probability of default and their score on the scorecard model.
3. Create weighted cases for the rejected applicants – weight applied is the "rejection rate" which adjusts the number of sampled rejects to accurately reflect the number of rejects from population.

Two undersampling problems (see OneNote screenshot): 1) Initial model, you had 3% default rare eevent that you solved using weights=weight in glm (Rare event). 2) But now after combing reject inference you have 50% reject and 50% accept rows. BUT real population is 75% ppl accepted for loan, 25% reject. So our combined data set should look like real population so you just undersample the reject inference to match 75-25 in population.

You have luxury to just undersample to make it go from 50-50 to 75-25 but initially when you modelled default it waas 97-3, so didnt have luxury to make model data be 97-3. Had to oversample

Hard Cutoff Augmentation

4. Set a cut-off score level above which applicant is deemed good and below applicants deemed bad.
5. Add inferred goods and bads with known goods and bads and rebuild scorecard.

Take your rejects, score them based on your model (score means create target variable), Inffer whetehr they default or not. Guess target for rejects, then combine itall into 1 data set. Then go back to modelling process - binning re bin, rebuild model, you will notice variable cahnge due to o new info.

Hard Cutoff Augmentation

pred probability or score

create new default.bad=1.

```
rejects_scored$pred <- predict(initial_score, newdata = rejects_scored,
                                                type = 'response')
rejects$bad <- as.numeric(rejects_scored$pred > 0.0545)
rejects$weight <- ifelse(rejects$bad == 1, 2.80, 0.59)
rejects$good <- abs(rejects$bad - 1)

comb_hard <- rbind(accepts, rejects)
```

first model Youden Index (could be F1 too)

add good col, cuz your accepts data set has it.

harder way to do it, just do undersample way. instead of 2 weghts.

NOW repeat all sets form before

Hard Cutoff Augmentation

```
rejects_scored$pred <- predict(initial_score, newdata = rejects_scored,
                               type = 'response')

rejects$bad <- as.numeric(rejects_scored$pred > 0.0545)
rejects$weight <- ifelse(rejects$bad == 1, 2.80, 0.59)
rejects$good <- abs(rejects$bad - 1)

comb_hard <- rbind(accepts, rejects)
```

how you got thi ? Used youden index but can use anything. Can use KS stat or F1 score or anything.

Hard Cutoff Augmentation

```
rejects_scored$pred <- predict(initial_score, newdata = rejects_scored,
                               type = 'response')
rejects$bad <- as.numeric(rejects_scored$pred > 0.0545)
rejects$weight <- ifelse(rejects$bad == 1, 2.80, 0.59)
rejects$good <- abs(rejects$bad - 1)

comb_hard <- rbind(accepts, rejects)
```

watch the video on how to do weights twice. But if you got enough data, then instead just sample down rejects to reflect population.

after rbind 2 sets, re do everything. We are extrapolating accepts model to reject cuz tahts all you got. Now after combining does model change its opinion

smbinning needed 1s to be good and 0s to be bad

Parceling Augmentation

nust change how you get 0s or 1s for target
1-3 steps al lsame.

This is just diff way of doing cutoff. Before no notion of grey area.

1. Build a scorecard model using the known good/bad population (accepted applications)
2. Score the rejected applications with this model to obtain each rejected applicant's probability of default and their score on the scorecard model.
3. Create weighted cases for the rejected applicants – weight applied is the "rejection rate" which adjusts the number of sampled rejects to accurately reflect the number of rejects from population.

parcelling means break into smaller groups

could be pred prob. high score is low prob. work oppposte.

Parceling Augmentation

4. Define score ranges manually or automatically with simple bucketing.
5. The inferred good/bad status of the rejected applicants will be assigned **randomly** and proportional to the number of goods and bads in the accepted population within each score range.
6. If desired, apply the event rate increase factor to P(bad) to increase the proportion of bads among the rejects (oversampling with the rejects)
7. Add the inferred goods and bads back in with the known goods and bads and rebuild the scorecard.

Parceling Augmentation – Example

	Accepted Applicants				Rejected Applicants		
Score Range	# Bad	# Good	% Bad	%Good	Rejects	# Inferred Bad	# Inferred Good
< 655	0	0	0%	0%	5	?	?

This range is 'never seen before range'. Is there anyone in our applicants that we accepted?
So the stuff the model was built off of that we've ever seen get a score that low?
Nope. Never seen someone with a score that low.
Welcome to the world of extrapolation. We are literally about to score people who we've never seen before in terms of something that low.
But there are on the rejected applicant side, five individuals, five individuals who score that low, even though we've never seen people score that low for the accepted. Okay, so what do we do?

Parceling Augmentation – Example

	Accepted Applicants				Rejected Applicants		
Score Range	# Bad	# Good	% Bad	%Good	Rejects	# Inferred Bad	# Inferred Good
< 655	0	0	0%	0%	5	5	0

Assume bad if no information to prove otherwise

in this range in our model our evidence is 0%, but assume bad cuz no info otherwise to disprove it.

Parceling Augmentation – Example

	Accepted Applicants				Rejected Applicants		
Score Range	# Bad	# Good	% Bad	%Good	Rejects	# Inferred Bad	# Inferred Good
< 655	0	0	0%	0%	5	5	0
655 – 665	300	360	45.5%	54.5%	190	?	?

Parceling Augmentation – Example

	Accepted Applicants				Rejected Applicants		
Score Range	# Bad	# Good	% Bad	%Good	Rejects	# Inferred Bad	# Inferred Good
< 655	0	0	0%	0%	5	5	0
655 – 665	300	360	45.5%	54.5%	190	86	?

somebanks may do reject bump because say if you give rejected applicants loans, then for same score range they would still have defaulted at higher rate. Thats why you artifically inflate bad%

you are applying this same % to rejects based on original data

$$0.455 \times 190 \approx 86$$

Randomly assign!

190 is total in reject set for that range.
45.5% of that is inferred bad defaulters

Parceling Augmentation – Example

	Accepted Applicants				Rejected Applicants		
Score Range	# Bad	# Good	% Bad	%Good	Rejects	# Inferred Bad	# Inferred Good
< 655	0	0	0%	0%	5	5	0
655 – 665	300	360	45.5%	54.5%	190	86	114

$$190 - 86 = 114$$

Parceling Augmentation – Example

Banks first use external info to make decisions until they get enough info within bank.

	Accepted Applicants				Rejected Applicants		
Score Range	# Bad	# Good	% Bad	%Good	Rejects	# Inferred Bad	# Inferred Good
< 655	0	0	0%	0%	5	5	0
655 – 665	300	360	45.5%	54.5%	190	86	114
665 – 675	450	700	39.1%	60.9%	250	98	152
…	…	…	…	…	…	…	…

```
parc <- seq(500, 725, 25)

accepts_scored$Score_parc <- cut(accepts_scored$Score, breaks = parc)
rejects_scored$Score_parc <- cut(rejects_scored$Score, breaks = parc)

table(accepts_scored$Score_parc, accepts_scored$bad)

parc_perc <- table(accepts_scored$Score_parc, accepts_scored$bad)[,2] /
                   rowSums(table(accepts_scored$Score_parc, accepts_scored$bad))

rejects$bad <- 0

rej_bump <- 1.25

for(i in 1:(length(parc) - 1)) {
    for(j in 1:length(rejects_scored$Score)) {
        if((rejects_scored$Score[j] > parc[i]) &
           (rejects_scored$Score[j] <= parc[i+1]) &
           (runif(n = 1, min = 0, max = 1) < (rej_bump*parc_perc[i]))) {
            rejects$bad[j] <- 1
        }
    }
}

table(rejects_scored$Score_parc, rejects$bad)
rejects$weight <- ifelse(rejects$bad == 1, 2.80, 0.59)
rejects$good <- abs(rejects$bad - 1)

comb_parc <- rbind(accepts, rejects)
```

```
parc <- seq(500, 725, 25)

accepts_scored$Score_parc <- cut(accepts_scored$Score, breaks = parc)
rejects_scored$Score_parc <- cut(rejects_scored$Score, breaks = parc)

table(accepts_scored$Score_parc, accepts_scored$bad)

parc_perc <- table(accepts_scored$Score_parc, accepts_scored$bad)[,2] /
                   rowSums(table(accepts_scored$Score_parc, accepts_scored$bad))

rejects$bad <- 0

rej_bump <- 1.25

for(i in 1:(length(parc) - 1)) {
    for(j in 1:length(rejects_scored$Score)) {
        if((rejects_scored$Score[j] > parc[i]) &
           (rejects_scored$Score[j] <= parc[i+1]) &
           (runif(n = 1, min = 0, max = 1) < (rej_bump*parc_perc[i]))) {
            rejects$bad[j] <- 1
        }
    }
}

table(rejects_scored$Score_parc, rejects$bad)
rejects$weight <- ifelse(rejects$bad == 1, 2.80, 0.59)
rejects$good <- abs(rejects$bad - 1)

comb_parc <- rbind(accepts, rejects)
```

Fuzzy Augmentation

1. Build a scorecard model using the known good/bad population (accepted applications)
2. Score the rejected applications with this model to obtain each rejected applicant's probability of being good, P(Good), and probability of being bad, P(Bad).
3. **Do not assign a reject to a good/bad class** – create **two weighted cases** for each rejected applicant using P(Good) and P(Bad).

Represent good and bad side of Ann Marie. Everyone angel devil same have both in model. but good evan willl have a higher weight. Every obs have good and bad just have diff weights.

Rememer loony tune cartoon story (a devil and angel version of character)

If predicted prob is 0.5, the 1s and 0s for that observation get 50% weights. Weights is what gets adjusted. If pred prob is 0.8, then you get4 times the weight.

Fuzzy Augmentation

if everyone get os and 1s,
then replicating data so it will
change your popilation
sames 75-25

4. Multiply P(Good) and P(Bad) by the user-specific rejection rate to form frequency variables.
5. For each rejected applicant, create **two observations** – one observation has a frequency variable (rejection weight × P(Good)) and a target variable of 0; other observation has a frequency variable (rejection weight × P(Bad)) and a target variable of 1.
6. Add inferred goods and bads back in with the known goods and bads and rebuild the scorecard.

Fuzzy Augmentation

So no longer will your models sit there and be like, I've never seen anybody that look like that, so I'm just going to have to assume they're are bad.
Like, Nope, you've seen everybody look like that, both good and bad.

```
rejects_scored$pred <- predict(initial_score, newdata = rejects_scored,
                               type = 'response')

rejects_g <- rejects
rejects_b <- rejects

rejects_g$bad <- 0
rejects_g$weight <- (1 - rejects_scored$pred)*2.80
rejects_g$good <- 1

rejects_b$bad <- 1
rejects_b$weight <- (rejects_scored$pred)*0.59
rejects_b$good <- 0

comb_fuzz <- rbind(accepts, rejects_g, rejects_b)
```

So for this because the way the data set was built, I didn't under sample. i Did double weights instead

Reject Inference Techniques

- Three common techniques for reject inference:
  1. Hard Cutoff Augmentation
  2. Parceling Augmentation
  3. Fuzzy Augmentation (DEFAULT in SAS EM)

- There are other techniques as well, but are not as highly recommended.

Other Reject Inference Techniques

1. Assign all rejects to bads.
2. Assign rejects in the same proportion of goods to bads as reflected in the accepted data set.
3. Similar in-house model on different data.
4. Approve all applicants for certain period of time.
5. Clustering
6. Memory based reasoning

Other Reject Inference Techniques

1. Assign all rejects to bads
   - Appropriate only if approval rate is very high (ex. 97%) and there is a high degree of confidence in adjudication process.
2. Assign rejects in the same proportion of goods to bads as reflected in the accepted data set.
3. Similar in-house model on different data.
4. Approve all applicants for certain period of time.
5. Clustering
6. Memory based reasoning

Other Reject Inference Techniques

1. Assign all rejects to bads.
2. Assign rejects in the same proportion of goods to bads as reflected in the accepted data set.
   - Assignment done completely at random!
   - Valid only if current system has no consistency.
3. Similar in-house model on different data.
4. Approve all applicants for certain period of time.
5. Clustering
6. Memory based reasoning

Other Reject Inference Techniques

1. Assign all rejects to bads.
2. Assign rejects in the same proportion of goods to bads as reflected in the accepted data set.
3. Similar in-house model on different data.
   - Performance on similar products used as proxy.
   - Hard to pass by regulators.
4. Approve all applicants for certain period of time.
5. Clustering
6. Memory based reasoning

Other Reject Inference Techniques

1. Assign all rejects to bads.
2. Assign rejects in the same proportion of goods to bads as reflected in the accepted data set.
3. Similar in-house model on different data.
4. Approve all applicants for certain period of time.
   - Provides actual performance of rejects instead of inferred.
   - Might be “legal” problems…
5. Clustering
6. Memory based reasoning

Other Reject Inference Techniques

If bank is discriminating by rejecting for loan, then including reject inference to help fight those things. There to help with them. sometimes they may have 2 models - one they show regualtor, one they actual use.

1. Assign all rejects to bads.
2. Assign rejects in the same proportion of goods to bads as reflected in the accepted data set.
3. Similar in-house model on different data.
4. Approve all applicants for certain period of time.
5. Clustering
6. Memory based reasoning

need to have some accepts and bad in every cluster to be able to model

FINAL SCORECARD CREATION

Process Flow

Data Collection
- Variable Selection
- Sample Size
- Sample / Performance Window

Data Cleaning
- Eliminate Duplicates
- Examine / Remove Outliers

Variable Grouping and Selection
- Weights of Evidence (WOE)
- Information Value (IV)
- Gini Criterion

Initial Scorecard Creation
- Logistic Regression
- Accuracy
- Threshold
- Assessment

Reject Inference
- Remove bias resulting from exclusion of rejects

Final Scorecard Creation
- Final Model Assessment

Final Scorecard Creation

- The mechanics of building the final scorecard model are identical with the initial scorecard creation except that analysis is performed **after reject inference**.
- Accuracy Measurements:
  - Repeat review of the logistic model estimated parameters, life, KS, ROC, etc.

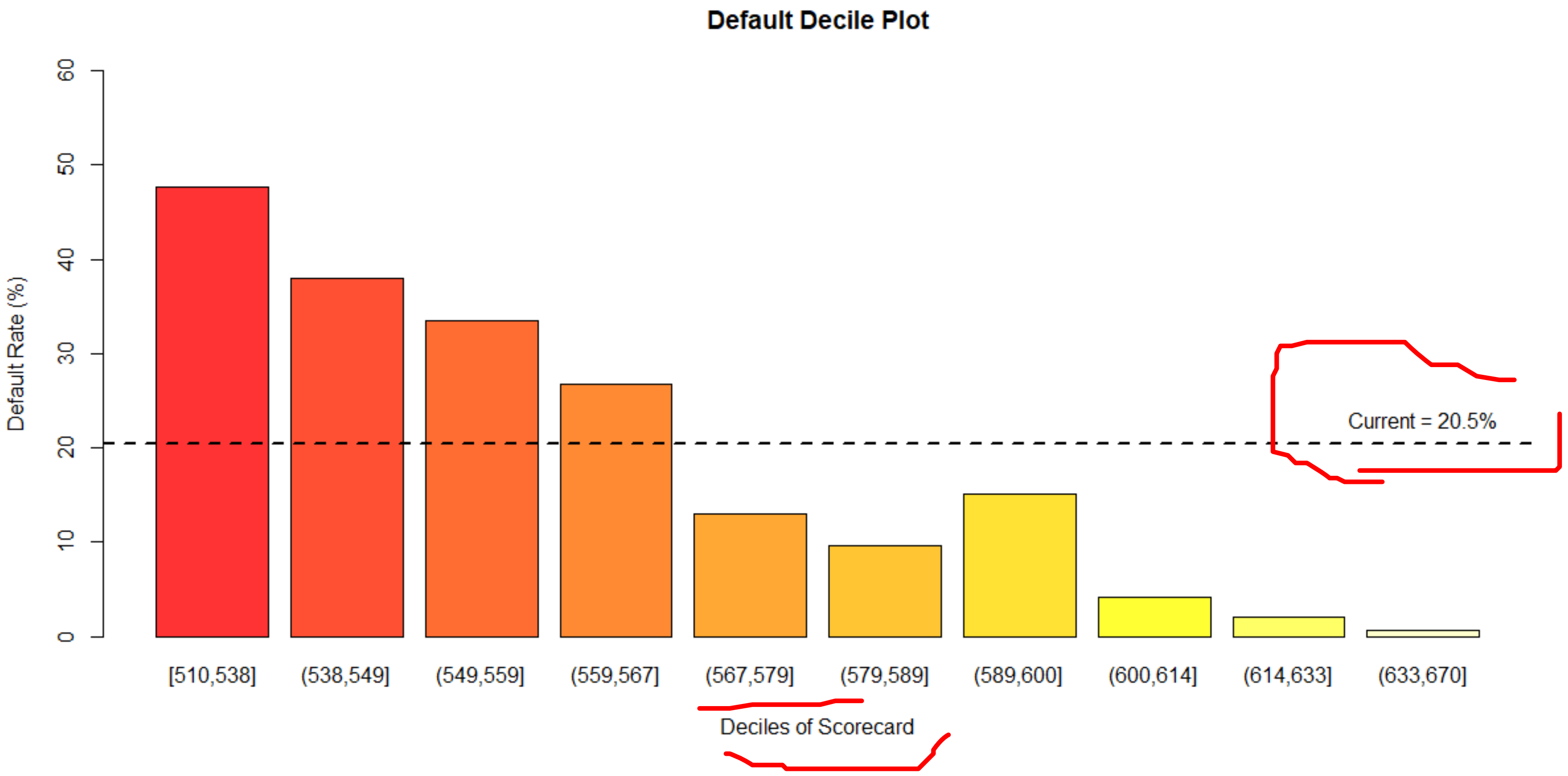
Defining Cut-off

dont just use any cutoff, evaluate the cost

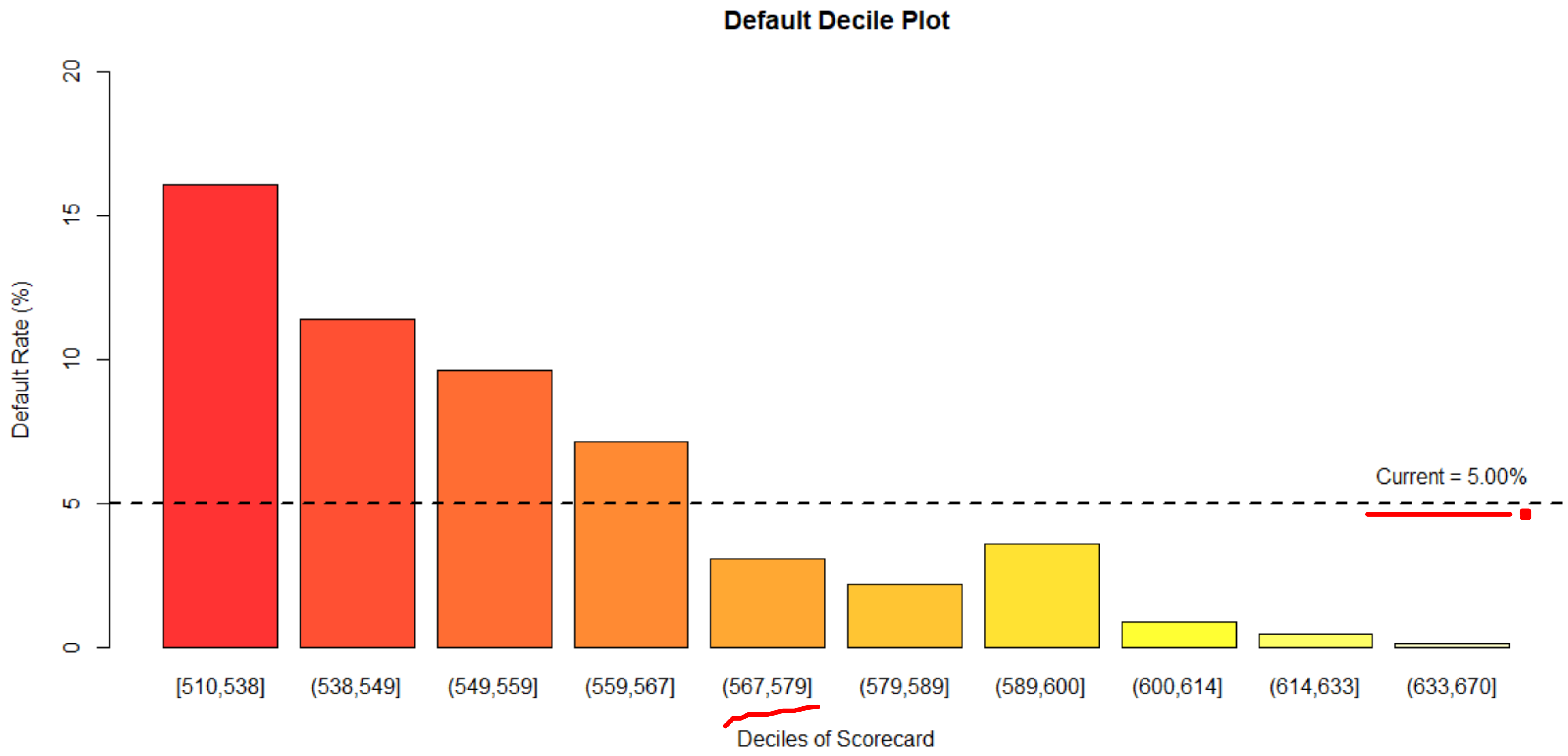
- A new scored should be better than the last in terms of one of the following:
  - Lower bad rate for the same approval rate.
  - Higher approval rate while holding the bad rate constant.

Default Decile Plot

this is how they do cutoff
based on cost. break down
in to 10 equal decile groups.



Default Decile Plot



Defining Cut-off

- Trade-off Plots:
  - The reference lines of approval rate and event (bad) rate are predefined by analyst.
  - How much risk are you willing to take on?

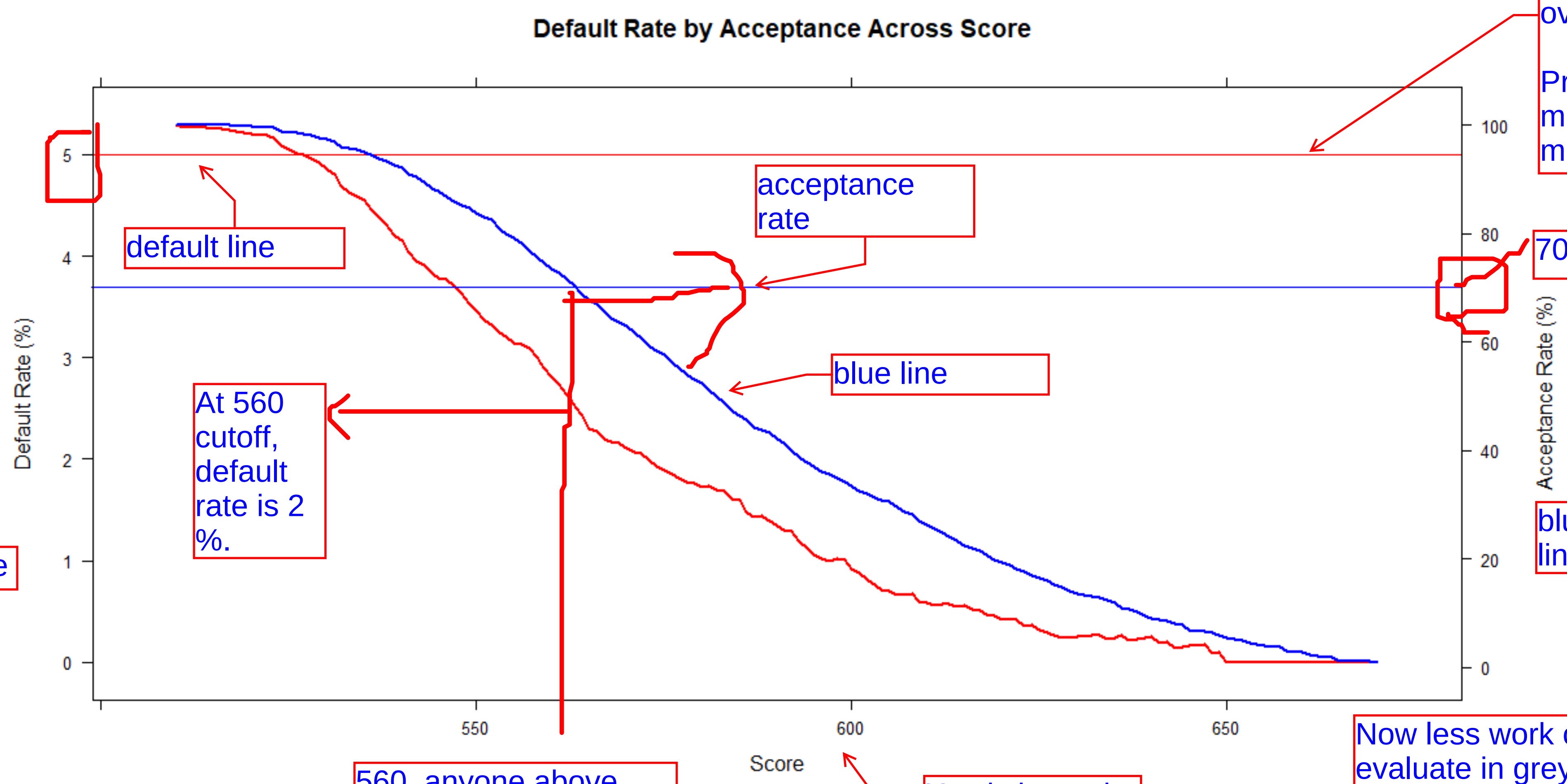
What does the bank want to do? Does the bank want to lower its default? maximize profit. If you want to lower your default, I can keep your acceptance rate the same.Does the bank want to increase its customer base and they're okay with their default rate?

Defining Cut-off

Dual axis plot

current default rate at bank is 5%. If want to keep this same at bank, then anyone over 500 accept them

Profit is i know how much each loan makes me $



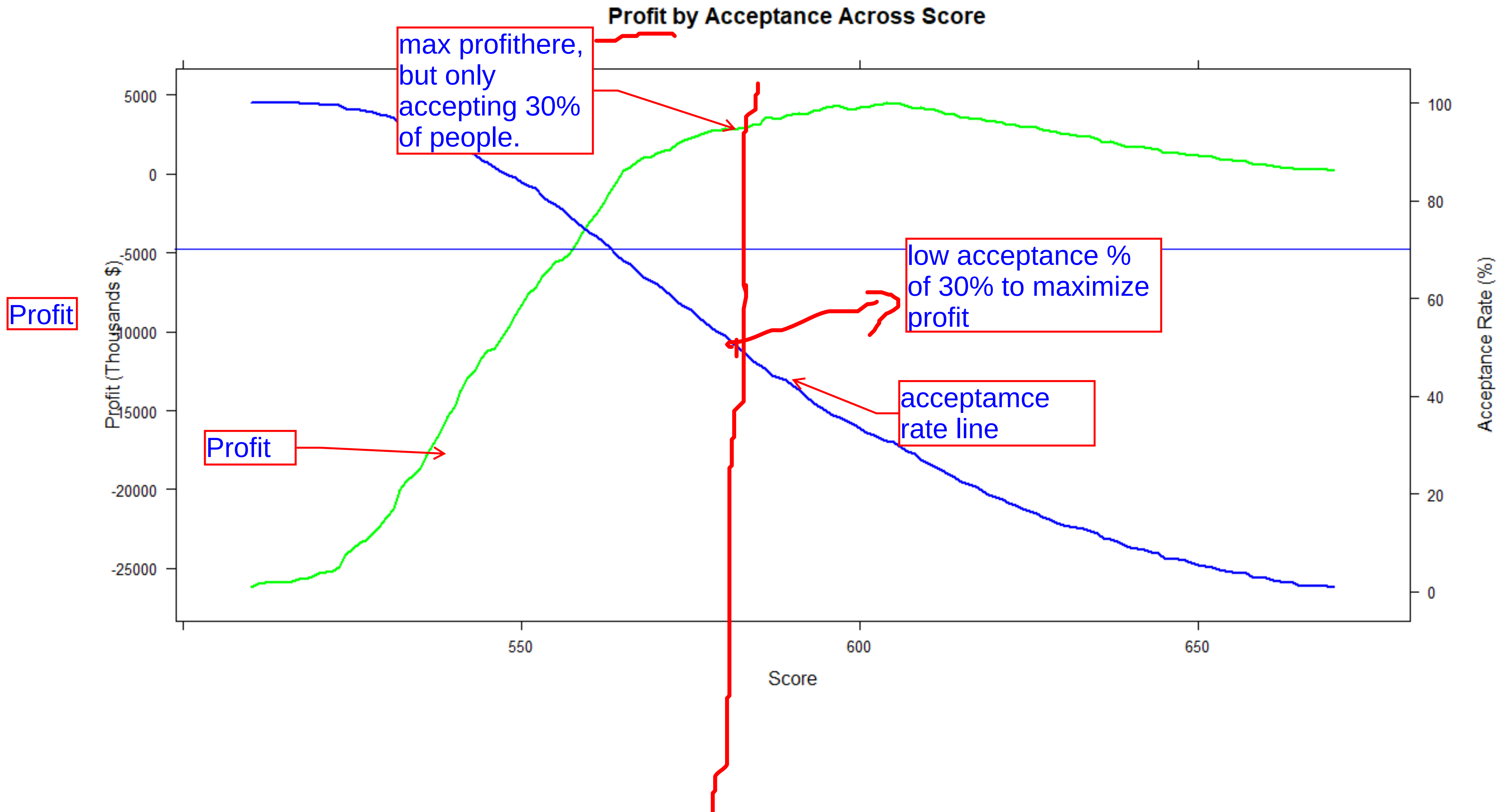
560, anyone above 560 you accept to keep same acceptance %

X axis is pred score from model (target)

Now less work cuz only have to evaluate in grey area ppl in middle (given default rate and acceptance rate of bank). So bankers only eval ppl in middle.

Defining Cut-off

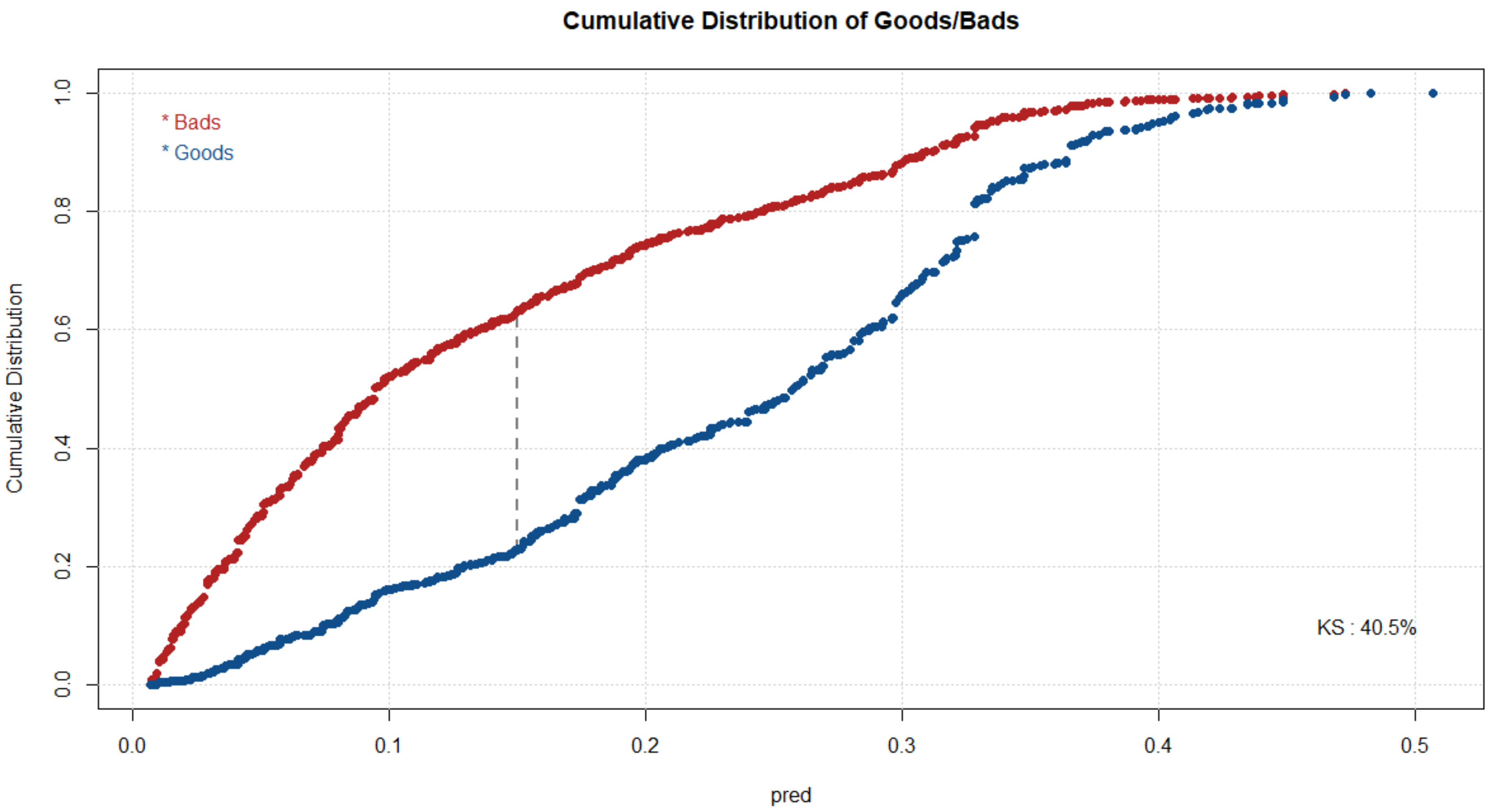
I know how much each loan makes me $, how much i lose when someone defaults.



Defining Cut-off

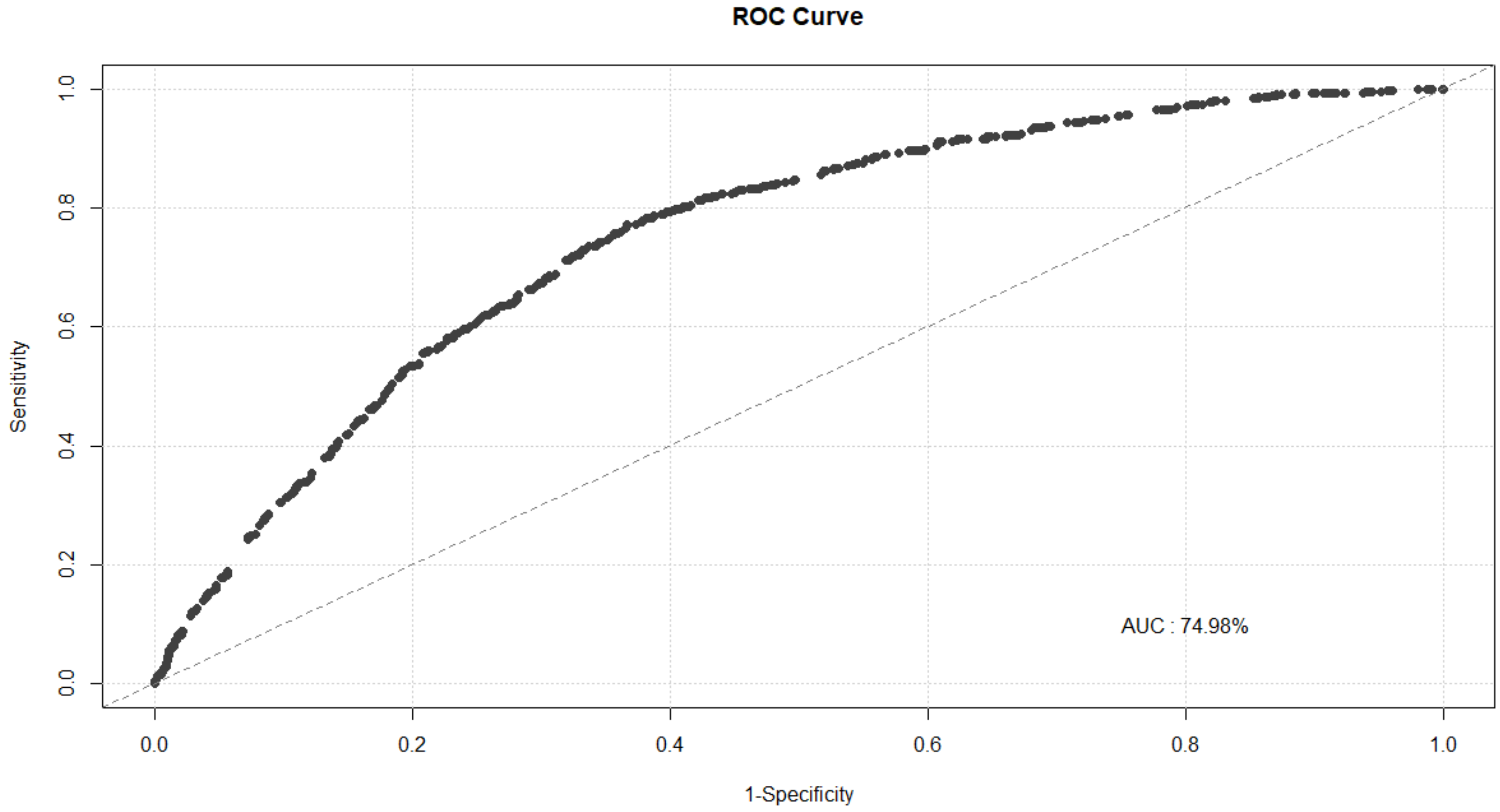
- Setting Multiple Cut-offs Example:
  - Anyone who scores above 210 points is accepted automatically.
  - Anyone who scores below 190 is declined.
  - Any scores in between 190 and 210 are referred to manual adjudication.

Final Scorecard – Example



Final Scorecard – Example

ROC curve good or bad? No idea, it depends. Need sth to compare with. Exisitng



CREDIT SCORING MODEL EXTENSIONS

Lack of Interactions

- Benefits of tree based algorithms are inherent interactions of every split of the tree.
  - Also a detriment to interpretation.

Multi-stage Model

- Benefits of tree based algorithms are inherent interactions of every split of the tree.
  - Also a detriment to interpretation.

- Multi-stage model:
  1. Decision Tree to initially get a couple of layers of splits.
  2. Build logistic regression based scorecard in each of the splits.
  3. Interpretation is now **within** a split (sub-group) of the data.

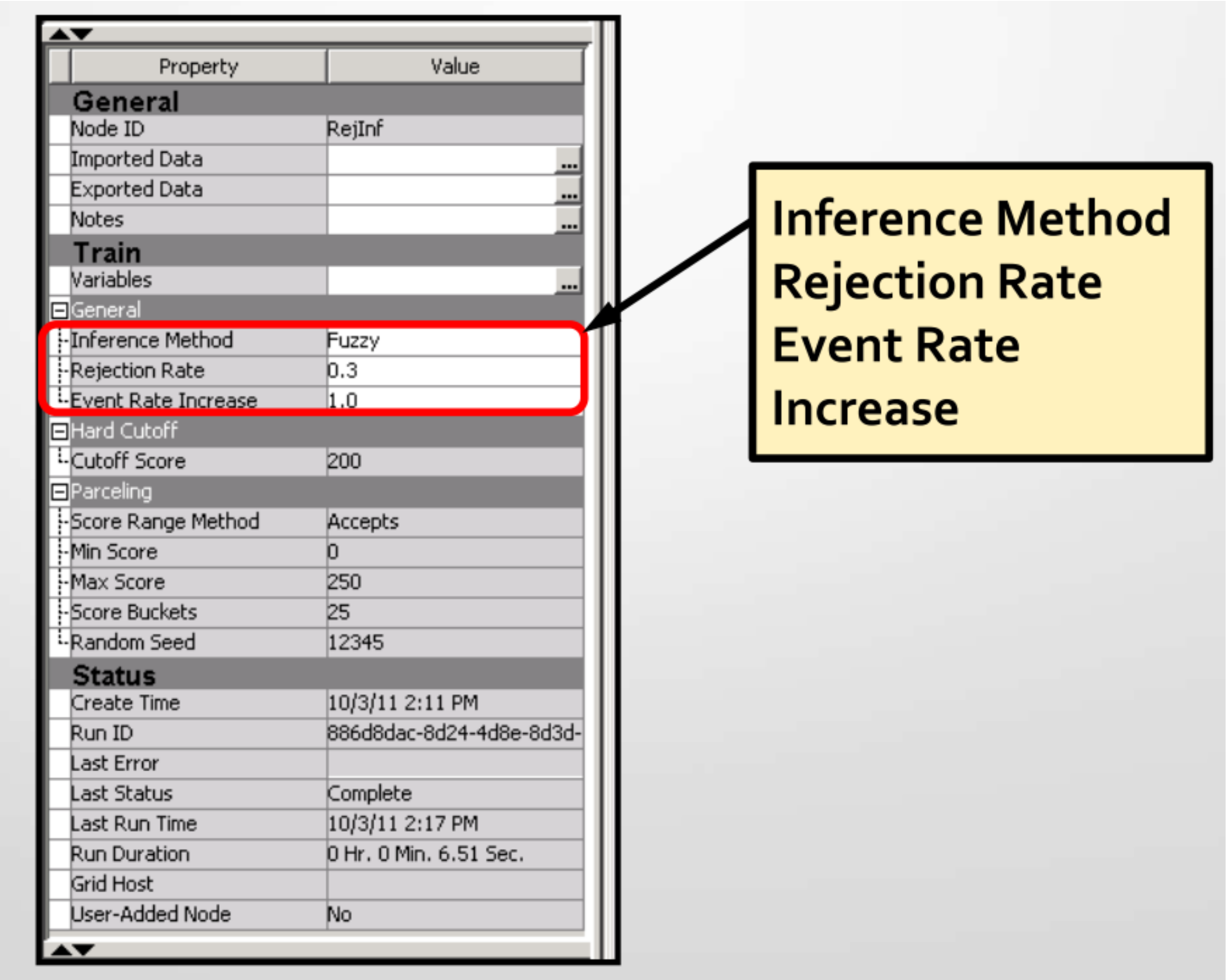
Machine Learning

- Model interpretation is **KEY** in the world of credit scoring.
- Scorecard layer may help drive interpretation of machine learning algorithms, but regulators are still hesitant.

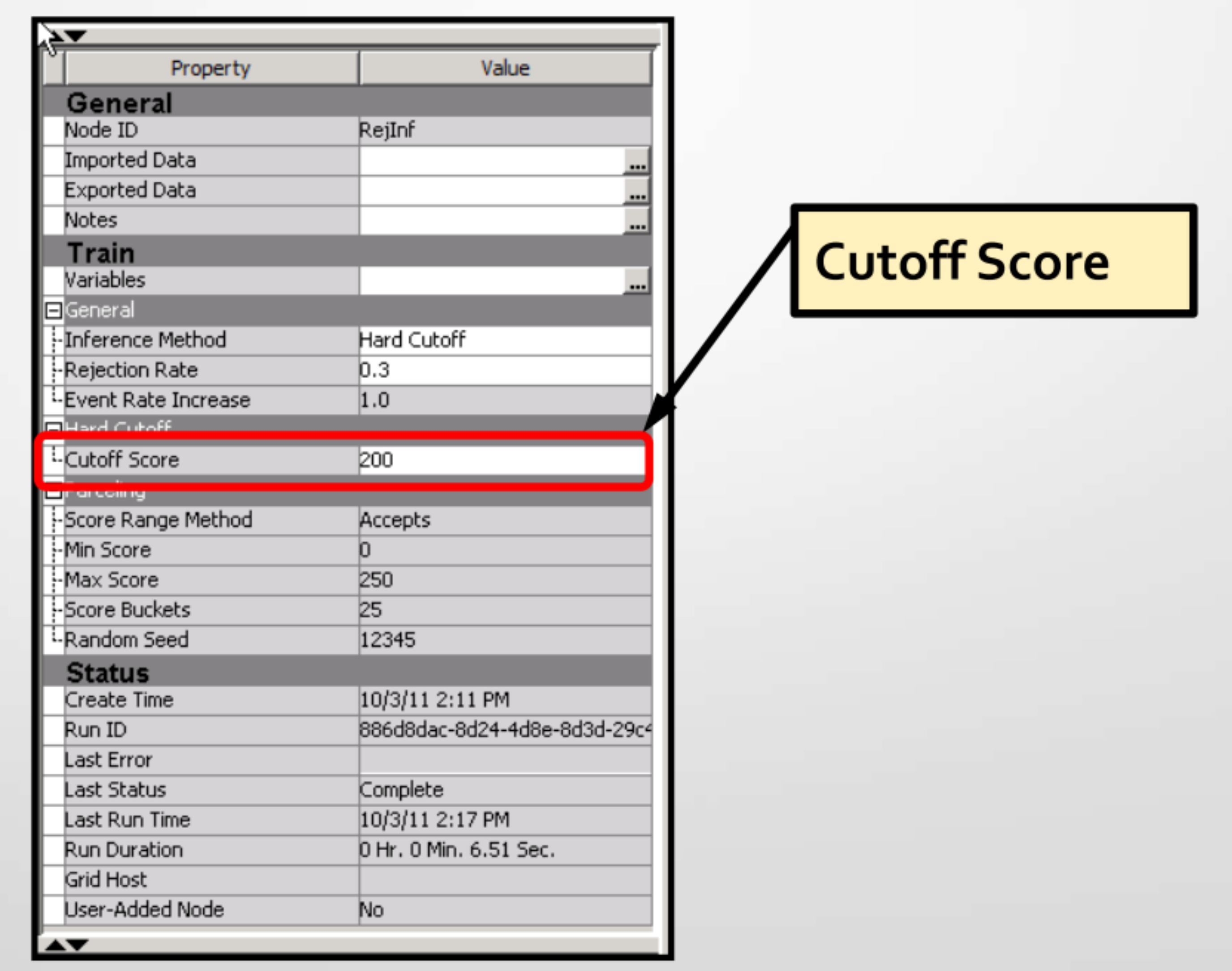
- Great for internal comparison and variable selection.
  - Build a neural network, tree based algorithm, etc. to see if model is statistically different than logistic regression scorecard.
  - Empirical examples have shown WOE based logistic regressions perform very well in comparison to more complicated approaches.

REJECT INFERENCE NODE IN SAS EM

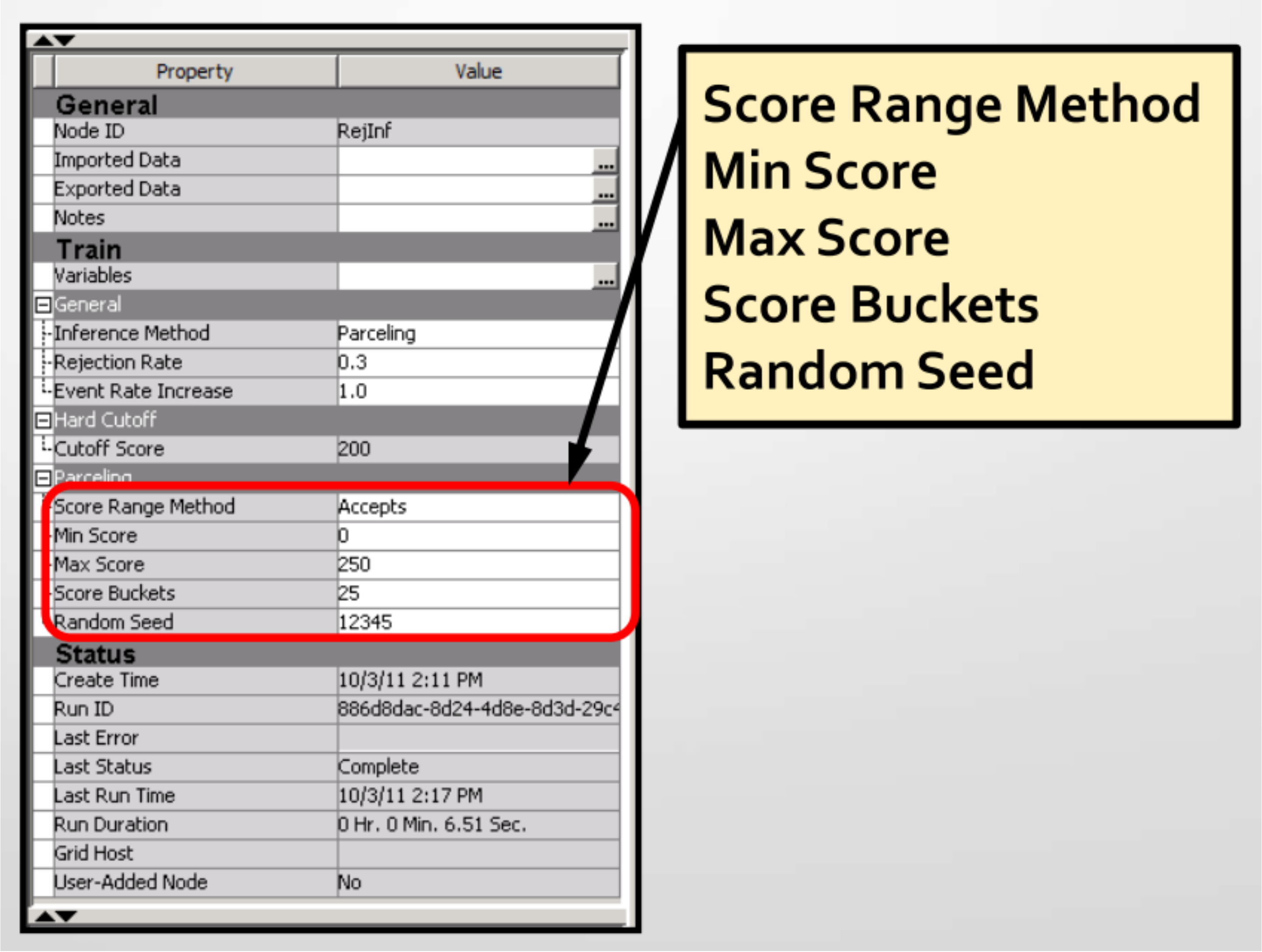
General Options



Hard Cut-off Options



Parceling Options



Property	Value
General	
Node ID	RejInf
Imported Data	...
Exported Data	...
Notes	...
Train	
Variables	...
General	
Inference Method	Parceling
Rejection Rate	0.3
Event Rate Increase	1.0
Hard Cutoff	
Cutoff Score	200
Parceling	
Score Range Method	Accepts
Min Score	0
Max Score	250
Score Buckets	25
Random Seed	12345
Status	
Create Time	10/3/11 2:11 PM
Run ID	886d8dac-8d24-4d8e-8d3d-29c
Last Error	
Last Status	Complete
Last Run Time	10/3/11 2:17 PM
Run Duration	0 Hr. 0 Min. 6.51 Sec.
Grid Host	
User-Added Node	No