Chapter 2

Example 2.3.1

# Code to use for binning (Section 2. Univariate analysis)

# 2.1 woe based on binning analysis

#Bureau score:

train$woe\_bureau\_score<- rep(NA, length(train$bureau\_score))

train$woe\_bureau\_score[which(is.na(train$bureau\_score))] <- -0.0910

train$woe\_bureau\_score[which(train$bureau\_score <= 308)] <- -0.7994

train$woe\_bureau\_score[which(train$bureau\_score > 308 & train$bureau\_score <= 404)] <- -0.0545

train$woe\_bureau\_score[which(train$bureau\_score > 404 & train$bureau\_score <= 483)] <- 0.7722

train$woe\_bureau\_score[which(train$bureau\_score > 483)] <- 1.0375

test$woe\_bureau\_score<- rep(NA, length(test$bureau\_score))

test$woe\_bureau\_score[which(is.na(test$bureau\_score))] <- -0.0910

test$woe\_bureau\_score[which(test$bureau\_score <= 308)] <- -0.7994

test$woe\_bureau\_score[which(test$bureau\_score > 308 & test$bureau\_score <= 404)] <- -0.0545

test$woe\_bureau\_score[which(test$bureau\_score > 404 & test$bureau\_score <= 483)] <- 0.7722

test$woe\_bureau\_score[which(test$bureau\_score > 483)] <- 1.0375

#CC utilization:

train$woe\_cc\_util<- rep(NA, length(train$cc\_util))

train$woe\_cc\_util[which(is.na(train$cc\_util))] <- 0

train$woe\_cc\_util[which(train$cc\_util <= 0.55)] <- 1.8323

train$woe\_cc\_util[which(train$cc\_util > 0.55 & train$cc\_util <= 0.70)] <- -0.4867

train$woe\_cc\_util[which(train$cc\_util > 0.70 & train$cc\_util <= 0.85)] <- -1.1623

train$woe\_cc\_util[which(train$cc\_util > 0.85)] <- -2.3562

test$woe\_cc\_util<- rep(NA, length(test$cc\_util))

test$woe\_cc\_util[which(is.na(test$cc\_util))] <- 0

test$woe\_cc\_util[which(test$cc\_util <= 0.55)] <- 1.8323

test$woe\_cc\_util[which(test$cc\_util > 0.55 & test$cc\_util <= 0.70)] <- -0.4867

test$woe\_cc\_util[which(test$cc\_util > 0.70 & test$cc\_util <= 0.85)] <- -1.1623

test$woe\_cc\_util[which(test$cc\_util > 0.85)] <- -2.3562

#Number of CCJ events:

train$woe\_num\_ccj<- rep(NA, length(train$num\_ccj))

train$woe\_num\_ccj[which(is.na(train$num\_ccj))] <- -0.0910

train$woe\_num\_ccj[which(train$num\_ccj <= 0)] <- 0.1877

train$woe\_num\_ccj[which(train$num\_ccj > 0 & train$num\_ccj <= 1)] <- -0.9166

train$woe\_num\_ccj[which(train$num\_ccj > 1)] <- -1.1322

test$woe\_num\_ccj<- rep(NA, length(test$num\_ccj))

test$woe\_num\_ccj[which(is.na(test$num\_ccj))] <- -0.0910

test$woe\_num\_ccj[which(test$num\_ccj <= 0)] <- 0.1877

test$woe\_num\_ccj[which(test$num\_ccj > 0 & test$num\_ccj <= 1)] <- -0.9166

test$woe\_num\_ccj[which(test$num\_ccj > 1)] <- -1.1322

#Maximum arrears in previous 12 months:

train$woe\_max\_arrears\_12m<- rep(NA, length(train$max\_arrears\_12m))

train$woe\_max\_arrears\_12m[which(is.na(train$max\_arrears\_12m))] <- 0

train$woe\_max\_arrears\_12m[which(train$max\_arrears\_12m <= 0)] <- 0.7027

train$woe\_max\_arrears\_12m[which(train$max\_arrears\_12m > 0 & train$max\_arrears\_12m <= 1)] <- -0.8291

train$woe\_max\_arrears\_12m[which(train$max\_arrears\_12m > 1 & train$max\_arrears\_12m <= 1.4)] <- -1.1908

train$woe\_max\_arrears\_12m[which(train$max\_arrears\_12m > 1.4)] <- -2.2223

test$woe\_max\_arrears\_12m<- rep(NA, length(test$max\_arrears\_12m))

test$woe\_max\_arrears\_12m[which(is.na(test$max\_arrears\_12m))] <- 0

test$woe\_max\_arrears\_12m[which(test$max\_arrears\_12m <= 0)] <- 0.7027

test$woe\_max\_arrears\_12m[which(test$max\_arrears\_12m > 0 & test$max\_arrears\_12m <= 1)] <- -0.8291

test$woe\_max\_arrears\_12m[which(test$max\_arrears\_12m > 1 & test$max\_arrears\_12m <= 1.4)] <- -1.1908

test$woe\_max\_arrears\_12m[which(test$max\_arrears\_12m > 1.4)] <- -2.2223

#Maximum arrears balance in previous 6 months:

train$woe\_max\_arrears\_bal\_6m<- rep(NA, length(train$max\_arrears\_bal\_6m))

train$woe\_max\_arrears\_bal\_6m[which(is.na(train$max\_arrears\_bal\_6m))] <- 0

train$woe\_max\_arrears\_bal\_6m[which(train$max\_arrears\_bal\_6m <= 0)] <- 0.5771

train$woe\_max\_arrears\_bal\_6m[which(train$max\_arrears\_bal\_6m > 0 & train$max\_arrears\_bal\_6m <= 300)] <- -0.7818

train$woe\_max\_arrears\_bal\_6m[which(train$max\_arrears\_bal\_6m > 300 & train$max\_arrears\_bal\_6m <= 600)] <- -1.2958

train$woe\_max\_arrears\_bal\_6m[which(train$max\_arrears\_bal\_6m > 600 & train$max\_arrears\_bal\_6m <= 900)] <- -1.5753

train$woe\_max\_arrears\_bal\_6m[which(train$max\_arrears\_bal\_6m > 900)] <- -2.2110

test$woe\_max\_arrears\_bal\_6m<- rep(NA, length(test$max\_arrears\_bal\_6m))

test$woe\_max\_arrears\_bal\_6m[which(is.na(test$max\_arrears\_bal\_6m))] <- 0

test$woe\_max\_arrears\_bal\_6m[which(test$max\_arrears\_bal\_6m <= 0)] <- 0.5771

test$woe\_max\_arrears\_bal\_6m[which(test$max\_arrears\_bal\_6m > 0 & test$max\_arrears\_bal\_6m <= 300)] <- -0.7818

test$woe\_max\_arrears\_bal\_6m[which(test$max\_arrears\_bal\_6m > 300 & test$max\_arrears\_bal\_6m <= 600)] <- -1.2958

test$woe\_max\_arrears\_bal\_6m[which(test$max\_arrears\_bal\_6m > 600 & test$max\_arrears\_bal\_6m <= 900)] <- -1.5753

test$woe\_max\_arrears\_bal\_6m[which(test$max\_arrears\_bal\_6m > 900)] <- -2.2110

#Employment length (years):

train$woe\_emp\_length<- rep(NA, length(train$emp\_length))

train$woe\_emp\_length[which(is.na(train$emp\_length))] <- 0

train$woe\_emp\_length[which(train$emp\_length <= 2)] <- -0.7514

train$woe\_emp\_length[which(train$emp\_length > 2 & train$emp\_length <= 4)] <- -0.3695

train$woe\_emp\_length[which(train$emp\_length > 4 & train$emp\_length <= 7)] <- 0.1783

train$woe\_emp\_length[which(train$emp\_length > 7)] <- 0.5827

test$woe\_emp\_length<- rep(NA, length(test$emp\_length))

test$woe\_emp\_length[which(is.na(test$emp\_length))] <- 0

test$woe\_emp\_length[which(test$emp\_length <= 2)] <- -0.7514

test$woe\_emp\_length[which(test$emp\_length > 2 & test$emp\_length <= 4)] <- -0.3695

test$woe\_emp\_length[which(test$emp\_length > 4 & test$emp\_length <= 7)] <- 0.1783

test$woe\_emp\_length[which(test$emp\_length > 7)] <- 0.5827

#Months since recent CC delinquency:

train$woe\_months\_since\_recent\_cc\_delinq<- rep(NA, length(train$months\_since\_recent\_cc\_delinq))

train$woe\_months\_since\_recent\_cc\_delinq[which(is.na(train$months\_since\_recent\_cc\_delinq))] <- 0

train$woe\_months\_since\_recent\_cc\_delinq[which(train$months\_since\_recent\_cc\_delinq <= 6)] <- -0.4176

train$woe\_months\_since\_recent\_cc\_delinq[which(train$months\_since\_recent\_cc\_delinq > 6 & train$months\_since\_recent\_cc\_delinq <= 11)] <- -0.1942

train$woe\_months\_since\_recent\_cc\_delinq[which(train$months\_since\_recent\_cc\_delinq > 11)] <- 1.3166

test$woe\_months\_since\_recent\_cc\_delinq<- rep(NA, length(test$months\_since\_recent\_cc\_delinq))

test$woe\_months\_since\_recent\_cc\_delinq[which(is.na(test$months\_since\_recent\_cc\_delinq))] <- 0

test$woe\_months\_since\_recent\_cc\_delinq[which(test$months\_since\_recent\_cc\_delinq <= 6)] <- -0.4176

test$woe\_months\_since\_recent\_cc\_delinq[which(test$months\_since\_recent\_cc\_delinq > 6 & test$months\_since\_recent\_cc\_delinq <= 11)] <- -0.1942

test$woe\_months\_since\_recent\_cc\_delinq[which(test$months\_since\_recent\_cc\_delinq > 11)] <- 1.3166

#Annual income:

train$woe\_annual\_income<- rep(NA, length(train$annual\_income))

train$woe\_annual\_income[which(is.na(train$annual\_income))] <- 0

train$woe\_annual\_income[which(train$annual\_income <= 35064)] <- -1.8243

train$woe\_annual\_income[which(train$annual\_income > 35064 & train$annual\_income <= 41999)] <- -0.8272

train$woe\_annual\_income[which(train$annual\_income > 41999 & train$annual\_income <= 50111)] <- -0.3294

train$woe\_annual\_income[which(train$annual\_income > 50111 & train$annual\_income <= 65050)] <- 0.2379

train$woe\_annual\_income[which(train$annual\_income > 65050)] <- 0.6234

test$woe\_annual\_income<- rep(NA, length(test$annual\_income))

test$woe\_annual\_income[which(is.na(test$annual\_income))] <- 0

test$woe\_annual\_income[which(test$annual\_income <= 35064)] <- -1.8243

test$woe\_annual\_income[which(test$annual\_income > 35064 & test$annual\_income <= 41999)] <- -0.8272

test$woe\_annual\_income[which(test$annual\_income > 41999 & test$annual\_income <= 50111)] <- -0.3294

test$woe\_annual\_income[which(test$annual\_income > 50111 & test$annual\_income <= 65050)] <- 0.2379

test$woe\_annual\_income[which(test$annual\_income > 65050)] <- 0.6234

Example 2.3.6

# 2.3 Plot performance metrics

par(mfrow=c(2,1))

hist(unlist(auc\_vector), xlab = 'AUC', ylab = 'Frequency', main = 'AUC Distribution: 20 Folds', col = 'royal blue')

hist(unlist(gini\_vector), xlab = 'Gini', ylab = 'Frequency', main = 'Gini Distribution: 20 Folds', col = 'dark magenta')

hist(unlist(ks\_vector), xlab = 'KS', ylab = 'Frequency', main = 'KS Distribution: 20 Folds', col = 'light green')

detach(data\_subset)

Example 2.4.2

# Create default flag as per Example 2.3.1

# From "default\_event" derive "default\_indicator" as "Yes" "No" as per Example 2.4.1

# 1.1.1. Glimpse

dplyr::glimpse(oneypd\_tree)

# 1.1.2. Date format

library(vars)

oneypd\_tree <- dplyr::mutate\_at(oneypd\_tree, vars(contains('date')),

funs(as.Date))

class(oneypd\_tree$origination\_date)

# 1.1.3. Round arrears count fields

oneypd\_tree$max\_arrears\_12m<- round(oneypd\_tree$max\_arrears\_12m,4)

oneypd\_tree$arrears\_months<- round(oneypd\_tree$arrears\_months,4)

# 1.1.4. Default flag definition

oneypd\_tree<- dplyr::mutate(oneypd\_tree,

default\_event = if\_else(oneypd\_tree$arrears\_event == 1 |

oneypd\_tree$term\_expiry\_event == 1 |

oneypd\_tree$bankrupt\_event == 1, 1,0))

# 1.1.5. Database split in train and test samples

# Recode default event variables for more convenient use

# 0-default, 1-non-default

oneypd\_tree$default\_flag<-

dplyr::if\_else(oneypd\_tree$default\_event == 1,0,1)

default\_indicator=ifelse(oneypd\_tree$default\_event==0,"No","Yes")

oneypd\_tree=data.frame(oneypd\_tree,default\_indicator)