BonusAssignment2\_Seshadri

Sri Seshadri

3/11/2018

# Objective:

Study the effect on coefficients of a model due to repeated resampling of training and validation samples.

# Executive Summary

The coefficients of a model trained on, one time sampling of training data, is biased. Its is necessary to train and test the model in multiple cross validated samples to get an unbiased estimate of the coefficients.

# Analysis

## Explore and Set up the data set

The structure of GermanCredit data set is explored below. It looks like there are several binary variables that are of the type numeric. The binary variables are converted into factors. Also there are couple of features that have only one unique value, those featuers are removed from the data for modeling. It’ll also be useful to scale and center the data for easier plotting of multiple coefficients in the same plot

data("GermanCredit")  
skimr::skim(GermanCredit)

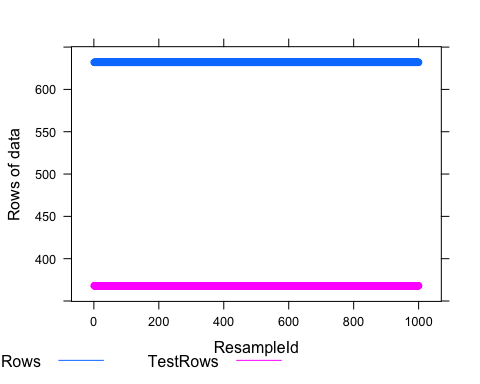
## Skim summary statistics  
## n obs: 1000   
## n variables: 62   
##   
## Variable type: factor   
## variable missing complete n n\_unique top\_counts ordered  
## Class 0 1000 1000 2 Goo: 700, Bad: 300, NA: 0 FALSE  
##   
## Variable type: integer   
## variable missing complete n mean sd p0  
## Age 0 1000 1000 35.55 11.38 19  
## Amount 0 1000 1000 3271.26 2822.74 250  
## Duration 0 1000 1000 20.9 12.06 4  
## InstallmentRatePercentage 0 1000 1000 2.97 1.12 1  
## NumberExistingCredits 0 1000 1000 1.41 0.58 1  
## NumberPeopleMaintenance 0 1000 1000 1.16 0.36 1  
## ResidenceDuration 0 1000 1000 2.85 1.1 1  
## p25 median p75 p100 hist  
## 27 33 42 75 ▇▇▆▃▂▁▁▁  
## 1365.5 2319.5 3972.25 18424 ▇▃▂▁▁▁▁▁  
## 12 18 24 72 ▇▅▅▃▁▁▁▁  
## 2 3 4 4 ▂▁▃▁▁▂▁▇  
## 1 1 2 4 ▇▁▅▁▁▁▁▁  
## 1 1 1 2 ▇▁▁▁▁▁▁▂  
## 2 3 4 4 ▂▁▆▁▁▃▁▇  
##   
## Variable type: numeric   
## variable missing complete n mean sd  
## CheckingAccountStatus.0.to.200 0 1000 1000 0.27 0.44   
## CheckingAccountStatus.gt.200 0 1000 1000 0.063 0.24   
## CheckingAccountStatus.lt.0 0 1000 1000 0.27 0.45   
## CheckingAccountStatus.none 0 1000 1000 0.39 0.49   
## CreditHistory.Critical 0 1000 1000 0.29 0.46   
## CreditHistory.Delay 0 1000 1000 0.088 0.28   
## CreditHistory.NoCredit.AllPaid 0 1000 1000 0.04 0.2   
## CreditHistory.PaidDuly 0 1000 1000 0.53 0.5   
## CreditHistory.ThisBank.AllPaid 0 1000 1000 0.049 0.22   
## EmploymentDuration.1.to.4 0 1000 1000 0.34 0.47   
## EmploymentDuration.4.to.7 0 1000 1000 0.17 0.38   
## EmploymentDuration.gt.7 0 1000 1000 0.25 0.43   
## EmploymentDuration.lt.1 0 1000 1000 0.17 0.38   
## EmploymentDuration.Unemployed 0 1000 1000 0.062 0.24   
## ForeignWorker 0 1000 1000 0.96 0.19   
## Housing.ForFree 0 1000 1000 0.11 0.31   
## Housing.Own 0 1000 1000 0.71 0.45   
## Housing.Rent 0 1000 1000 0.18 0.38   
## Job.Management.SelfEmp.HighlyQualified 0 1000 1000 0.15 0.36   
## Job.SkilledEmployee 0 1000 1000 0.63 0.48   
## Job.UnemployedUnskilled 0 1000 1000 0.022 0.15   
## Job.UnskilledResident 0 1000 1000 0.2 0.4   
## OtherDebtorsGuarantors.CoApplicant 0 1000 1000 0.041 0.2   
## OtherDebtorsGuarantors.Guarantor 0 1000 1000 0.052 0.22   
## OtherDebtorsGuarantors.None 0 1000 1000 0.91 0.29   
## OtherInstallmentPlans.Bank 0 1000 1000 0.14 0.35   
## OtherInstallmentPlans.None 0 1000 1000 0.81 0.39   
## OtherInstallmentPlans.Stores 0 1000 1000 0.047 0.21   
## Personal.Female.NotSingle 0 1000 1000 0.31 0.46   
## Personal.Female.Single 0 1000 1000 0 0   
## Personal.Male.Divorced.Seperated 0 1000 1000 0.05 0.22   
## Personal.Male.Married.Widowed 0 1000 1000 0.092 0.29   
## Personal.Male.Single 0 1000 1000 0.55 0.5   
## Property.CarOther 0 1000 1000 0.33 0.47   
## Property.Insurance 0 1000 1000 0.23 0.42   
## Property.RealEstate 0 1000 1000 0.28 0.45   
## Property.Unknown 0 1000 1000 0.15 0.36   
## Purpose.Business 0 1000 1000 0.097 0.3   
## Purpose.DomesticAppliance 0 1000 1000 0.012 0.11   
## Purpose.Education 0 1000 1000 0.05 0.22   
## Purpose.Furniture.Equipment 0 1000 1000 0.18 0.39   
## Purpose.NewCar 0 1000 1000 0.23 0.42   
## Purpose.Other 0 1000 1000 0.012 0.11   
## Purpose.Radio.Television 0 1000 1000 0.28 0.45   
## Purpose.Repairs 0 1000 1000 0.022 0.15   
## Purpose.Retraining 0 1000 1000 0.009 0.094  
## Purpose.UsedCar 0 1000 1000 0.1 0.3   
## Purpose.Vacation 0 1000 1000 0 0   
## SavingsAccountBonds.100.to.500 0 1000 1000 0.1 0.3   
## SavingsAccountBonds.500.to.1000 0 1000 1000 0.063 0.24   
## SavingsAccountBonds.gt.1000 0 1000 1000 0.048 0.21   
## SavingsAccountBonds.lt.100 0 1000 1000 0.6 0.49   
## SavingsAccountBonds.Unknown 0 1000 1000 0.18 0.39   
## Telephone 0 1000 1000 0.6 0.49   
## p0 p25 median p75 p100 hist  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▃  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▃  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▅  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▃  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 1 1 1 ▇▁▁▁▁▁▁▇  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▅  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▃  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 1 1 1 1 ▁▁▁▁▁▁▁▇  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 1 1 1 ▃▁▁▁▁▁▁▇  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 1 1 1 ▅▁▁▁▁▁▁▇  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 1 1 1 1 ▁▁▁▁▁▁▁▇  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 1 1 1 1 ▂▁▁▁▁▁▁▇  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▃  
## 0 0 0 0 0 ▁▁▁▇▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 1 1 1 ▆▁▁▁▁▁▁▇  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▃  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▃  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 1 1 ▇▁▁▁▁▁▁▃  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 0 ▁▁▁▇▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▁  
## 0 0 1 1 1 ▅▁▁▁▁▁▁▇  
## 0 0 0 0 1 ▇▁▁▁▁▁▁▂  
## 0 0 1 1 1 ▆▁▁▁▁▁▁▇

integerVars <- GermanCredit %>% dplyr::select\_if(is.integer) %>% names  
Vars <- colnames(GermanCredit)  
tobeConvertedVars <- Vars[!Vars %in% integerVars]  
  
GermanCredit[tobeConvertedVars] <- lapply(GermanCredit[tobeConvertedVars], factor)  
UniqueVars <- Vars[caret::nearZeroVar(GermanCredit,uniqueCut = 0)]  
Vars2select <- Vars[!Vars %in% UniqueVars]  
GermanCredit <- GermanCredit %>% dplyr::select(!!Vars2select)  
  
trans <- caret::preProcess(GermanCredit,center = T, scale = T)  
GermanCredit <- predict(trans, GermanCredit)

The GermanCredit data in caret package is split into training and test in the ratio 63.2:36.8 repeatedly for 1000 times. The rsample package is used to split the data iteratively.

The the number of rows of data for each split is verified in figure 1.

set.seed(1)  
# Data sampled without replacement  
  
samples <- rsample::mc\_cv(data = GermanCredit,prop = 0.632,times = 1000)  
  
# Verify the dimensions of the first of the 1000 splits  
  
SampleVerification <- purrr::map\_df(1:1000,.f = function(x) data.frame(ResampleId = x,TrainRows = nrow(analysis(samples$splits[[x]])), TestRows = nrow(assessment(samples$splits[[x]]))) )  
  
lattice::xyplot(TrainRows + TestRows ~ ResampleId,data = SampleVerification,ylab = "Rows of data",auto.key=list(space="bottom", columns=4, cex.title=1,lines=TRUE, points=FALSE, box = T))



Verify split

## Linear model fits and evaluation.

Linear model is fit on the training sets and the model is evaluated on the test sets for each of the 1000 train-test samples. Those variables that have only one unique value are removed as predictors.

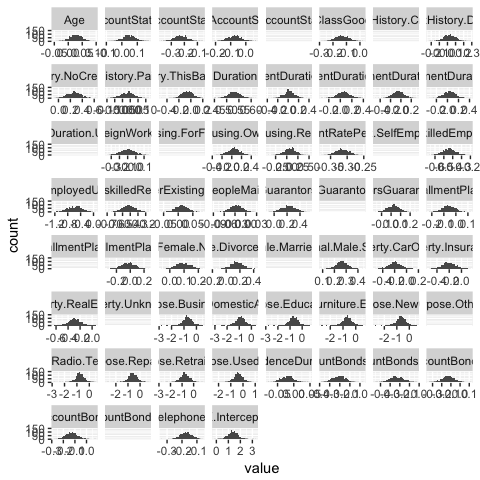
# Function to get model stats  
ModelSummaries <- function(split){  
 # get the train and test sets from the split  
 Train <- rsample::analysis(split$splits[[1]])  
 Test <- rsample::assessment(split$splits[[1]])  
 Vars <- colnames(Train)  
 # get variables that have only one unique value and remove them  
 UniqueVars <- caret::nearZeroVar(Train,uniqueCut = 0)  
 if(length(UniqueVars) > 0){  
 Train <- Train[,UniqueVars\*-1]  
 Test <- Test[,UniqueVars\*-1]  
 }  
   
   
 # linear model fit  
 lm.fit <- lm(Amount ~ ., data = Train)  
 coefs <- coefficients(lm.fit)  
 AdjR2.Train <- summary(lm.fit)$adj.r.squared  
 yPred <- predict(lm.fit,newdata = Test)  
 R2.test <- cor(yPred,Test$Amount)^2  
 return(list(Betas = coefs,Train.R2 = AdjR2.Train, Test.R2 = R2.test))  
}  
  
ModelStats <- purrr::map(1:1000,function(x) ModelSummaries(samples[x,]))  
  
# Check if named vectors of Beta coefficients are the same for all the 1000 splits  
lengthVer <- sapply(1:1000,function(x) names(ModelStats[[x]]$Betas))  
test2 <- sapply(1:60, function(x) {length(unique(lengthVer[x,]))})  
#plot(test2)  
  
# Retrieve Beta coefficents as dataframe  
  
df.coef <- sapply(1:1000, function(x){ModelStats[[x]]$Betas})  
df.coef <- t(df.coef)  
  
Rsquared <- data.frame(Train = train <- sapply(1:1000, function(x){ModelStats[[x]]$Train.R2}),  
 Test = sapply(1:1000, function(x){ModelStats[[x]]$Test.R2}))  
  
Rsquared <- Rsquared %>% mutate(PercFall = (Train - Test)/Train)

Figure 2 shows the scaled coefficients of the predictors.

df.coef <- data.frame(df.coef)  
df.coef.plotdf <- tidyr::gather(df.coef)  
ggplot(data = df.coef.plotdf, mapping = aes(x=value)) + geom\_histogram() + facet\_wrap(~key, scales = 'free\_x')

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 11000 rows containing non-finite values (stat\_bin).

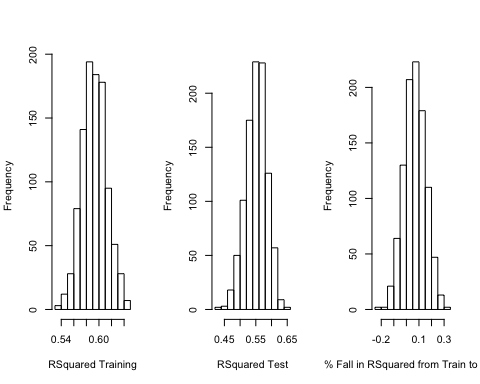


Scaled Coefficients plot

Figure 3 shows the distribution of R-Squared of Training and Test data and as well the % fall in R-Squared from training to test.

It can be seen that the % Fall in R Squared is within a range of -20% to 30%.

par(mfrow=c(1,3))  
hist(Rsquared$Train,xlab = "RSquared Training", main="")  
hist(Rsquared$Test,xlab = "RSquared Test", main="")  
hist(Rsquared$PercFall, xlab = "% Fall in RSquared from Train to Test", main="")



library(Rmisc)

## Loading required package: plyr

## -------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.  
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:  
## library(plyr); library(dplyr)

## -------------------------------------------------------------------------

##   
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

Stats <- data.frame(t(sapply(1:60, function(x){CI(df.coef[,x])}))) %>% dplyr::mutate(Predictor = colnames(df.coef)) %>%   
 dplyr::select(Predictor,lower,mean,upper)  
  
trainSamples <- sample(1000,632)  
#nearZeroVar(GermanCredit[train,],uniqueCut = 0)  
lmfit.once <- lm(Amount ~ ., data = GermanCredit[trainSamples,])  
coefs.omce <- coef(lmfit.once)  
Stats <- Stats %>% dplyr::mutate(OneTimeSplitCoef = coefs.omce, WithinCI = if\_else(coefs.omce >= lower & coefs.omce <= upper,"Within","Outside"))  
  
knitr::kable(Stats, caption = "Comparison of single coefficient to averge of 1000 fits")

Comparison of single coefficient to averge of 1000 fits

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Predictor | lower | mean | upper | OneTimeSplitCoef | WithinCI |
| X.Intercept. | 1.3610675 | 1.3883982 | 1.4157290 | 1.0076342 | Outside |
| Duration | 0.5293622 | 0.5309347 | 0.5325073 | 0.4872938 | Outside |
| InstallmentRatePercentage | -0.3165723 | -0.3153660 | -0.3141596 | -0.3045645 | Outside |
| ResidenceDuration | -0.0205579 | -0.0193493 | -0.0181408 | -0.0012570 | Outside |
| Age | 0.0255229 | 0.0268692 | 0.0282155 | 0.0187196 | Outside |
| NumberExistingCredits | 0.0110226 | 0.0123172 | 0.0136119 | 0.0197262 | Outside |
| NumberPeopleMaintenance | -0.0275179 | -0.0264781 | -0.0254383 | -0.0548516 | Outside |
| Telephone1 | -0.1750750 | -0.1726407 | -0.1702064 | -0.1299213 | Outside |
| ForeignWorker1 | -0.1040195 | -0.0986412 | -0.0932629 | -0.0122258 | Outside |
| ClassGood | -0.1336153 | -0.1305027 | -0.1273901 | -0.0976242 | Outside |
| CheckingAccountStatus.lt.01 | -0.0748525 | -0.0719788 | -0.0691051 | 0.0200919 | Outside |
| CheckingAccountStatus.0.to.2001 | 0.0447519 | 0.0473957 | 0.0500395 | 0.1019934 | Outside |
| CheckingAccountStatus.gt.2001 | -0.2290817 | -0.2258868 | -0.2226919 | -0.2629068 | Outside |
| CheckingAccountStatus.none1 | NA | NA | NA | NA | NA |
| CreditHistory.NoCredit.AllPaid1 | 0.2535960 | 0.2606260 | 0.2676560 | 0.0358525 | Outside |
| CreditHistory.ThisBank.AllPaid1 | -0.0628903 | -0.0569821 | -0.0510739 | -0.0489794 | Outside |
| CreditHistory.PaidDuly1 | -0.0244654 | -0.0217219 | -0.0189783 | 0.0864416 | Outside |
| CreditHistory.Delay1 | 0.0202810 | 0.0249907 | 0.0297005 | 0.1181631 | Outside |
| CreditHistory.Critical1 | NA | NA | NA | NA | NA |
| Purpose.NewCar1 | -0.6661461 | -0.6420297 | -0.6179133 | -0.6664775 | Outside |
| Purpose.UsedCar1 | -0.4125948 | -0.3884939 | -0.3643930 | -0.2902017 | Outside |
| Purpose.Furniture.Equipment1 | -0.6794603 | -0.6552790 | -0.6310976 | -0.7004469 | Outside |
| Purpose.Radio.Television1 | -0.7600349 | -0.7360964 | -0.7121579 | -0.7989457 | Outside |
| Purpose.DomesticAppliance1 | -0.9098339 | -0.8848707 | -0.8599074 | -0.9329383 | Outside |
| Purpose.Repairs1 | -0.6422480 | -0.6173233 | -0.5923987 | -0.8524553 | Outside |
| Purpose.Education1 | -0.7181040 | -0.6938231 | -0.6695422 | -0.6701855 | Within |
| Purpose.Retraining1 | -0.7903655 | -0.7640032 | -0.7376408 | -1.0415600 | Outside |
| Purpose.Business1 | -0.7283467 | -0.7041555 | -0.6799643 | -0.7783022 | Outside |
| Purpose.Other1 | NA | NA | NA | NA | NA |
| SavingsAccountBonds.lt.1001 | -0.1316095 | -0.1284673 | -0.1253251 | -0.0652545 | Outside |
| SavingsAccountBonds.100.to.5001 | -0.2103486 | -0.2063924 | -0.2024363 | -0.2305228 | Outside |
| SavingsAccountBonds.500.to.10001 | -0.2353459 | -0.2309910 | -0.2266360 | -0.1632795 | Outside |
| SavingsAccountBonds.gt.10001 | -0.1345730 | -0.1303231 | -0.1260732 | -0.0734524 | Outside |
| SavingsAccountBonds.Unknown1 | NA | NA | NA | NA | NA |
| EmploymentDuration.lt.11 | 0.0320726 | 0.0387220 | 0.0453713 | 0.0887161 | Outside |
| EmploymentDuration.1.to.41 | 0.0139163 | 0.0206067 | 0.0272970 | 0.0880706 | Outside |
| EmploymentDuration.4.to.71 | 0.0470244 | 0.0540457 | 0.0610669 | 0.2346389 | Outside |
| EmploymentDuration.gt.71 | -0.0595789 | -0.0527727 | -0.0459665 | 0.0612719 | Outside |
| EmploymentDuration.Unemployed1 | NA | NA | NA | NA | NA |
| Personal.Male.Divorced.Seperated1 | 0.1474485 | 0.1528003 | 0.1581521 | 0.2172886 | Outside |
| Personal.Female.NotSingle1 | 0.0909682 | 0.0938667 | 0.0967652 | 0.0448338 | Outside |
| Personal.Male.Single1 | 0.2588820 | 0.2619492 | 0.2650164 | 0.2710714 | Outside |
| Personal.Male.Married.Widowed1 | NA | NA | NA | NA | NA |
| OtherDebtorsGuarantors.None1 | 0.0228669 | 0.0267009 | 0.0305350 | 0.0014942 | Outside |
| OtherDebtorsGuarantors.CoApplicant1 | 0.2226994 | 0.2291624 | 0.2356255 | 0.3369058 | Outside |
| OtherDebtorsGuarantors.Guarantor1 | NA | NA | NA | NA | NA |
| Property.RealEstate1 | -0.2927526 | -0.2862939 | -0.2798351 | -0.2358026 | Outside |
| Property.Insurance1 | -0.2098111 | -0.2031548 | -0.1964984 | -0.1448284 | Outside |
| Property.CarOther1 | -0.2010101 | -0.1944734 | -0.1879366 | -0.0916335 | Outside |
| Property.Unknown1 | NA | NA | NA | NA | NA |
| OtherInstallmentPlans.Bank1 | -0.0624438 | -0.0592244 | -0.0560049 | -0.0850008 | Outside |
| OtherInstallmentPlans.Stores1 | -0.0334157 | -0.0275384 | -0.0216611 | -0.0317454 | Within |
| OtherInstallmentPlans.None1 | NA | NA | NA | NA | NA |
| Housing.Rent1 | 0.0666029 | 0.0746474 | 0.0826919 | 0.0461640 | Outside |
| Housing.Own1 | 0.0369494 | 0.0448396 | 0.0527299 | 0.0792720 | Outside |
| Housing.ForFree1 | NA | NA | NA | NA | NA |
| Job.UnemployedUnskilled1 | -0.5949737 | -0.5818403 | -0.5687068 | -0.3422740 | Outside |
| Job.UnskilledResident1 | -0.4269291 | -0.4219083 | -0.4168875 | -0.5008756 | Outside |
| Job.SkilledEmployee1 | -0.4456681 | -0.4411288 | -0.4365895 | -0.4424048 | Within |
| Job.Management.SelfEmp.HighlyQualified1 | NA | NA | NA | NA | NA |

table(Stats$WithinCI)

##   
## Outside Within   
## 46 3

# Conclusion

It is seen that 47 of 49 coeffiicents of the one time split model are outside of the confidence interval. This is due to the 1000 samples of each coefficients making the confidence intterval tight. The one time sample coefficient is a biased estimate, whereas the avergae of 1000 samples of each coefficent is unbiased.