Homework 3.2 - Resampling

Hamed

2/23/2020

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(rsample)

## Loading required package: tidyr

library(purrr)

##   
## Attaching package: 'purrr'

## The following object is masked from 'package:caret':  
##   
## lift

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(ggplot2)  
library(scales)

##   
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':  
##   
## discard

library(mlbench)  
library(kernlab)

##   
## Attaching package: 'kernlab'

## The following object is masked from 'package:scales':  
##   
## alpha

## The following object is masked from 'package:purrr':  
##   
## cross

## The following object is masked from 'package:ggplot2':  
##   
## alpha

library(sessioninfo)  
theme\_set(theme\_bw())  
library(RCurl)

## Loading required package: bitops

##   
## Attaching package: 'RCurl'

## The following object is masked from 'package:tidyr':  
##   
## complete

## Get the German Credit dataset from the UCI machine learning repository

UCI\_german<-getURL("http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data")  
  
names<-c('checking\_account','month','Credit\_history','Purpose',  
 'Credit\_amount','Savings','employment','Installment\_rate',  
 'status','Other\_debtors','Present\_residence\_since','Property',  
 'Age','installment\_plans','Housing','#credits','Job','#people',  
 'Telephone','foreign','Cost\_Matrix')  
  
credit\_data<-read.table(textConnection(UCI\_german),sep=" ",  
 col.names=names)  
str(credit\_data)

## 'data.frame': 1000 obs. of 21 variables:  
## $ checking\_account : Factor w/ 4 levels "A11","A12","A13",..: 1 2 4 1 1 4 4 2 4 2 ...  
## $ month : int 6 48 12 42 24 36 24 36 12 30 ...  
## $ Credit\_history : Factor w/ 5 levels "A30","A31","A32",..: 5 3 5 3 4 3 3 3 3 5 ...  
## $ Purpose : Factor w/ 10 levels "A40","A41","A410",..: 5 5 8 4 1 8 4 2 5 1 ...  
## $ Credit\_amount : int 1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...  
## $ Savings : Factor w/ 5 levels "A61","A62","A63",..: 5 1 1 1 1 5 3 1 4 1 ...  
## $ employment : Factor w/ 5 levels "A71","A72","A73",..: 5 3 4 4 3 3 5 3 4 1 ...  
## $ Installment\_rate : int 4 2 2 2 3 2 3 2 2 4 ...  
## $ status : Factor w/ 4 levels "A91","A92","A93",..: 3 2 3 3 3 3 3 3 1 4 ...  
## $ Other\_debtors : Factor w/ 3 levels "A101","A102",..: 1 1 1 3 1 1 1 1 1 1 ...  
## $ Present\_residence\_since: int 4 2 3 4 4 4 4 2 4 2 ...  
## $ Property : Factor w/ 4 levels "A121","A122",..: 1 1 1 2 4 4 2 3 1 3 ...  
## $ Age : int 67 22 49 45 53 35 53 35 61 28 ...  
## $ installment\_plans : Factor w/ 3 levels "A141","A142",..: 3 3 3 3 3 3 3 3 3 3 ...  
## $ Housing : Factor w/ 3 levels "A151","A152",..: 2 2 2 3 3 3 2 1 2 2 ...  
## $ X.credits : int 2 1 1 1 2 1 1 1 1 2 ...  
## $ Job : Factor w/ 4 levels "A171","A172",..: 3 3 2 3 3 2 3 4 2 4 ...  
## $ X.people : int 1 1 2 2 2 2 1 1 1 1 ...  
## $ Telephone : Factor w/ 2 levels "A191","A192": 2 1 1 1 1 2 1 2 1 1 ...  
## $ foreign : Factor w/ 2 levels "A201","A202": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Cost\_Matrix : int 1 2 1 1 2 1 1 1 1 2 ...

# Fit a SVM model to predict the type of credit (good or bad) with the following resampling techniques:

## 5 fold cross validation

# convert class variable into factor   
credit\_data$Cost\_Matrix=as.factor(credit\_data$Cost\_Matrix)  
  
# First split the dataset , 85% Training data ad 15% testing data  
set.seed(123)  
smp\_size <- floor(0.85 \* nrow(credit\_data))  
train\_ind <- sample(seq\_len(nrow(credit\_data)), size = smp\_size)  
train.credit <- credit\_data[train\_ind, ]  
test.credit <- credit\_data[-train\_ind, ]  
  
  
dim(train.credit) #shape of the train data

## [1] 850 21

dim(test.credit) #shape of test data

## [1] 150 21

#Modelling  
train\_control <- trainControl(method="cv", number=5)  
model <- train(Cost\_Matrix~., data=train.credit, trControl=train\_control, method="svmLinear")  
# summarize results  
print(model)

## Support Vector Machines with Linear Kernel   
##   
## 850 samples  
## 20 predictor  
## 2 classes: '1', '2'   
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold)   
## Summary of sample sizes: 681, 680, 680, 680, 679   
## Resampling results:  
##   
## Accuracy Kappa   
## 0.7434911 0.3315135  
##   
## Tuning parameter 'C' was held constant at a value of 1

# prediction  
pred=predict(model,newdata = test.credit)  
pred

## [1] 1 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2 2 2 1 2 1 1  
## [38] 1 1 2 1 1 1 2 1 1 1 1 1 2 1 1 2 2 1 2 1 1 2 2 2 1 1 2 1 1 1 1 1 1 2 1 1 1  
## [75] 1 1 2 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 2 1 2 1 1 2 2 1 2 2 1 1 1 1 2 1 2 2 1  
## [112] 1 2 1 1 1 1 1 1 2 1 1 2 2 1 1 2 2 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1  
## [149] 1 1  
## Levels: 1 2

# Show the accuracy for the training dataset  
model$results

## C Accuracy Kappa AccuracySD KappaSD  
## 1 1 0.7434911 0.3315135 0.02633049 0.06254558

# Show the accuracy results for each resample  
model$resample

## Accuracy Kappa Resample  
## 1 0.7159763 0.2658824 Fold1  
## 2 0.7588235 0.3938076 Fold2  
## 3 0.7176471 0.2673730 Fold3  
## 4 0.7764706 0.3883734 Fold4  
## 5 0.7485380 0.3421312 Fold5

# Show the accuracy for the test dataset  
mean(pred == test.credit$Cost\_Matrix)

## [1] 0.76

## 10 fold cross-validation with 3 repeats

# define training control  
train\_control2 <- trainControl(method="repeatedcv", number=10, repeats=3)  
model2<-train(Cost\_Matrix~.,data=train.credit,trControl=train\_control2, method="svmLinear")  
  
# summarize results  
print(model2)

## Support Vector Machines with Linear Kernel   
##   
## 850 samples  
## 20 predictor  
## 2 classes: '1', '2'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold, repeated 3 times)   
## Summary of sample sizes: 765, 765, 766, 764, 765, 764, ...   
## Resampling results:  
##   
## Accuracy Kappa   
## 0.7431785 0.3352325  
##   
## Tuning parameter 'C' was held constant at a value of 1

pred2=predict(model2,newdata = test.credit)  
pred2

## [1] 1 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2 2 2 1 2 1 1  
## [38] 1 1 2 1 1 1 2 1 1 1 1 1 2 1 1 2 2 1 2 1 1 2 2 2 1 1 2 1 1 1 1 1 1 2 1 1 1  
## [75] 1 1 2 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 2 1 2 1 1 2 2 1 2 2 1 1 1 1 2 1 2 2 1  
## [112] 1 2 1 1 1 1 1 1 2 1 1 2 2 1 1 2 2 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1  
## [149] 1 1  
## Levels: 1 2

# Show the accuracy for the training dataset  
model2$results

## C Accuracy Kappa AccuracySD KappaSD  
## 1 1 0.7431785 0.3352325 0.04874966 0.1190653

# Show the accuracy results for each resample  
model2$resample

## Accuracy Kappa Resample  
## 1 0.7411765 0.37666667 Fold01.Rep1  
## 2 0.7411765 0.28951368 Fold02.Rep1  
## 3 0.7738095 0.33830846 Fold03.Rep1  
## 4 0.7441860 0.36467428 Fold04.Rep1  
## 5 0.7529412 0.35208711 Fold05.Rep1  
## 6 0.7325581 0.34372926 Fold06.Rep1  
## 7 0.7380952 0.32456140 Fold07.Rep1  
## 8 0.7058824 0.24778761 Fold08.Rep1  
## 9 0.7023810 0.24242424 Fold09.Rep1  
## 10 0.7790698 0.43066202 Fold10.Rep1  
## 11 0.7647059 0.35410334 Fold01.Rep2  
## 12 0.8255814 0.53895640 Fold02.Rep2  
## 13 0.6904762 0.29457364 Fold03.Rep2  
## 14 0.7764706 0.41379310 Fold04.Rep2  
## 15 0.6279070 0.02893437 Fold05.Rep2  
## 16 0.7500000 0.34666667 Fold06.Rep2  
## 17 0.7411765 0.31250000 Fold07.Rep2  
## 18 0.6941176 0.22727273 Fold08.Rep2  
## 19 0.7906977 0.46730902 Fold09.Rep2  
## 20 0.7857143 0.41666667 Fold10.Rep2  
## 21 0.7976190 0.48484848 Fold01.Rep3  
## 22 0.7294118 0.30796460 Fold02.Rep3  
## 23 0.7882353 0.46391030 Fold03.Rep3  
## 24 0.7882353 0.44964029 Fold04.Rep3  
## 25 0.7325581 0.27439472 Fold05.Rep3  
## 26 0.6860465 0.21074099 Fold06.Rep3  
## 27 0.6352941 0.08981002 Fold07.Rep3  
## 28 0.7857143 0.43243243 Fold08.Rep3  
## 29 0.6823529 0.14525140 Fold09.Rep3  
## 30 0.8117647 0.48679245 Fold10.Rep3

# Show the accuracy for the test dataset  
mean(pred2 == test.credit$Cost\_Matrix)

## [1] 0.76

## Leave group out with 5 iterations and 85% data used for training define training control

# Modelling  
train\_control3 <- trainControl(method="LOOCV",number = 5)  
model3<-train(Cost\_Matrix~.,data=train.credit,trControl=train\_control3, method="svmLinear")  
  
# summarize results  
print(model3)

## Support Vector Machines with Linear Kernel   
##   
## 850 samples  
## 20 predictor  
## 2 classes: '1', '2'   
##   
## No pre-processing  
## Resampling: Leave-One-Out Cross-Validation   
## Summary of sample sizes: 849, 849, 849, 849, 849, 849, ...   
## Resampling results:  
##   
## Accuracy Kappa   
## 0.7352941 0.3017474  
##   
## Tuning parameter 'C' was held constant at a value of 1

# Prediction  
pred3=predict(model3,newdata=test.credit)  
pred3

## [1] 1 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2 2 2 1 2 1 1  
## [38] 1 1 2 1 1 1 2 1 1 1 1 1 2 1 1 2 2 1 2 1 1 2 2 2 1 1 2 1 1 1 1 1 1 2 1 1 1  
## [75] 1 1 2 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 2 1 2 1 1 2 2 1 2 2 1 1 1 1 2 1 2 2 1  
## [112] 1 2 1 1 1 1 1 1 2 1 1 2 2 1 1 2 2 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1  
## [149] 1 1  
## Levels: 1 2

# Show the accuracy for the training dataset  
model3$results

## C Accuracy Kappa  
## 1 1 0.7352941 0.3017474

# Show the accuracy results for each resample  
model3$resample

## NULL

# Show the accuracy for the test dataset  
mean(pred3 == test.credit$Cost\_Matrix)

## [1] 0.76

## Bootstrap with 25 iterations

# Modeling  
train\_control4 <- trainControl(method="boot", number=25)  
model4 <- train(Cost\_Matrix~.,data=train.credit,trControl=train\_control4,method="svmLinear")  
  
# summarize results  
print(model4)

## Support Vector Machines with Linear Kernel   
##   
## 850 samples  
## 20 predictor  
## 2 classes: '1', '2'   
##   
## No pre-processing  
## Resampling: Bootstrapped (25 reps)   
## Summary of sample sizes: 850, 850, 850, 850, 850, 850, ...   
## Resampling results:  
##   
## Accuracy Kappa   
## 0.7306301 0.3069666  
##   
## Tuning parameter 'C' was held constant at a value of 1

# Prediction  
pred4=predict(model4,newdata=test.credit)  
pred4

## [1] 1 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2 2 2 1 2 1 1  
## [38] 1 1 2 1 1 1 2 1 1 1 1 1 2 1 1 2 2 1 2 1 1 2 2 2 1 1 2 1 1 1 1 1 1 2 1 1 1  
## [75] 1 1 2 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 2 1 2 1 1 2 2 1 2 2 1 1 1 1 2 1 2 2 1  
## [112] 1 2 1 1 1 1 1 1 2 1 1 2 2 1 1 2 2 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1  
## [149] 1 1  
## Levels: 1 2

# Show the accuracy for the training dataset  
model4$results

## C Accuracy Kappa AccuracySD KappaSD  
## 1 1 0.7306301 0.3069666 0.02380917 0.04811941

# Show the accuracy results for each resample  
model4$resample

## Accuracy Kappa Resample  
## 1 0.7418301 0.3426334 Resample01  
## 2 0.7365079 0.3141576 Resample02  
## 3 0.7092652 0.2830317 Resample03  
## 4 0.7223975 0.2874949 Resample04  
## 5 0.7289720 0.3289844 Resample05  
## 6 0.7507599 0.3914279 Resample06  
## 7 0.7361564 0.3141462 Resample07  
## 8 0.7260274 0.3169591 Resample08  
## 9 0.6902357 0.2460265 Resample09  
## 10 0.7340067 0.2906122 Resample10  
## 11 0.7450331 0.3042307 Resample11  
## 12 0.7662338 0.3550113 Resample12  
## 13 0.6903226 0.2052980 Resample13  
## 14 0.7419355 0.3113024 Resample14  
## 15 0.7378641 0.3083810 Resample15  
## 16 0.6928105 0.2702826 Resample16  
## 17 0.7558528 0.3509471 Resample17  
## 18 0.7753165 0.3800840 Resample18  
## 19 0.7420382 0.3303670 Resample19  
## 20 0.7093750 0.2533869 Resample20  
## 21 0.7109635 0.2619846 Resample21  
## 22 0.7254902 0.2923297 Resample22  
## 23 0.7719870 0.4098748 Resample23  
## 24 0.7188498 0.2569733 Resample24  
## 25 0.7055215 0.2682379 Resample25

# Show the accuracy for the test dataset  
mean(pred4 == test.credit$Cost\_Matrix)

## [1] 0.76

# Which resampling method gives the best estimate of the test error?

* Selected the method with the highest test accuracy as it was the best

#5 Fold cross-validation  
mean(pred == test.credit$Cost\_Matrix)

## [1] 0.76

# 10 fold cross-validation  
mean(pred2 == test.credit$Cost\_Matrix)

## [1] 0.76

# Leave group out with 5 iterations and 85% data used for training define training control  
mean(pred4 == test.credit$Cost\_Matrix)

## [1] 0.76

# Bootstrap with 25 iterations  
mean(pred4 == test.credit$Cost\_Matrix)

## [1] 0.76