Data Mining Code Instructions Document

Project Title: Data Driven Travel Insurance

Section: 501 (9AM – 12 Noon)

Team 1 (Aakash Desai, Jinali Gandhi, Srishti Kabtiyal, Kanchan Markandeya)

Instructions to use this document

A dynamic table of content has been added for ease of navigation through this document. You can press Cntr+Click on a content tile that you wish to visit.

Table of Contents

[1. Reading Datafile on R 3](#_Toc482305747)

[2. Creating buckets for ‘Arrival Delay’. 4](#_Toc482305748)

[3. ‘Cleaning’ the Dataset 5](#_Toc482305749)

[3.1 Setting Values as NULL 5](#_Toc482305750)

[3.2 Converting some parameters to factors. 6](#_Toc482305751)

[3.3 Converting variables to obtain the hour of the day 6](#_Toc482305752)

[3.4 Converting variables to integer format 6](#_Toc482305753)

[3.5 Removing records with NA values 7](#_Toc482305754)

[4. Creating training and validation dataset 7](#_Toc482305755)

[5. Run linear regression on dataset 7](#_Toc482305756)

[6. Calculating AE, RMSE, MAE, SSE and MSE 8](#_Toc482305757)

[7. Forward selection 9](#_Toc482305758)

[7.1 Displaying results for forward selection 11](#_Toc482305759)

[7.2 BIC Graph for forward selection 12](#_Toc482305760)

[7.3 RSS Graph for forward selection 13](#_Toc482305761)

[7.4 Adjusted R-square graph for forward selection 14](#_Toc482305762)

[7.5 Cp graph for forward selection 15](#_Toc482305763)

[7.6 R-square graph for forward selection 16](#_Toc482305764)

[7.7 Adjusted R-square graph for forward selection 17](#_Toc482305765)

[7.8 BIC Graph for forward selection 18](#_Toc482305766)

[8. Backward Selection 18](#_Toc482305767)

[8.1 BIC Graph for backward selection 21](#_Toc482305768)

[8.2 RSS Graph for backward selection 22](#_Toc482305769)

[8.3 Adjusted R-square graph for backward selection 22](#_Toc482305770)

[8.4 Cp graph for backward selection 23](#_Toc482305771)

[8.5 R-square graph for backward selection 24](#_Toc482305772)

[8.6 Adjusted R-square graph for backward selection 25](#_Toc482305773)

[8.7 BIC Graph for backward selection 25](#_Toc482305774)

[9. Best Subset Model 26](#_Toc482305775)

[9.1 Calculating Errors 28](#_Toc482305776)

[9.2 Calculating AE, RMSE, MAE, SSE, MSE 29](#_Toc482305777)

[10. Feature Selection using Lasso 30](#_Toc482305778)

[10.1 Mean-Squared Error graph for Lasso 31](#_Toc482305779)

[10.2 Lasso Model Predict 32](#_Toc482305780)

[11. Ridge regression 33](#_Toc482305781)

[11.2 Mean-Squared Error Chart 35](#_Toc482305782)

[11.3 Ridge Model 36](#_Toc482305783)

[12. Regression Tree Model 38](#_Toc482305784)

[12.1 Plotting Regression Tree 38](#_Toc482305785)

[12.2 Plot of tree size vs deviance 39](#_Toc482305786)

[12.3 Pruned Regression Tree 40](#_Toc482305787)

[12.4 Predicted vs actual delay for pruned tree 41](#_Toc482305788)

[13. Classification 42](#_Toc482305789)

[13.1 Data cleaning for classification 42](#_Toc482305790)

[13.2 Creating training and validation Dataset 44](#_Toc482305791)

[13.3 Sampling for classification 45](#_Toc482305792)

[13.4 Data synthesis for 2 hours delay 45](#_Toc482305793)

[13.4 Classification tree with oversampling for 2 hours delay 46](#_Toc482305794)

[13.5 Data synthesis for 4 hours delay 46](#_Toc482305795)

[14. Association Rules 48](#_Toc482305796)

[14.1 Data cleaning for association rules 48](#_Toc482305797)

[14.2 Running code 50](#_Toc482305798)

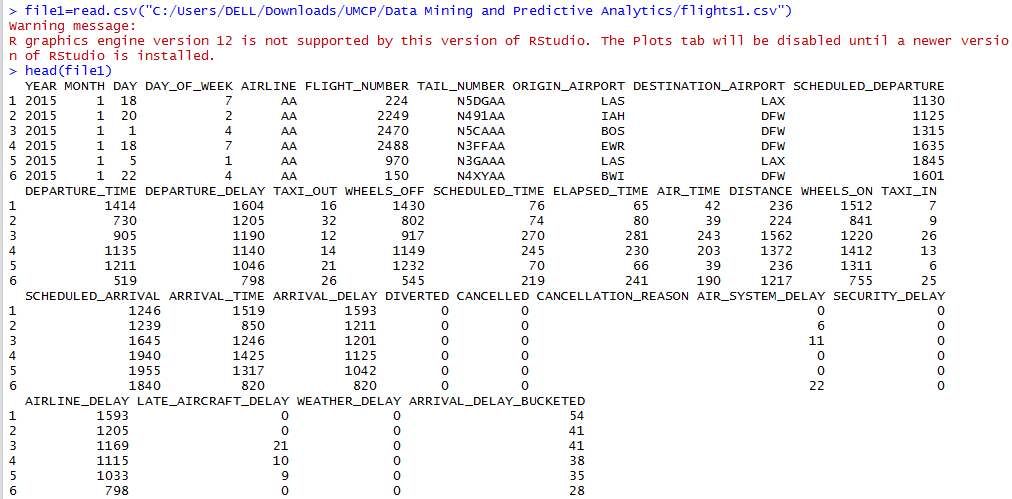
[14.3 Plot of lift vs support with confidence shading 50](#_Toc482305799)

## Reading Datafile on R

The first step towards executing any command or creating a model on a dataset is to add the data file on R. Following is the command to accomplish the same:

file1=read.csv("C:/Users/DELL/Downloads/UMCP/Data Mining and Predictive Analytics/flights1.csv")

head(file1)



## Creating buckets for ‘Arrival Delay’.

This was done to reduce the number of values for the arrival delay variable on the dataset. The exact timings of delay didn’t give any valuable insights into the model, which is why we categorized the arrival delay timings into buckets of 15 minutes durations.

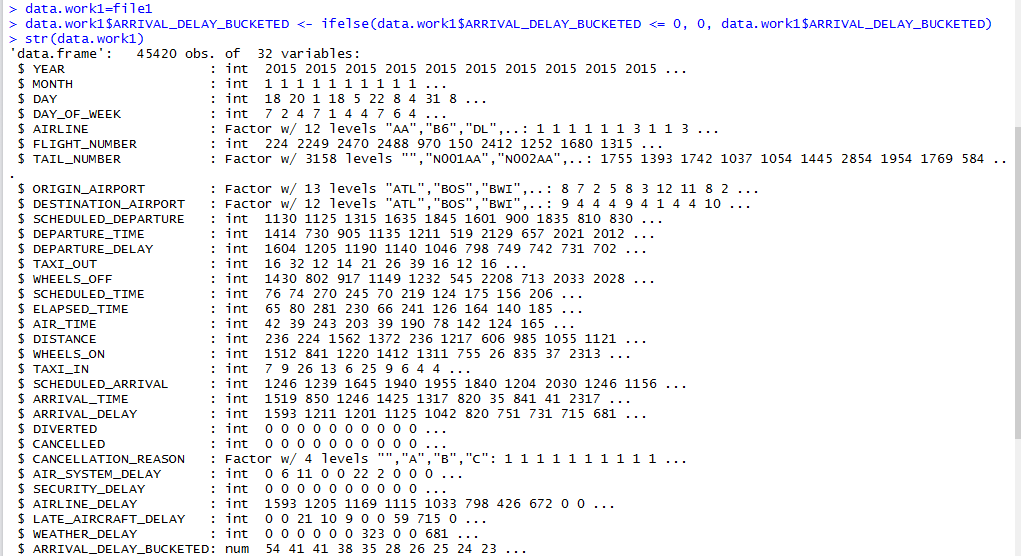
data.work1=file1

#data.work1$ARRIVAL\_DELAY\_BUCKETED <- data.work1$ARRIVAL\_DELAY/15

#data.work1$ARRIVAL\_DELAY\_BUCKETED <- as.integer(data.work1$ARRIVAL\_DELAY\_BUCKETED)

data.work1$ARRIVAL\_DELAY\_BUCKETED <- ifelse(data.work1$ARRIVAL\_DELAY\_BUCKETED <= 0, 0, data.work1$ARRIVAL\_DELAY\_BUCKETED)

str(data.work1)



## ‘Cleaning’ the Dataset

### 3.1 Setting Values as NULL

Some of the attributes have to be set NULL. This was so because these columns these columns had very few values entries in dataset. Following are the commands to set the values as NULL.

data.work1$DIVERTED=NULL

data.work1$CANCELLED=NULL

data.work1$CANCELLATION\_REASON=NULL

data.work1$YEAR=NULL

data.work1$DAY=NULL

data.work1$MONTH=NULL

data.work1$FLIGHT\_NUMBER=NULL

data.work1$TAIL\_NUMBER=NULL

data.work1$DEPARTURE\_TIME=NULL

data.work1$WHEELS\_OFF=NULL

data.work1$WHEELS\_ON=NULL

data.work1$ARRIVAL\_TIME=NULL

data.work1$AIR\_SYSTEM\_DELAY=NULL

data.work1$SECURITY\_DELAY=NULL

data.work1$AIRLINE\_DELAY=NULL

data.work1$LATE\_AIRCRAFT\_DELAY=NULL

data.work1$WEATHER\_DELAY=NULL

data.work1$DEPARTURE\_DELAY=NULL

data.work1$ARRIVAL\_DELAY=NULL

### 3.2 Converting some parameters to factors.

data.work1$DAY\_OF\_WEEK <- as.factor(data.work1$DAY\_OF\_WEEK)

data.work1$AIRLINE <- as.factor(data.work1$AIRLINE)

### 3.3 Converting variables to obtain the hour of the day

The time variables were divided by 100 and then rounded off to find out the hour of the day. Following is the set of command lines to accomplish the same:

data.work1$SCHEDULED\_DEPARTURE=data.work1$SCHEDULED\_DEPARTURE/100

data.work1$SCHEDULED\_DEPARTURE=round(data.work1$SCHEDULED\_DEPARTURE)

data.work1$SCHEDULED\_ARRIVAL=data.work1$SCHEDULED\_ARRIVAL/100

data.work1$SCHEDULED\_ARRIVAL=round(data.work1$SCHEDULED\_ARRIVAL)

### 3.4 Converting variables to integer format

data.work1$SCHEDULED\_DEPARTURE <- as.integer(data.work1$SCHEDULED\_DEPARTURE)

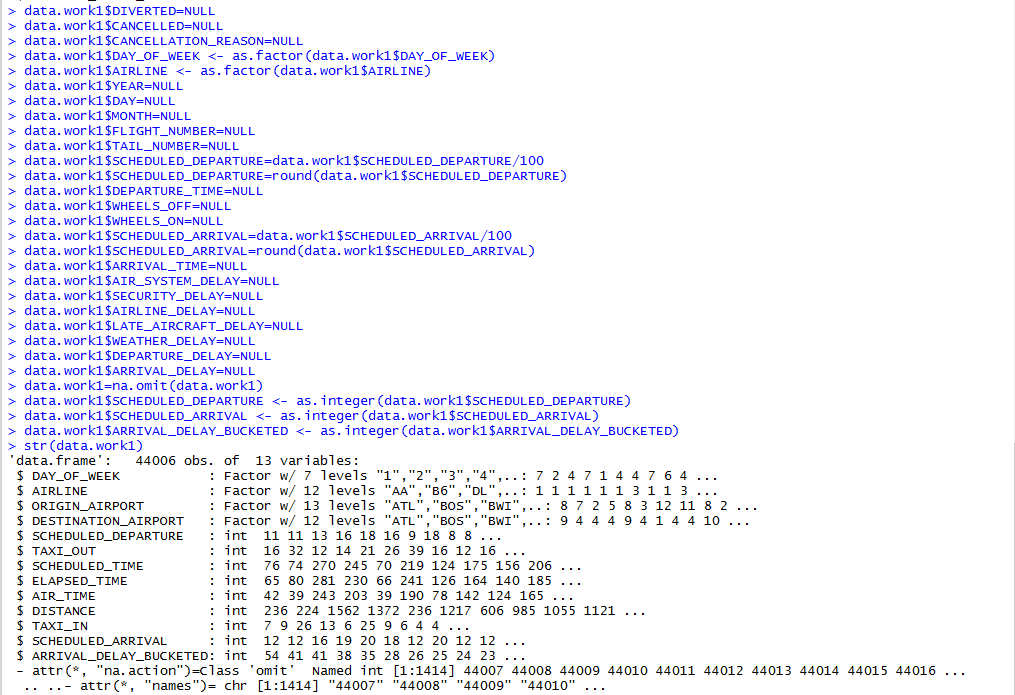
data.work1$SCHEDULED\_ARRIVAL <- as.integer(data.work1$SCHEDULED\_ARRIVAL)

data.work1$ARRIVAL\_DELAY\_BUCKETED <- as.integer(data.work1$ARRIVAL\_DELAY\_BUCKETED)

### 3.5 Removing records with NA values

data.work1=na.omit(data.work1)

str(data.work1)



## Creating training and validation dataset

set.seed(12356)

train=sample(nrow(data.work1),0.7\*nrow(data.work1))

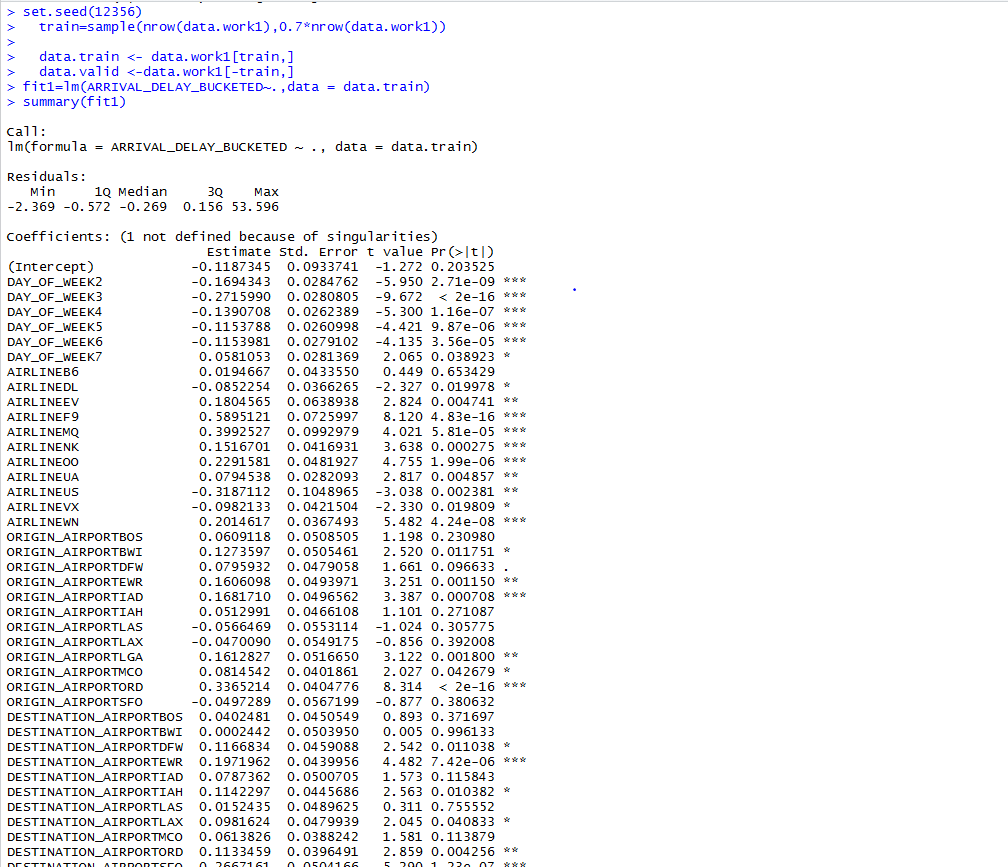
data.train <- data.work1[train,]

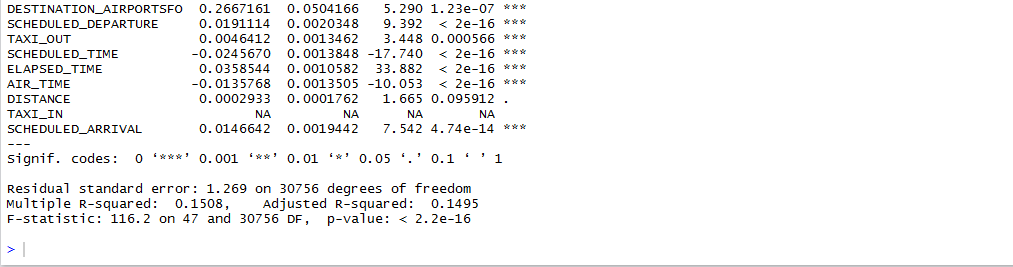
data.valid <-data.work1[-train,]

## Run linear regression on dataset

fit1=lm(ARRIVAL\_DELAY\_BUCKETED~.,data = data.train)

summary(fit1)





## Calculating AE, RMSE, MAE, SSE and MSE

Metrics <- c("AE","RMSE","MAE","SSE","MSE")

x1 <- mean(actual.linear - pred.linear)

x2 <- sqrt(mean((actual.linear - pred.linear)^2))

x3 <- mean(abs(actual.linear - pred.linear))

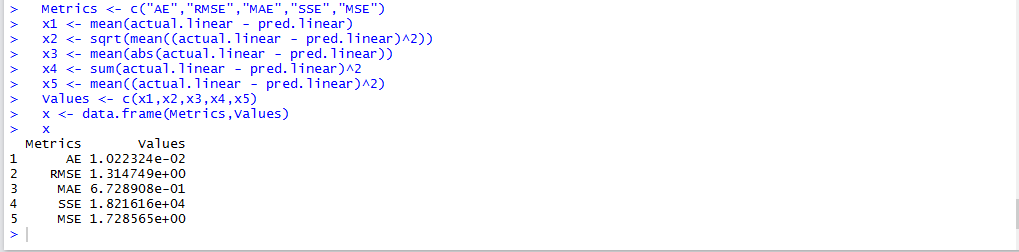
x4 <- sum(actual.linear - pred.linear)^2

x5 <- mean((actual.linear - pred.linear)^2)

Values <- c(x1,x2,x3,x4,x5)

x <- data.frame(Metrics,Values)

x



As shown in the output above, the MSE obtained for the linear model is 1.72

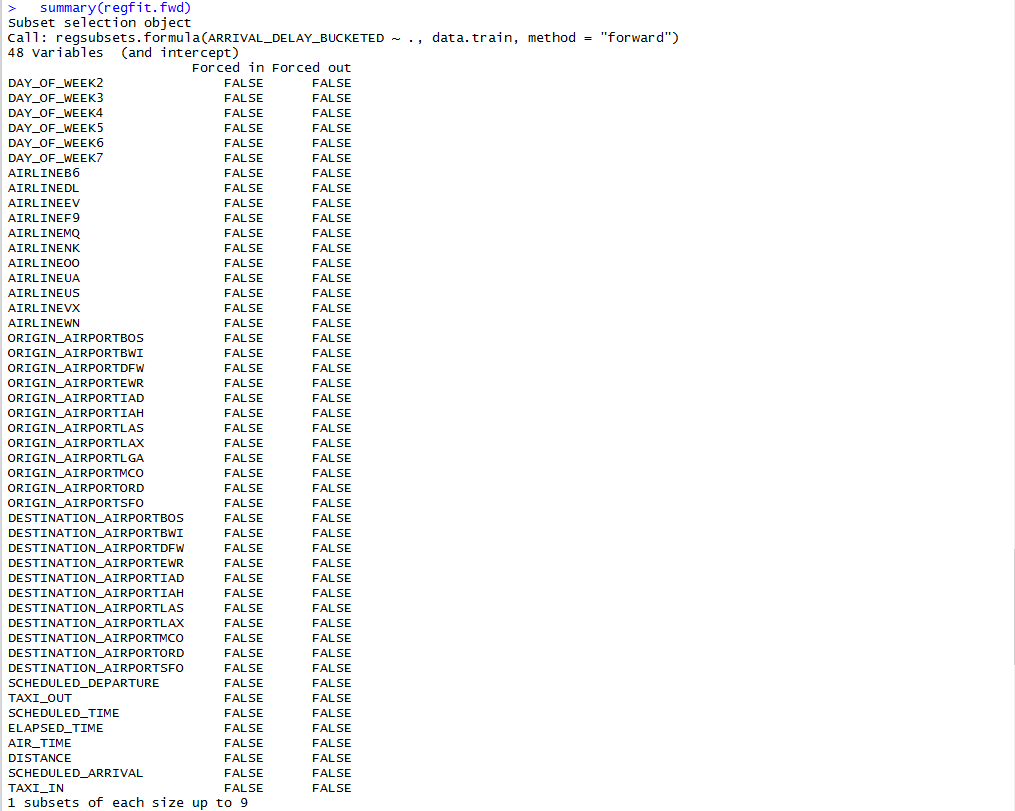
## Forward selection

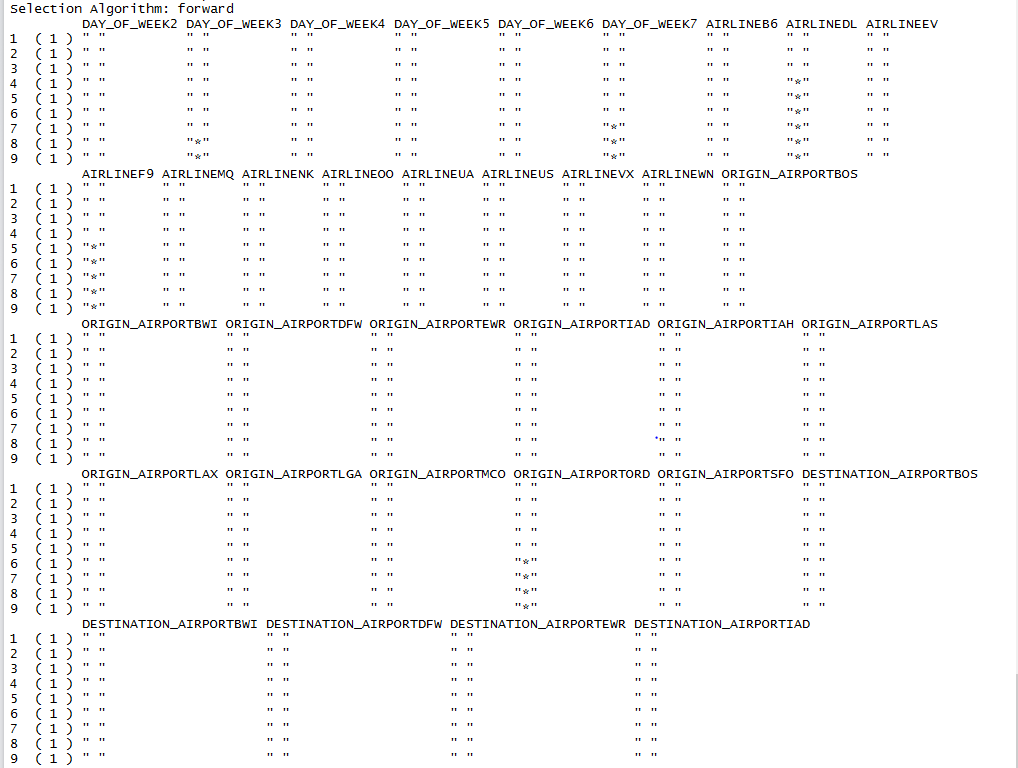
library(leaps)

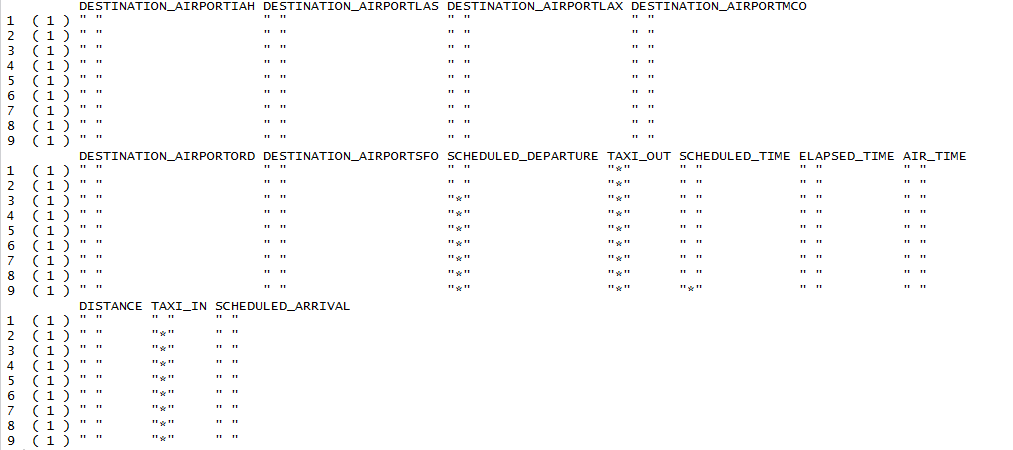
regfit.fwd=regsubsets(ARRIVAL\_DELAY\_BUCKETED~., data.train, method="forward")

summary(regfit.fwd)









### 

### 7.1 Displaying results for forward selection

regfit.summaryfwd = summary(regfit.fwd)

names(regfit.summaryfwd)

regfit.summaryfwd$rsq

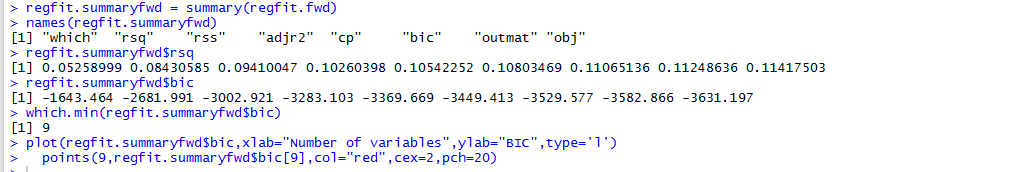
regfit.summaryfwd$bic

which.min(regfit.summaryfwd$bic)

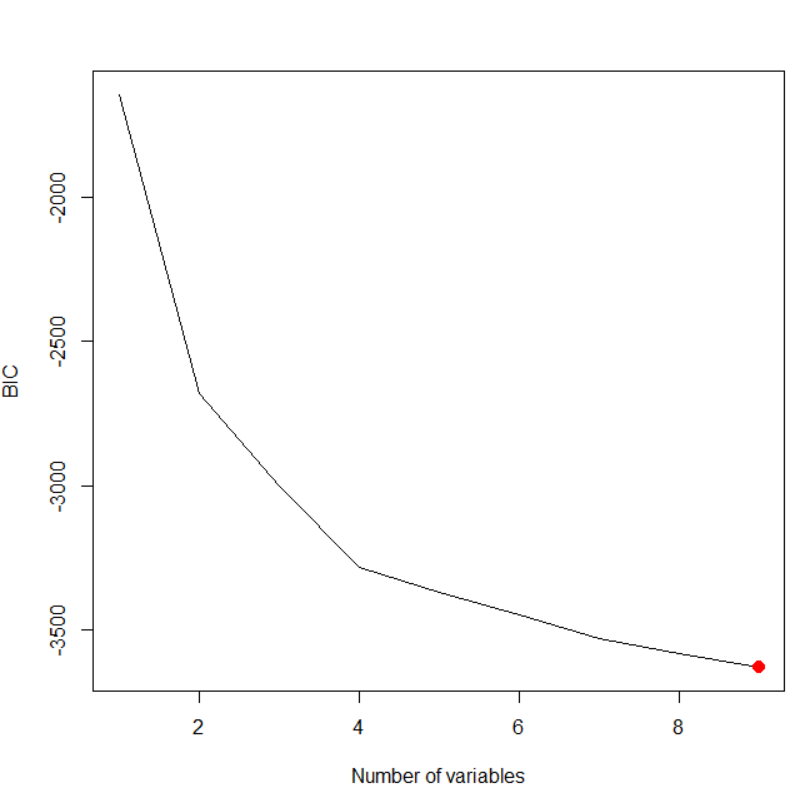
plot(regfit.summaryfwd$bic,xlab="Number of variables",ylab="BIC",type='l')

points(9,regfit.summaryfwd$bic[9],col="red",cex=2,pch=20)

plot(regfit.summaryfwd$rss,xlab="Number of variables",ylab="RSS",type='l')



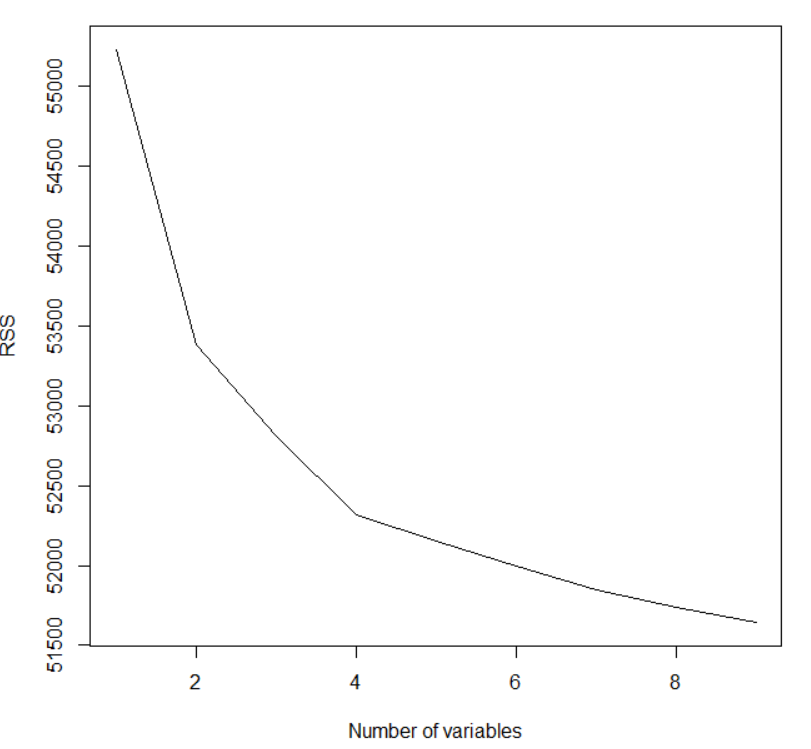
### 7.2 BIC Graph for forward selection



### 7.3 RSS Graph for forward selection

plot(regfit.summaryfwd$rss,xlab="Number of variables",ylab="RSS",type='l')





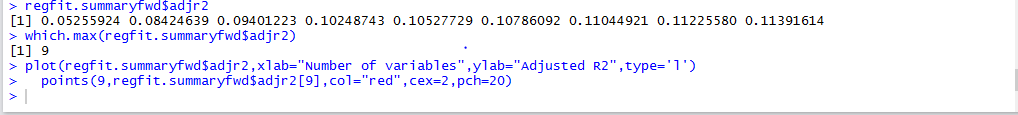
### 7.4 Adjusted R-square graph for forward selection

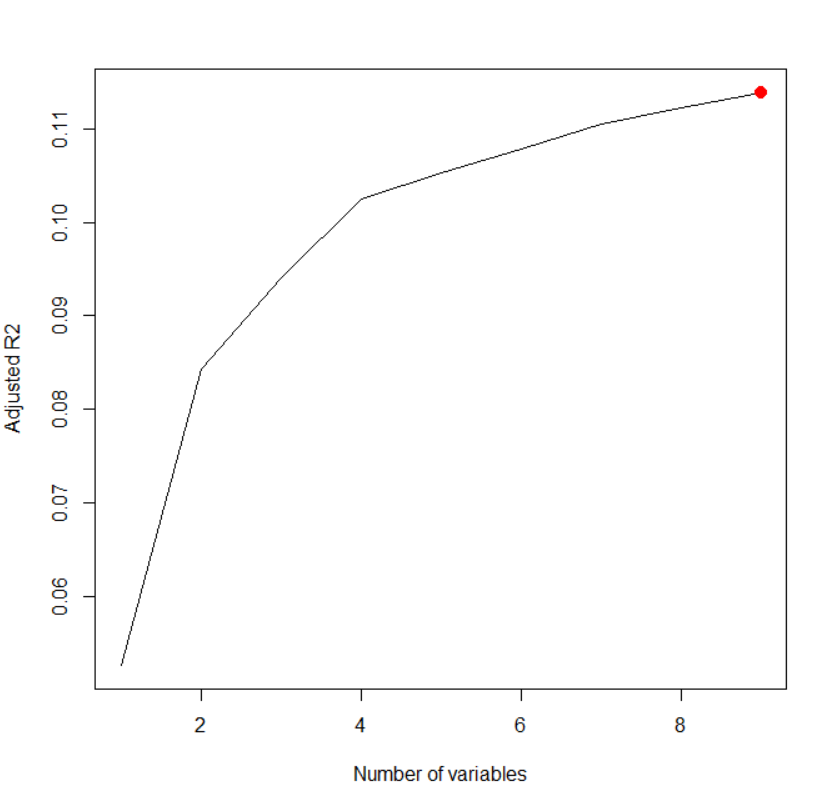
regfit.summaryfwd$adjr2

which.max(regfit.summaryfwd$adjr2)

plot(regfit.summaryfwd$adjr2,xlab="Number of variables",ylab="Adjusted R2",type='l')

points(9,regfit.summaryfwd$adjr2[9],col="red",cex=2,pch=20)





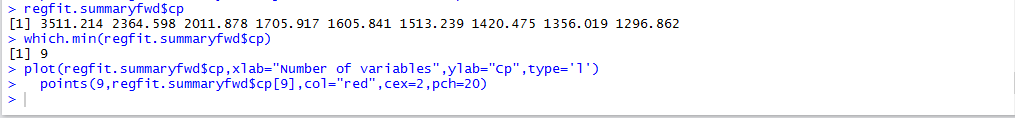
### 7.5 Cp graph for forward selection

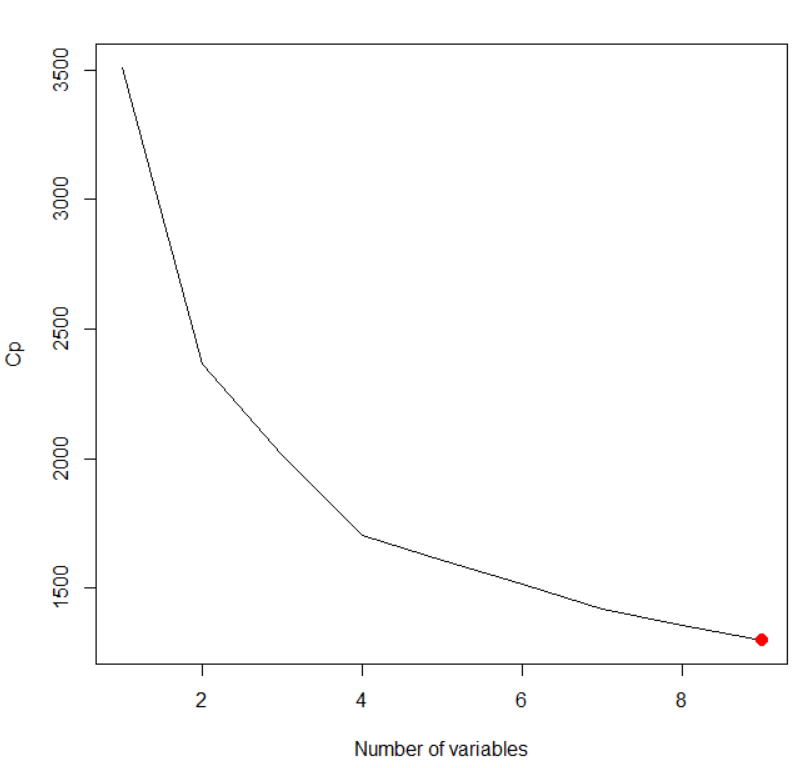
regfit.summaryfwd$cp

which.min(regfit.summaryfwd$cp)

plot(regfit.summaryfwd$cp,xlab="Number of variables",ylab="Cp",type='l')

points(9,regfit.summaryfwd$cp[9],col="red",cex=2,pch=20)

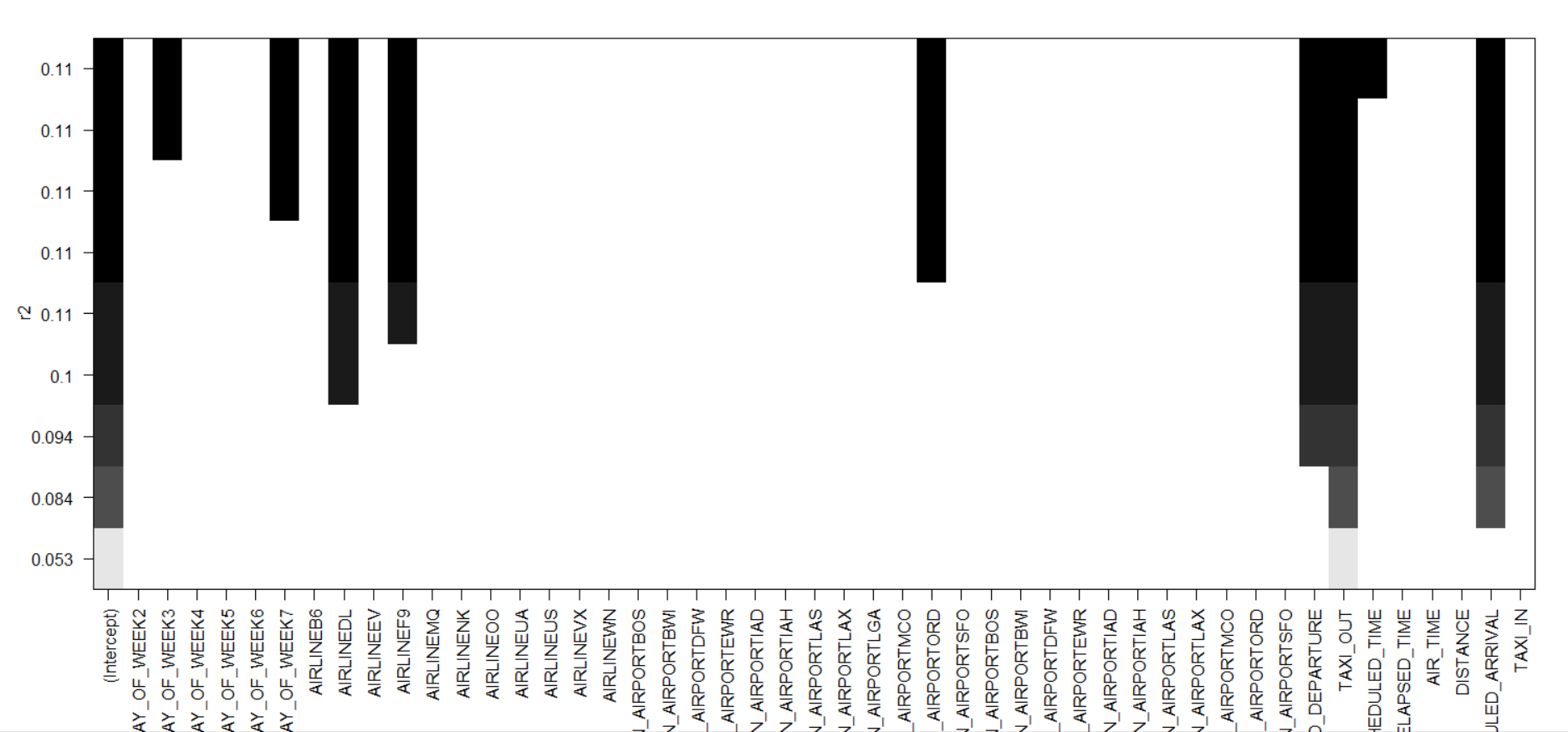




### 7.6 R-square graph for forward selection

plot(regfit.fwd,scale="r2")

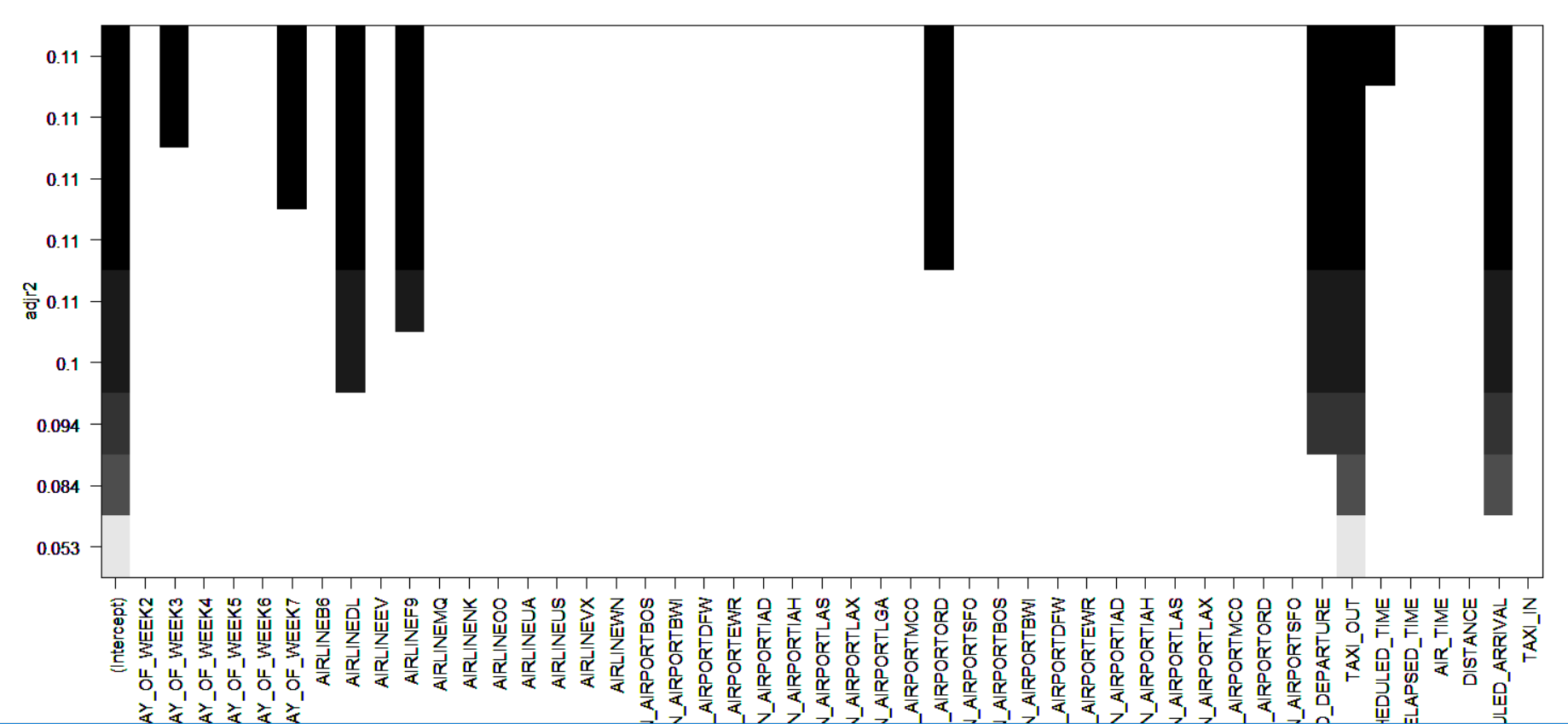




### 7.7 Adjusted R-square graph for forward selection

plot(regfit.fwd,scale="adjr2")

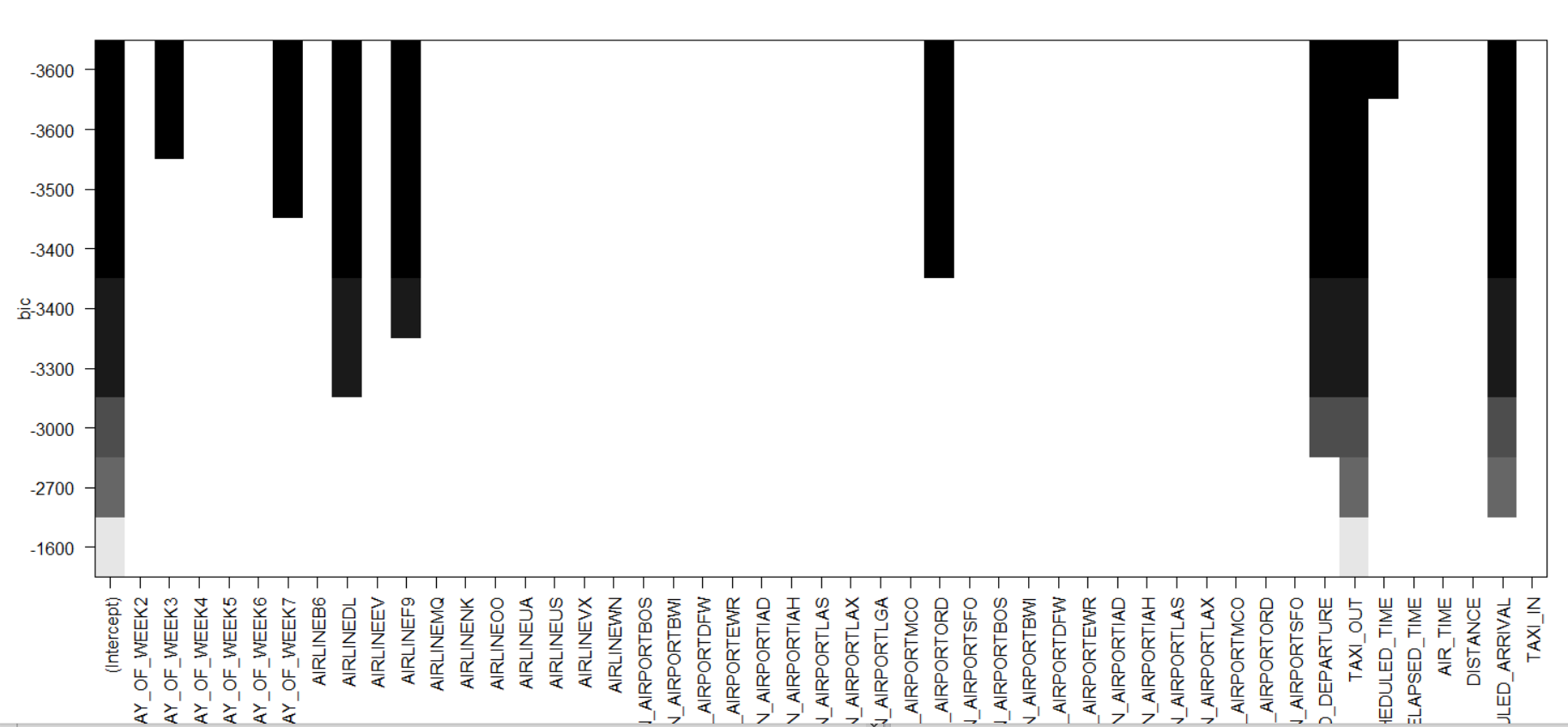




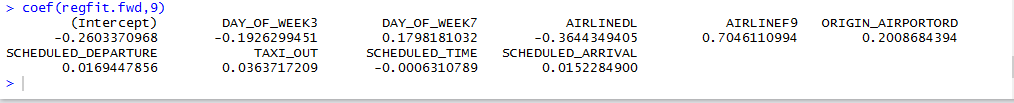
### 7.8 BIC Graph for forward selection

plot(regfit.fwd,scale="bic")





coef(regfit.fwd,9)

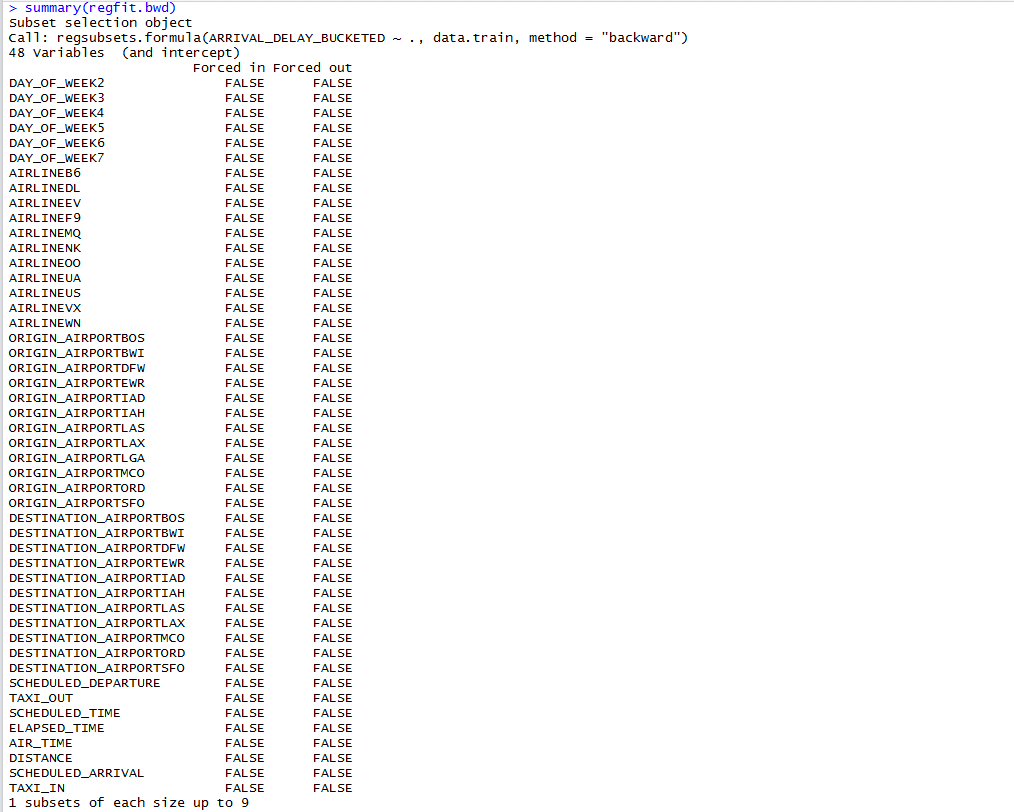


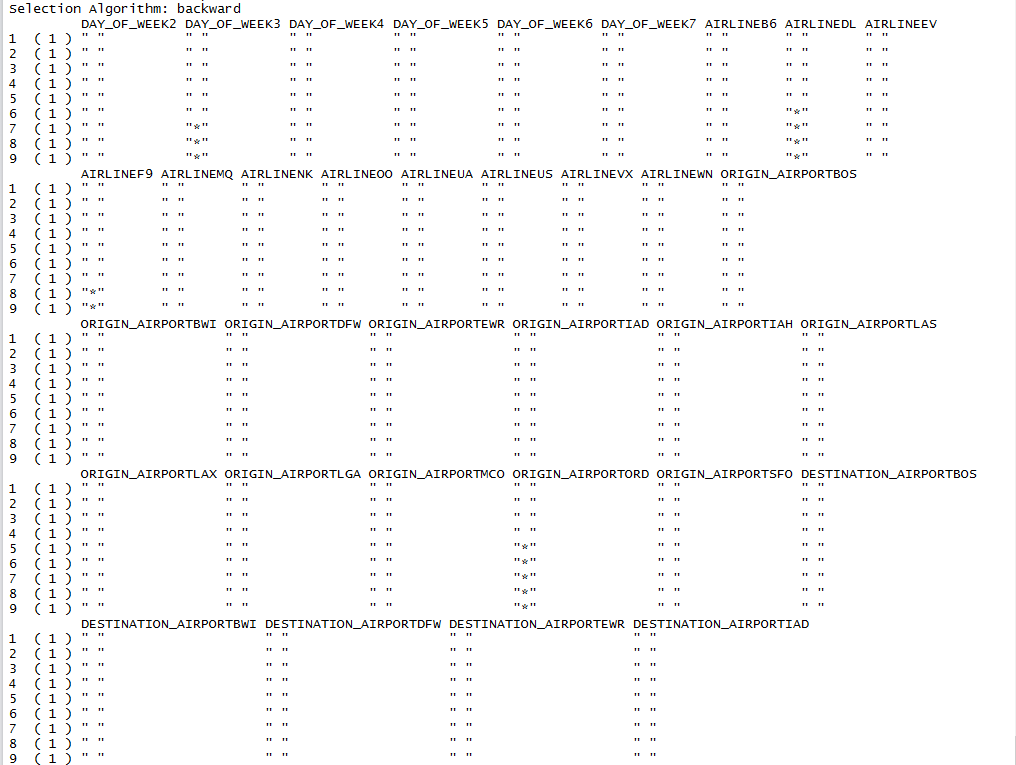
## Backward Selection

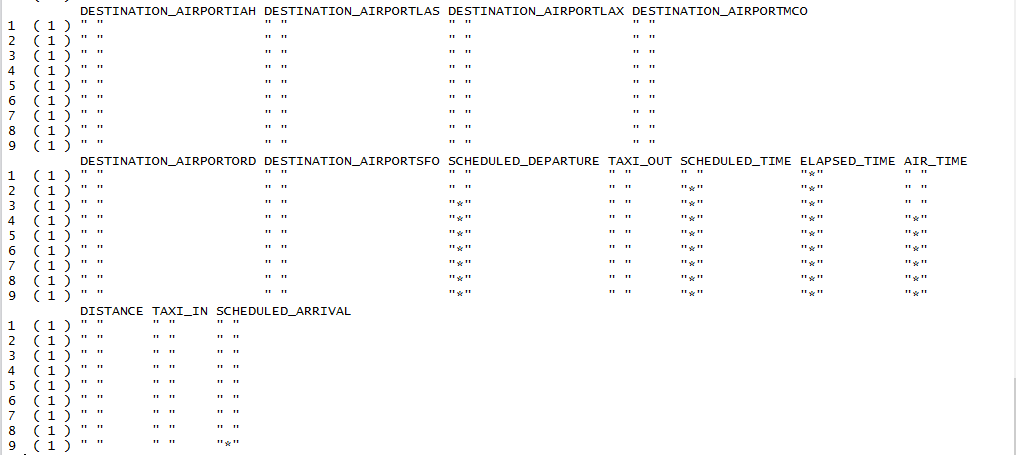
library(leaps)

regfit.bwd=regsubsets(ARRIVAL\_DELAY\_BUCKETED~., data.train, method="backward")









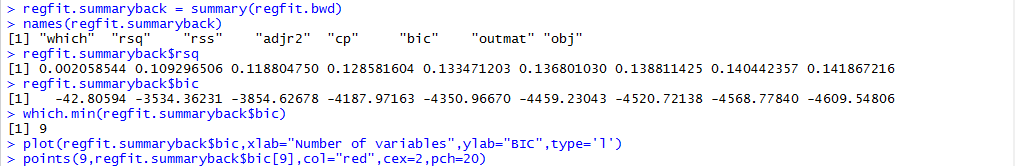
regfit.summaryback = summary(regfit.bwd)

names(regfit.summaryback)

regfit.summaryback$rsq

regfit.summaryback$bic

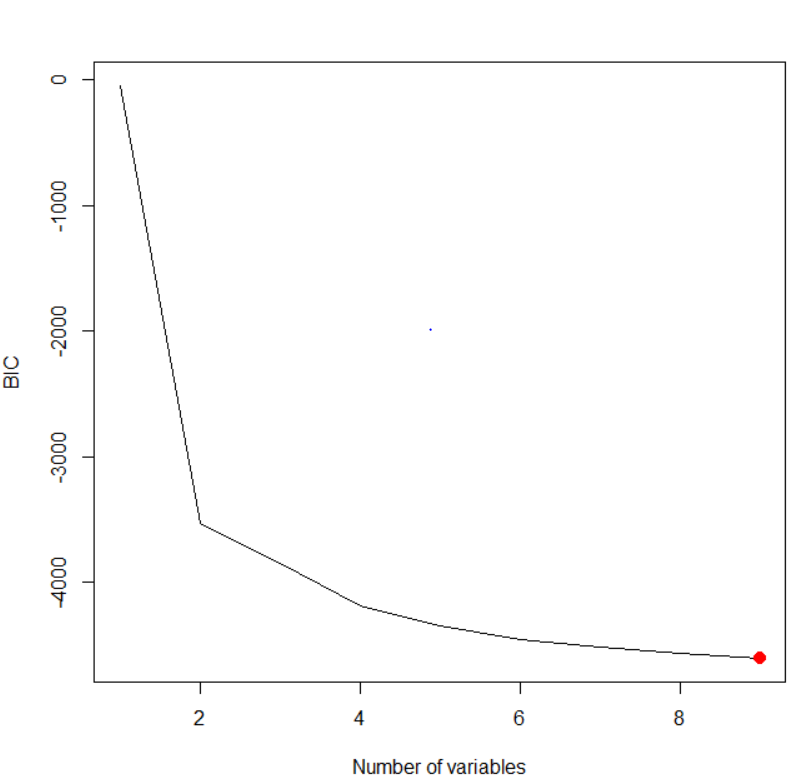
which.min(regfit.summaryback$bic)



### 8.1 BIC Graph for backward selection

plot(regfit.summaryback$bic,xlab="Number of variables",ylab="BIC",type='l')

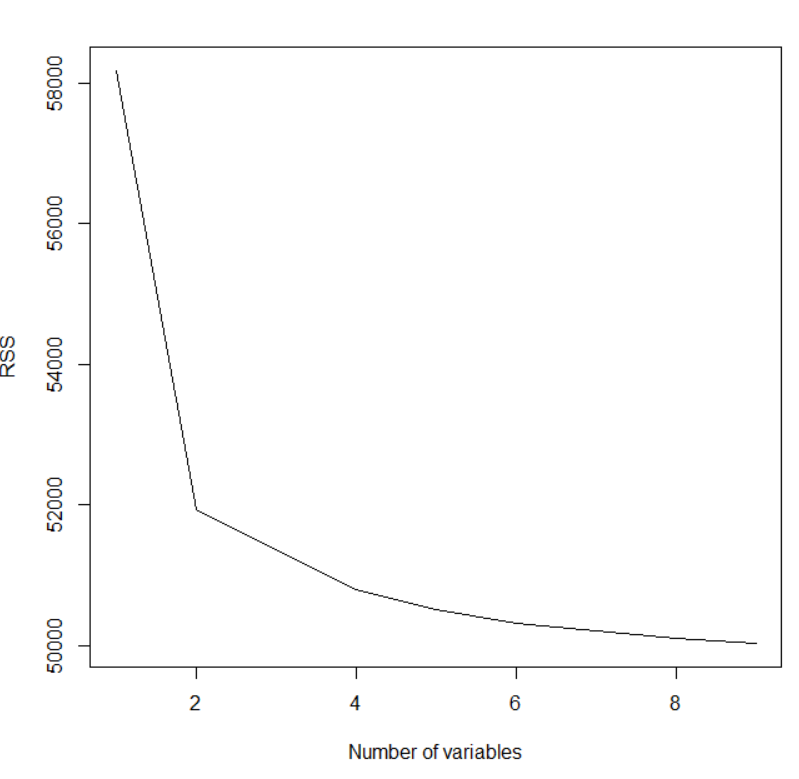
points(9,regfit.summaryback$bic[9],col="red",cex=2,pch=20)



### 8.2 RSS Graph for backward selection

plot(regfit.summaryback$rss,xlab="Number of variables",ylab="RSS",type='l')





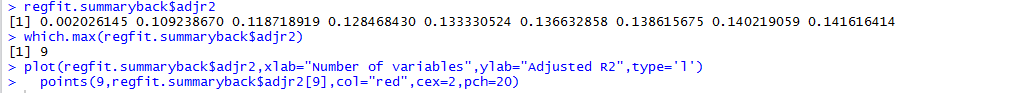
### 8.3 Adjusted R-square graph for backward selection

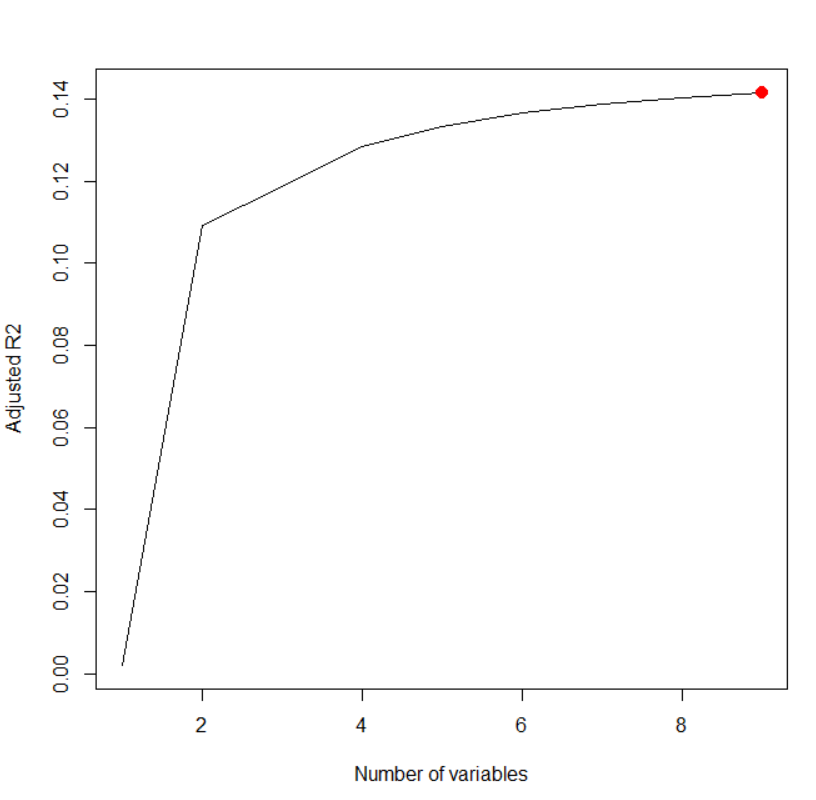
regfit.summaryback$adjr2

which.max(regfit.summaryback$adjr2)

plot(regfit.summaryback$adjr2,xlab="Number of variables",ylab="Adjusted R2",type='l')

points(9,regfit.summaryback$adjr2[9],col="red",cex=2,pch=20)





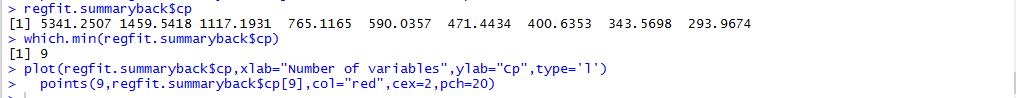
### 8.4 Cp graph for backward selection

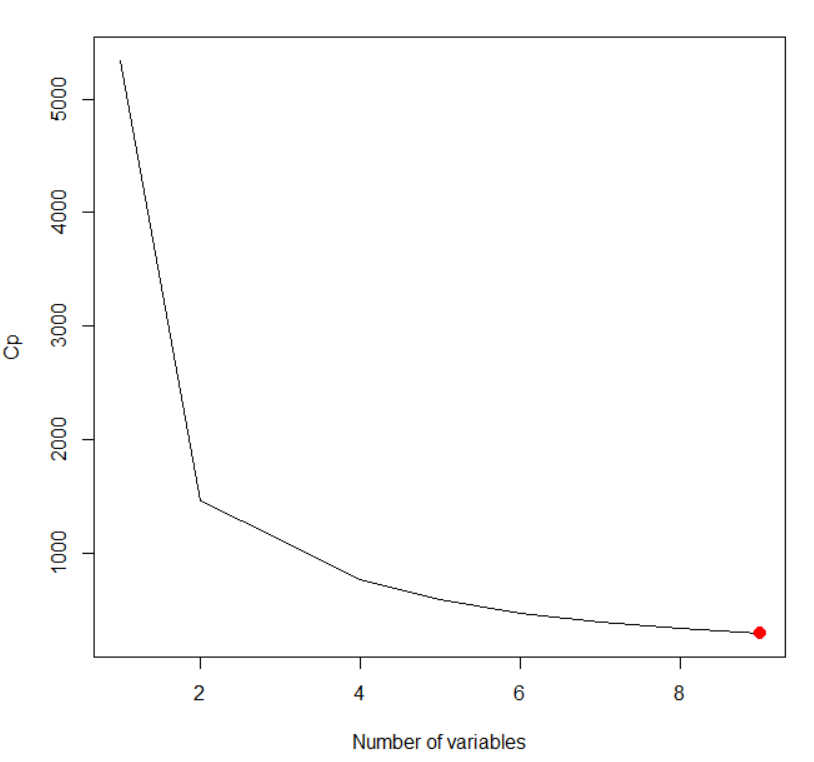
regfit.summaryback$cp

which.min(regfit.summaryback$cp)

plot(regfit.summaryback$cp,xlab="Number of variables",ylab="Cp",type='l')

points(9,regfit.summaryback$cp[9],col="red",cex=2,pch=20)

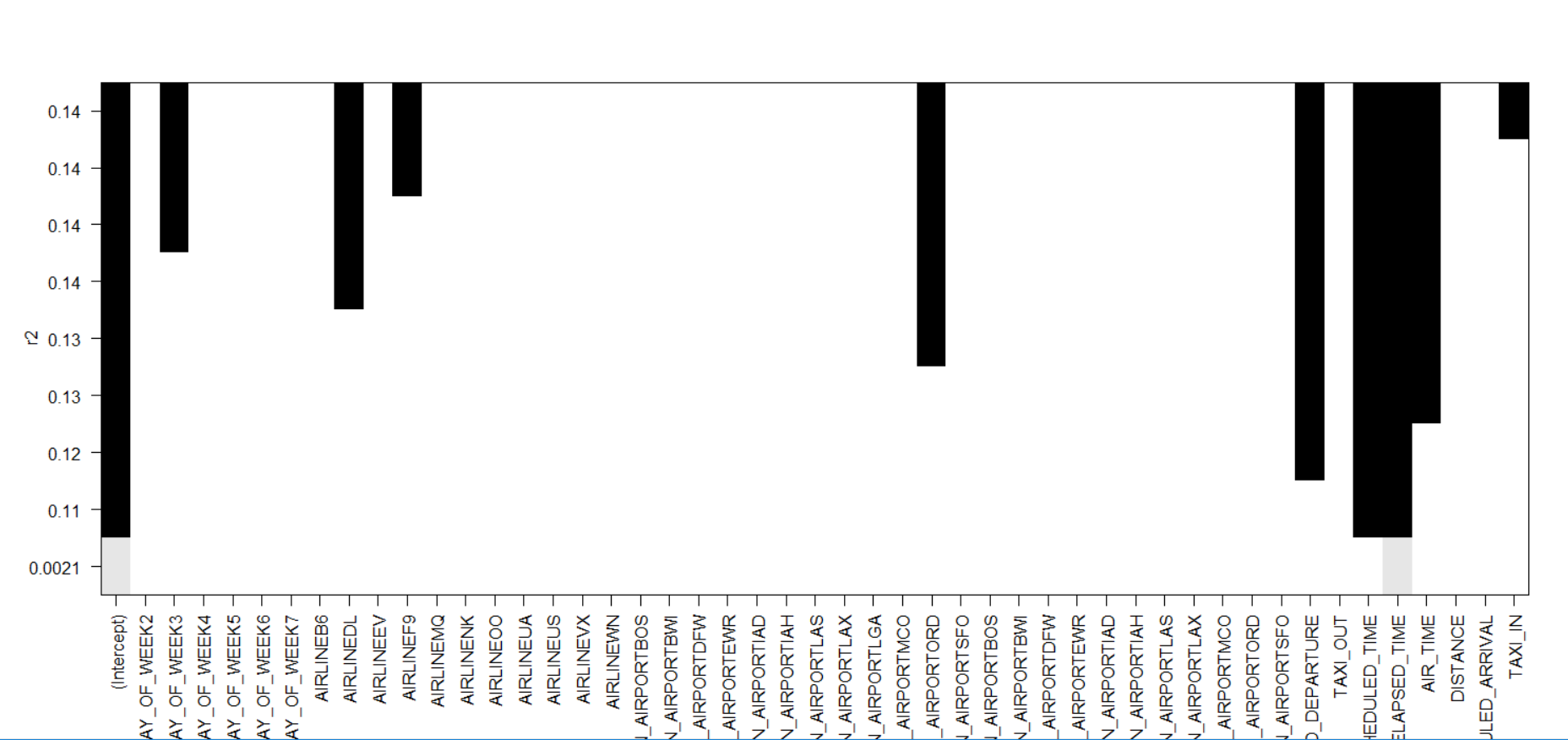




### 8.5 R-square graph for backward selection

plot(regfit.bwd,scale="r2")

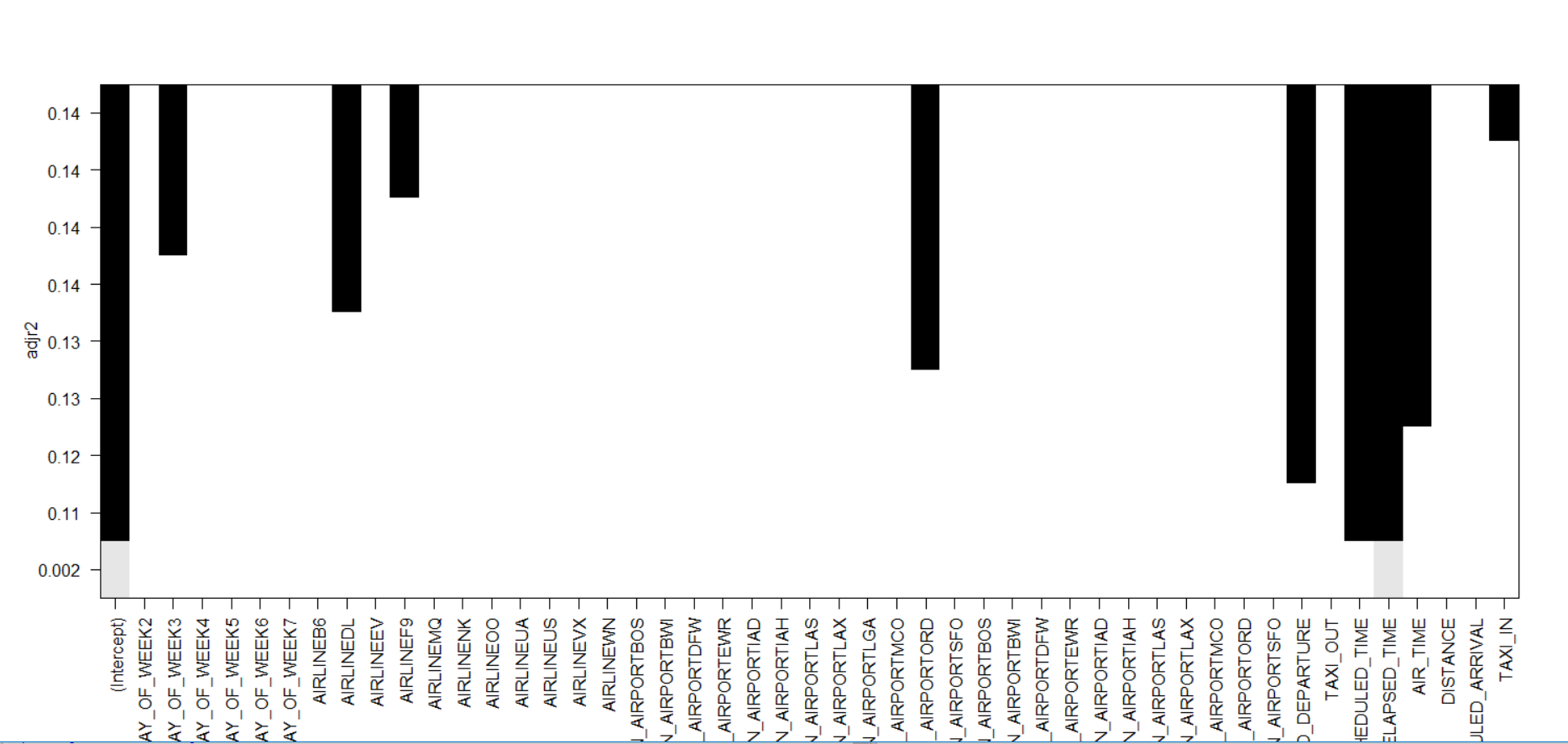




### 8.6 Adjusted R-square graph for backward selection

plot(regfit.bwd,scale="adjr2")

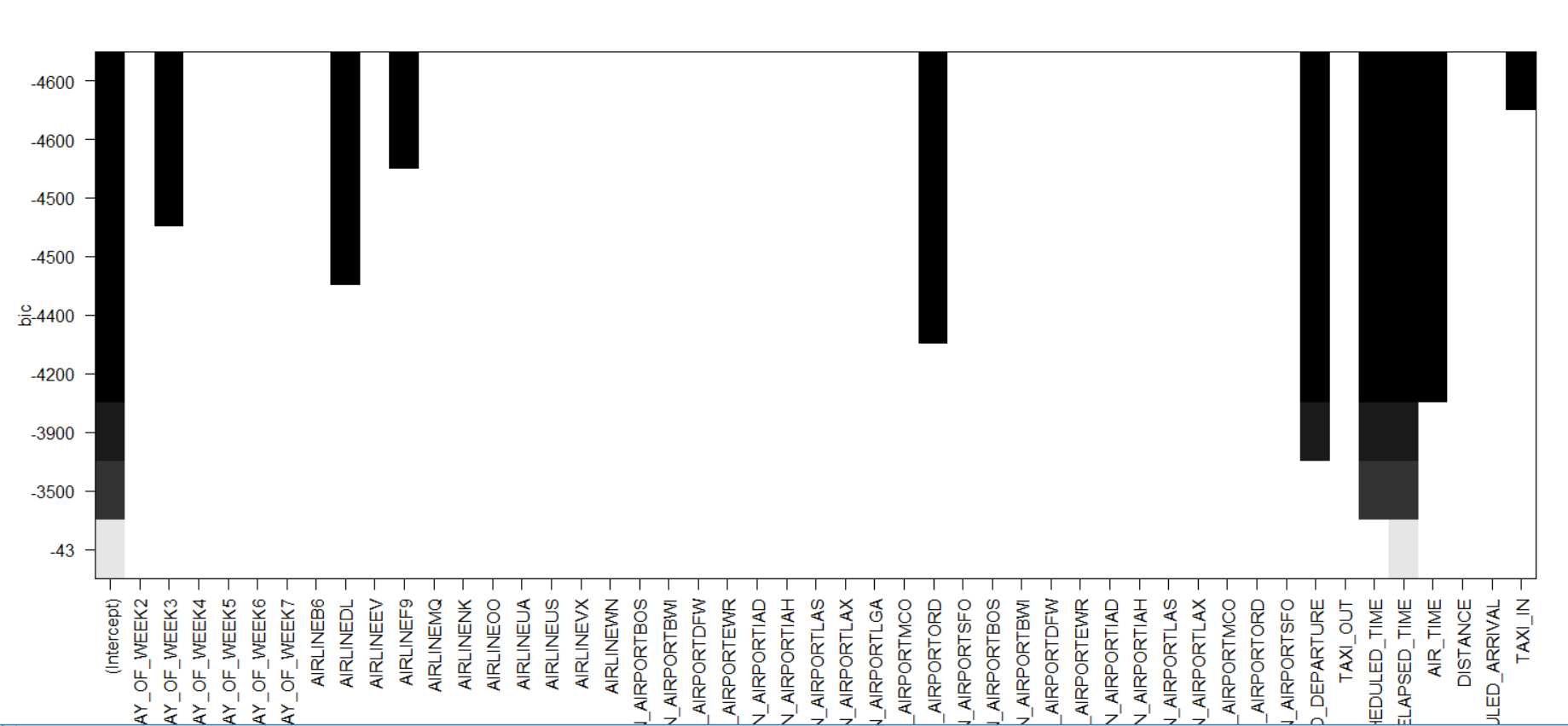




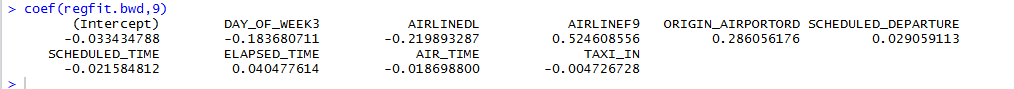
### 8.7 BIC Graph for backward selection

plot(regfit.bwd,scale="bic")





coef(regfit.bwd,9)



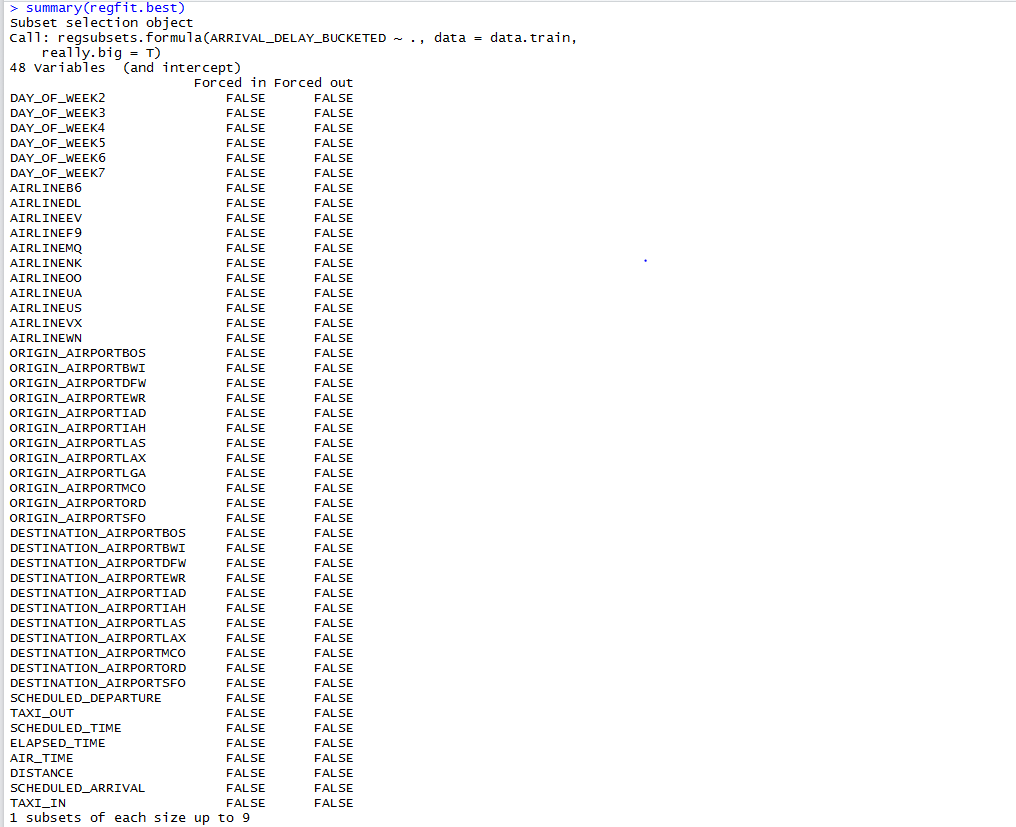
## Best Subset Model

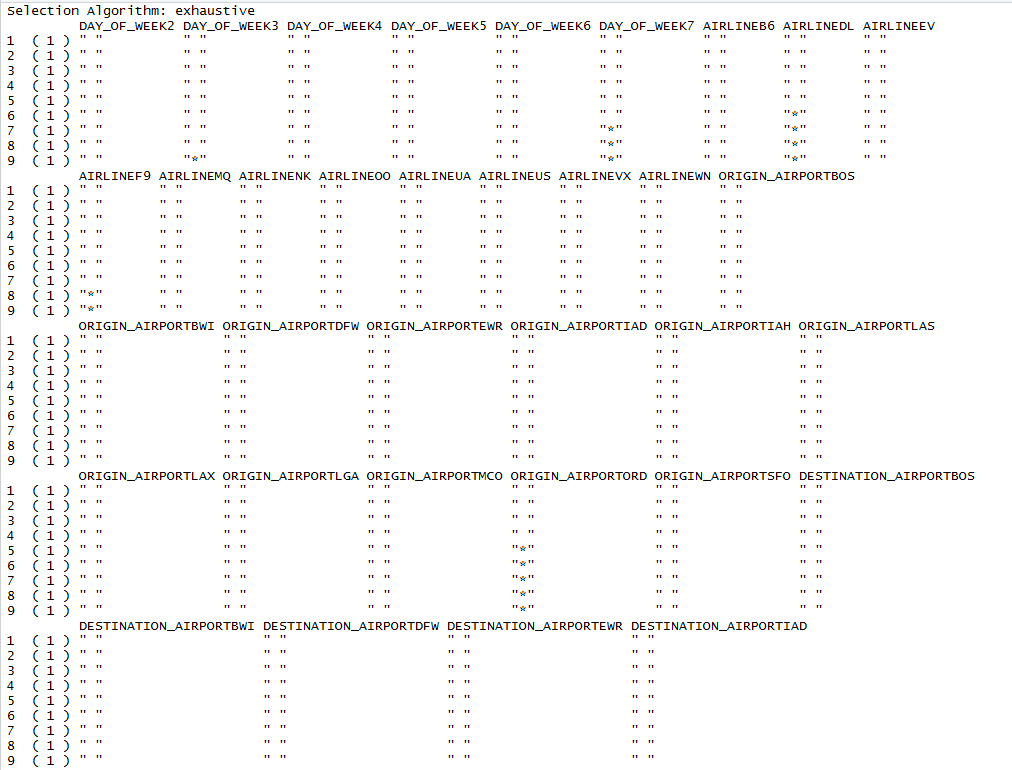
set.seed(12356)

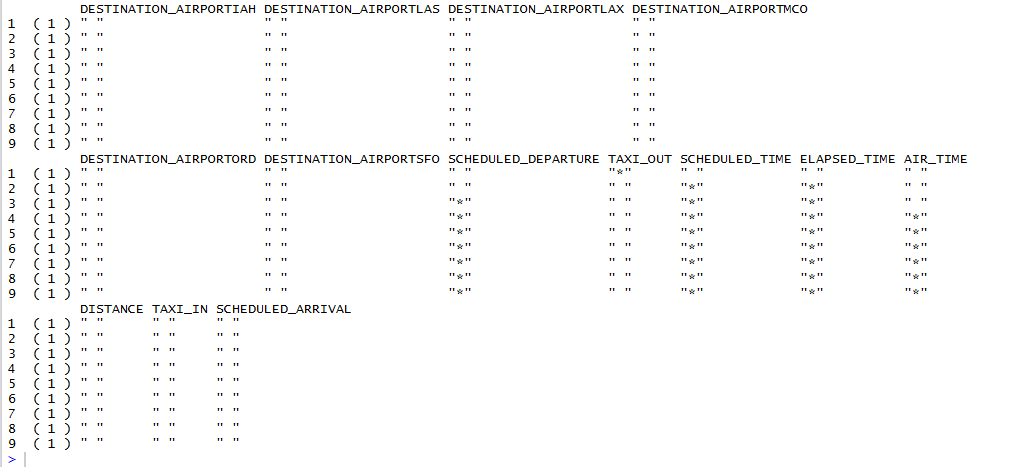
regfit.best=regsubsets(ARRIVAL\_DELAY\_BUCKETED~.,data=data.train,really.big = T)

summary(regfit.best)









## 

### 9.1 Calculating Errors

test.mat=model.matrix(ARRIVAL\_DELAY\_BUCKETED~.,data=data.valid)

val.errors=rep(NA,14)

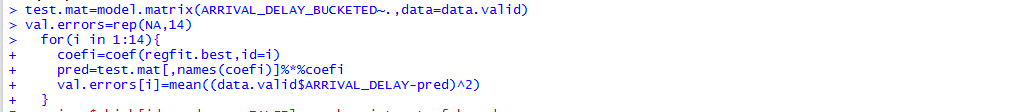
for(i in 1:14){

coefi=coef(regfit.best,id=i)

pred=test.mat[,names(coefi)]%\*%coefi

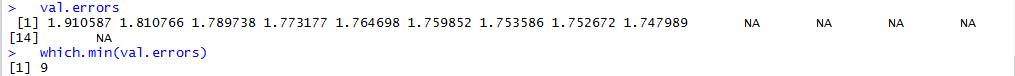
val.errors[i]=mean((data.valid$ARRIVAL\_DELAY-pred)^2)

}

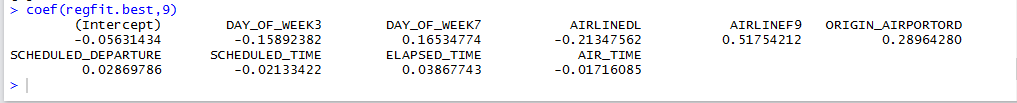


val.errors

which.min(val.errors)



coef(regfit.best,9)



### 9.2 Calculating AE, RMSE, MAE, SSE, MSE

bestsubsetmodel <- lm(ARRIVAL\_DELAY\_BUCKETED~DAY\_OF\_WEEK+AIRLINE+ORIGIN\_AIRPORT+SCHEDULED\_DEPARTURE+SCHEDULED\_TIME+ELAPSED\_TIME+AIR\_TIME, data=data.train)

predict\_test <- predict(bestsubsetmodel, newdata = data.valid)

actual <- data.valid$ARRIVAL\_DELAY

Metrics <- c("AE","RMSE","MAE","SSE","MSE")

x1 <- mean(actual - predict\_test)

x2 <- sqrt(mean((actual - predict\_test)^2))

x3 <- mean(abs(actual - predict\_test))

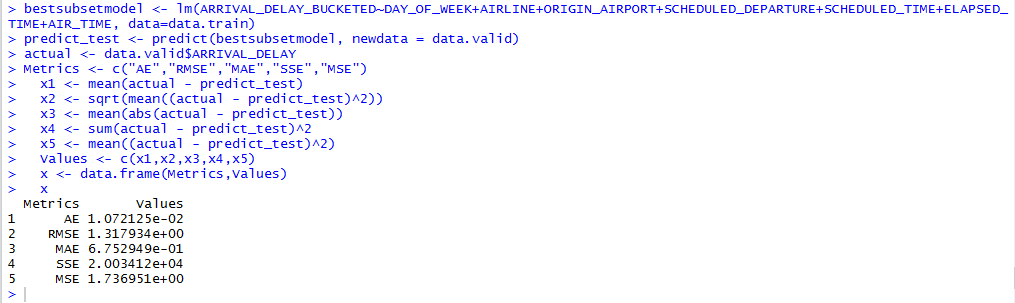
x4 <- sum(actual - predict\_test)^2

x5 <- mean((actual - predict\_test)^2)

Values <- c(x1,x2,x3,x4,x5)

x <- data.frame(Metrics,Values)

x



## Feature Selection using Lasso

#Lasso regression

set.seed(12356)

x = model.matrix(ARRIVAL\_DELAY\_BUCKETED~.,data.train)[,-1]

y = data.train$ARRIVAL\_DELAY\_BUCKETED

#install.packages("glmnet")

library(glmnet)

grid = 10^seq(10,-2,length=100)

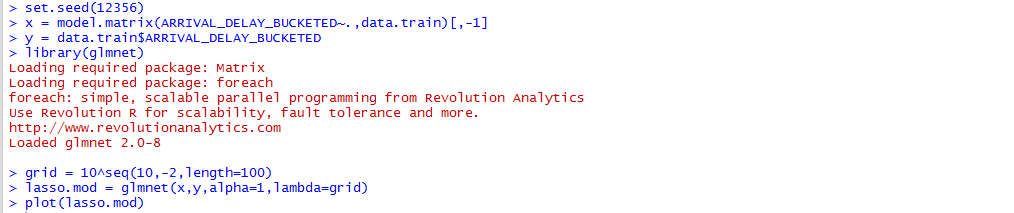
lasso.mod = glmnet(x,y,alpha=1,lambda=grid)

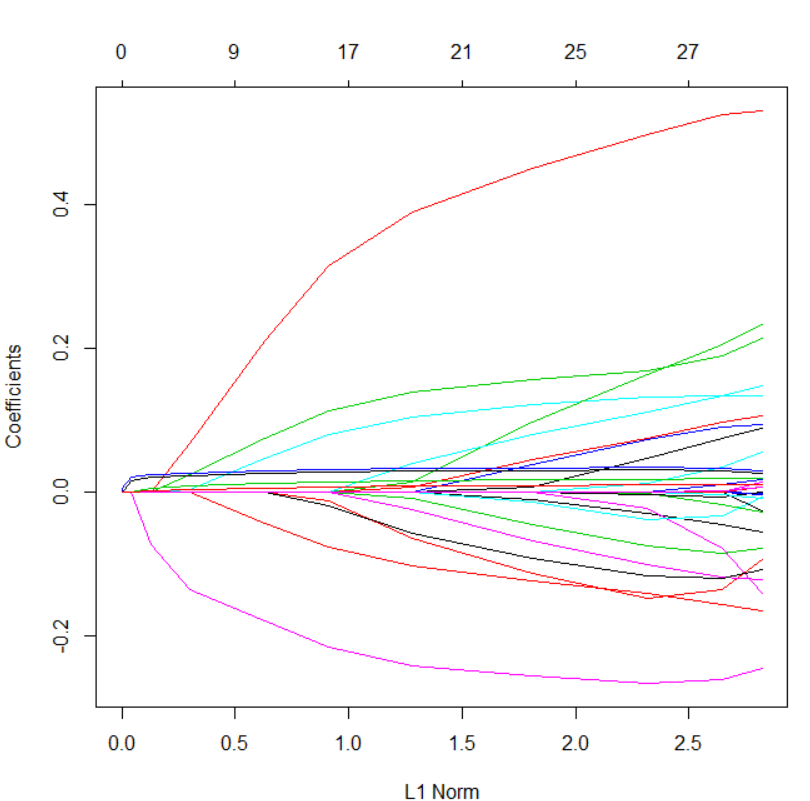
plot(lasso.mod)

grid = 10^seq(10,-2,length=100)

lasso.mod = glmnet(x,y,alpha=1,lambda=grid)

plot(lasso.mod)



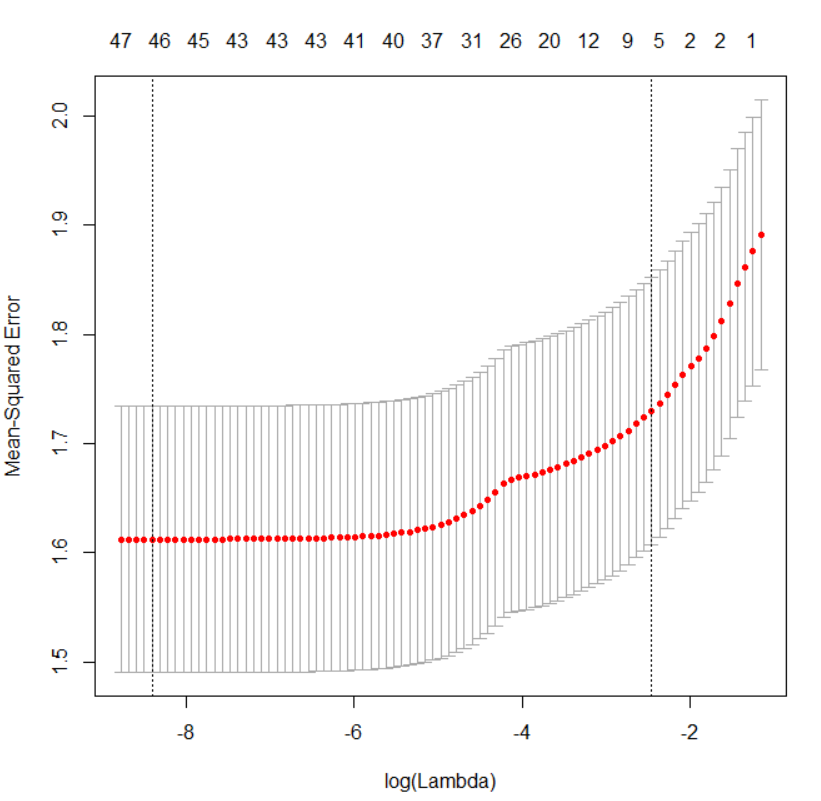


cv.out = cv.glmnet(x,y,alpha=1)

plot(cv.out)

### 10.1 Mean-Squared Error graph for Lasso





### 10.2 Lasso Model Predict

bestlam=cv.out$lambda.min

bestlam

lasso.pred = predict(lasso.mod,s=bestlam,newx=x[1:nrow(data.valid),])

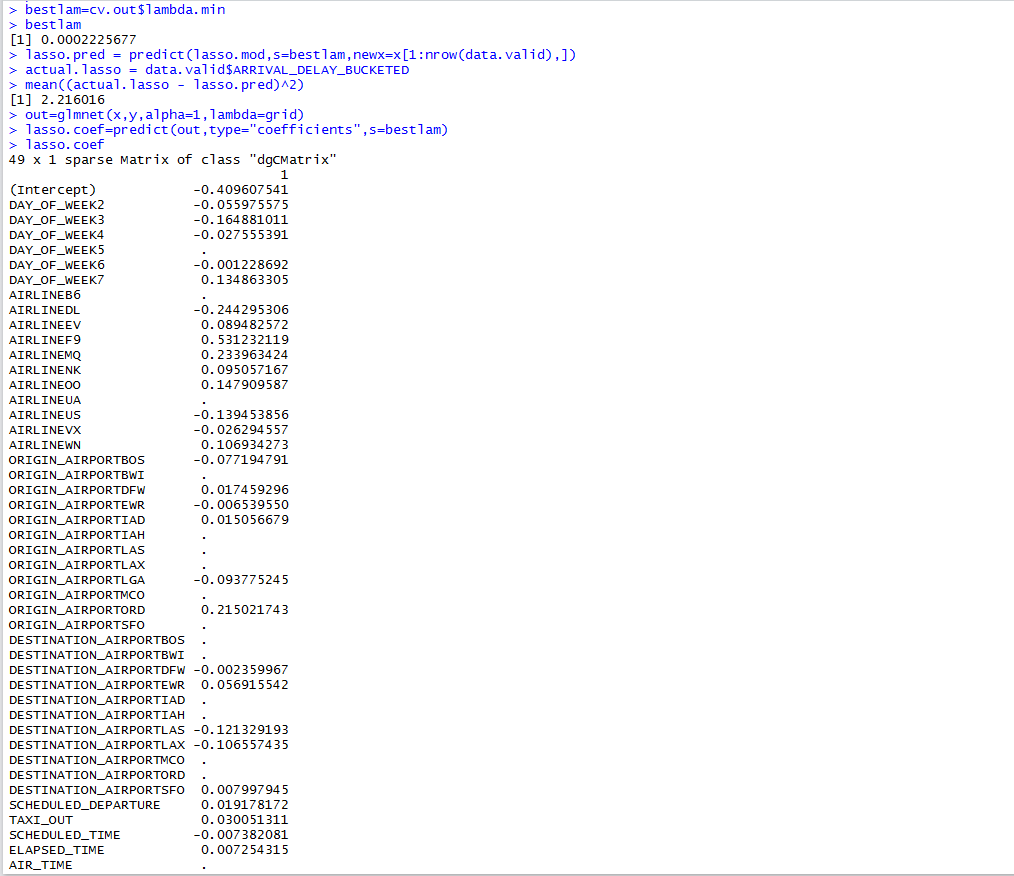
actual.lasso = data.valid$ARRIVAL\_DELAY\_BUCKETED

mean((actual.lasso - lasso.pred)^2)

out=glmnet(x,y,alpha=1,lambda=grid)

lasso.coef=predict(out,type="coefficients",s=bestlam)

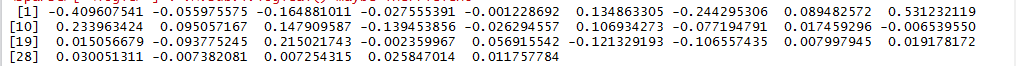
lasso.coef





lasso.coef[lasso.coef!=0]





## Ridge regression

set.seed(12356)

x = model.matrix(ARRIVAL\_DELAY\_BUCKETED~.,data.train)[,-1]

y = data.train$ARRIVAL\_DELAY\_BUCKETED

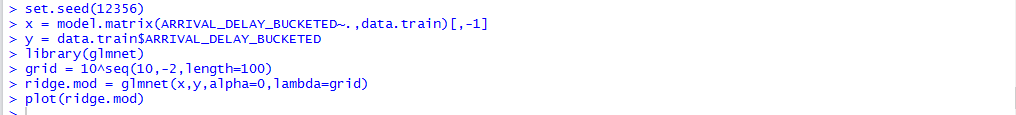
#install.packages("glmnet")

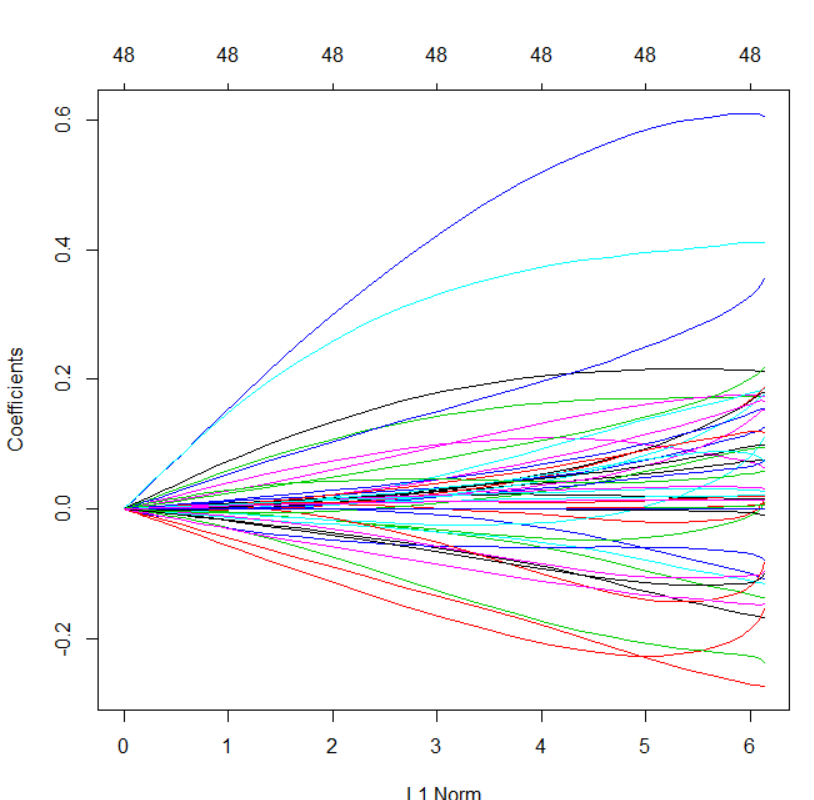
library(glmnet)

grid = 10^seq(10,-2,length=100)

ridge.mod = glmnet(x,y,alpha=0,lambda=grid)

plot(ridge.mod)



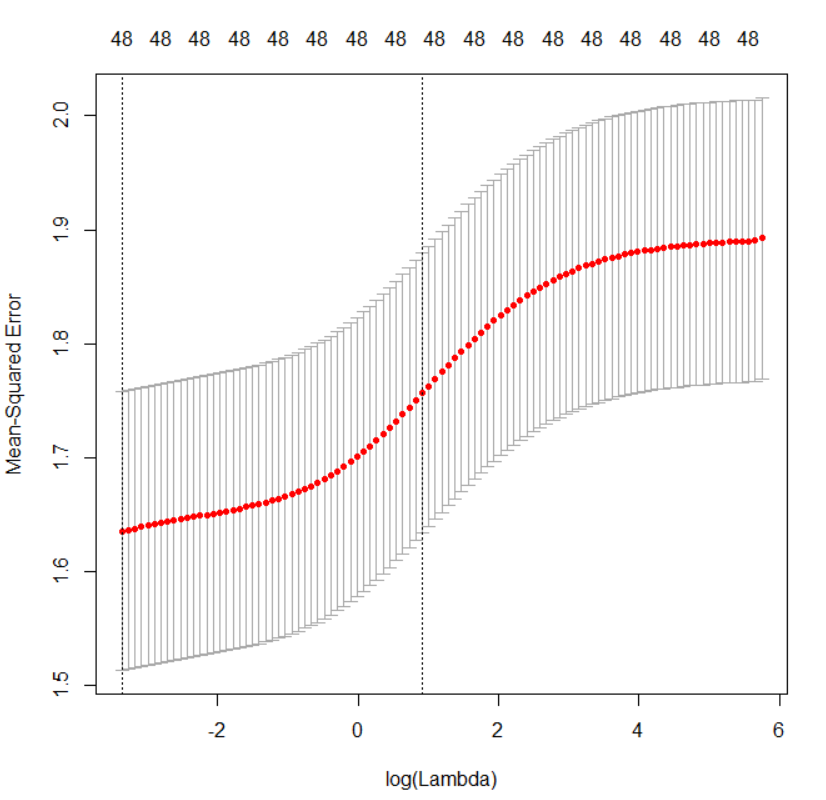


cv.out = cv.glmnet(x,y,alpha=0)

plot(cv.out)



### 11.2 Mean-Squared Error Chart



### 11.3 Ridge Model

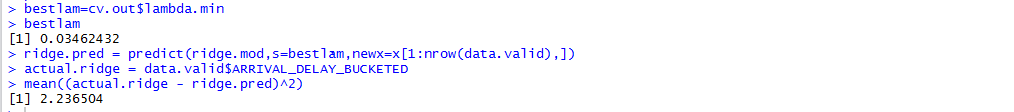
bestlam=cv.out$lambda.min

bestlam

ridge.pred = predict(ridge.mod,s=bestlam,newx=x[1:nrow(data.valid),])

actual.ridge = data.valid$ARRIVAL\_DELAY\_BUCKETED

mean((actual.ridge - ridge.pred)^2)

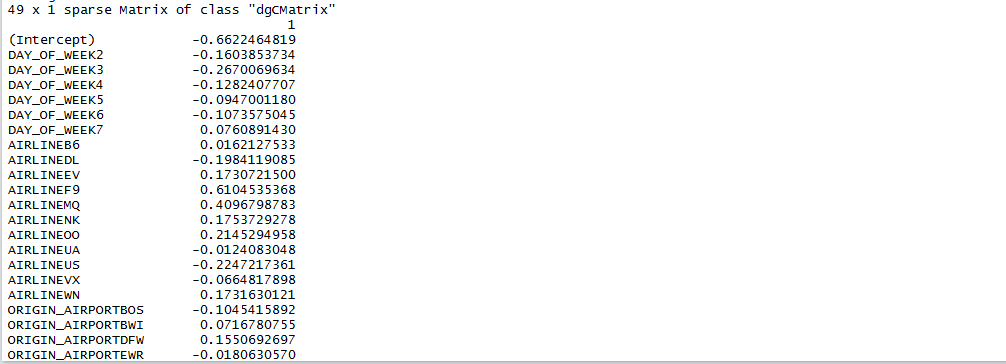


out=glmnet(x,y,alpha=0,lambda=grid)

ridge.coef=predict(out,type="coefficients",s=bestlam)

ridge.coef



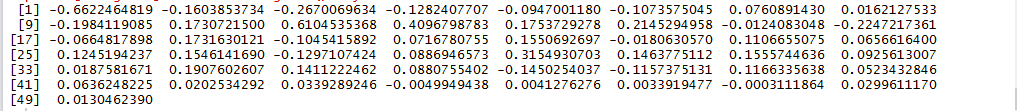






ridge.coef[ridge.coef!=0]





## Regression Tree Model

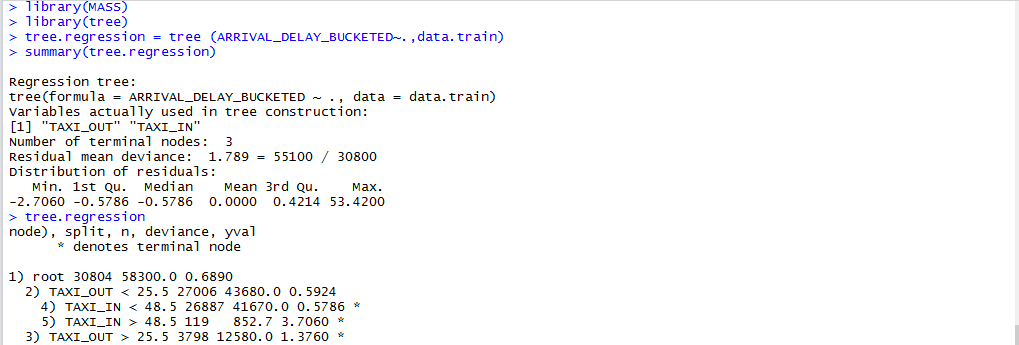
library(MASS)

library(tree)

tree.regression = tree (ARRIVAL\_DELAY\_BUCKETED~.,data.train)

summary(tree.regression)

tree.regression

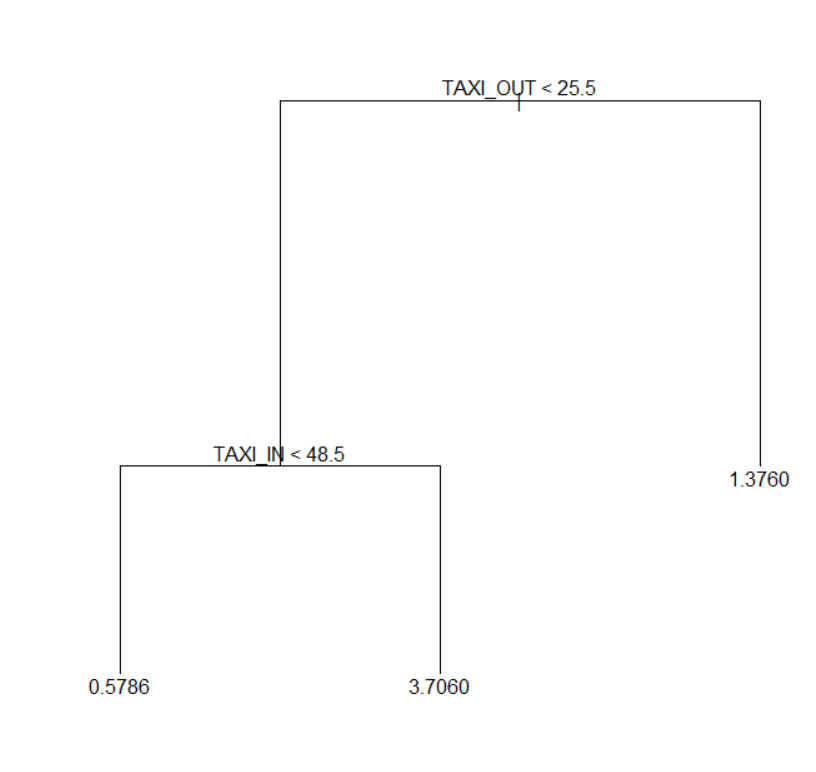


## 

### 12.1 Plotting Regression Tree

plot(tree.regression)

text(tree.regression,pretty=0)

### 12.2 Plot of tree size vs deviance

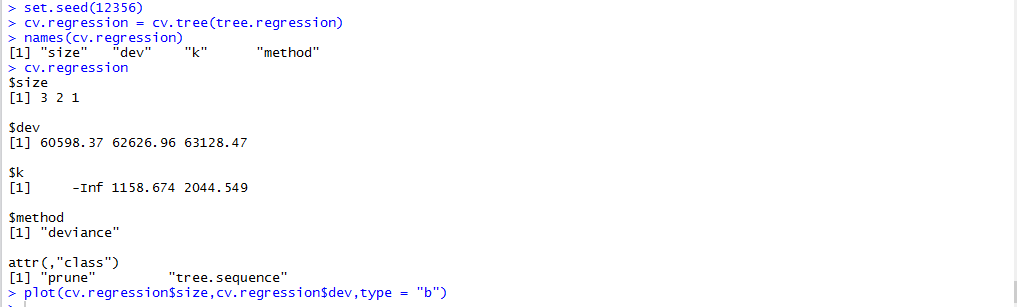
set.seed(12356)

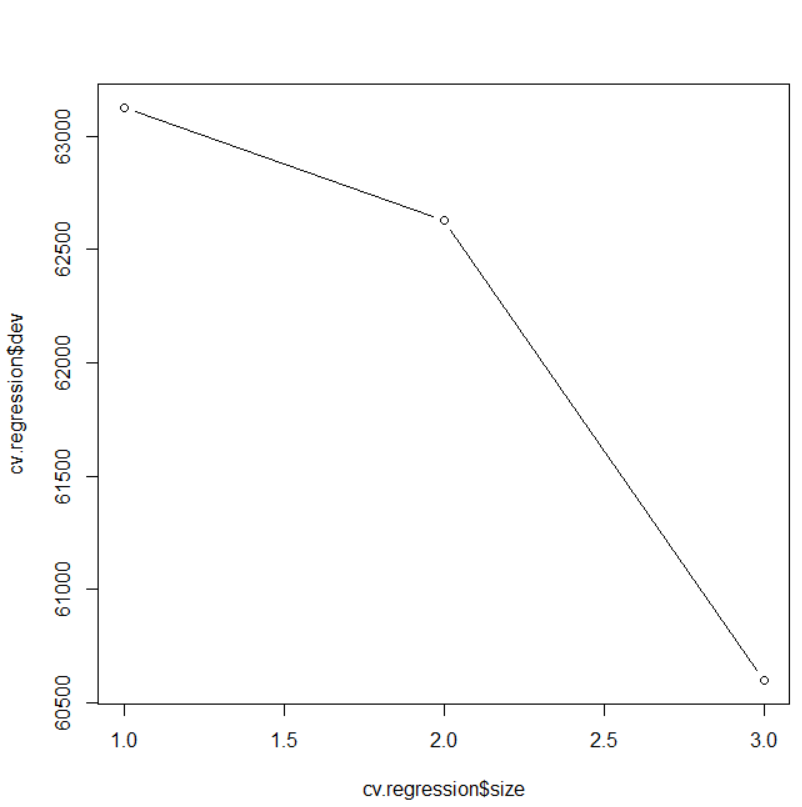
cv.regression = cv.tree(tree.regression)

names(cv.regression)

cv.regression

plot(cv.regression$size,cv.regression$dev,type = "b")





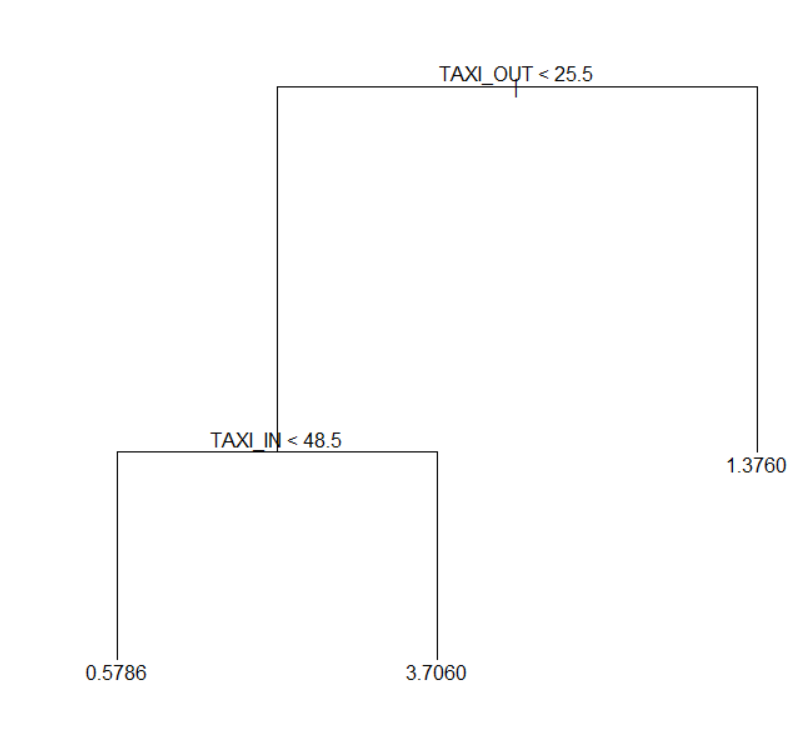
### 12.3 Pruned Regression Tree

prune.regression = prune.tree(tree.regression,best=3)

plot(prune.regression)

text(prune.regression,pretty=0)





### 12.4 Predicted vs actual delay for pruned tree

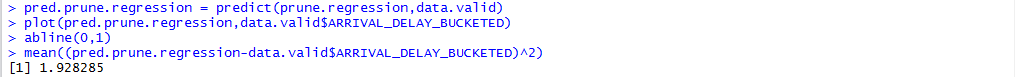
pred.prune.regression = predict(prune.regression,data.valid)

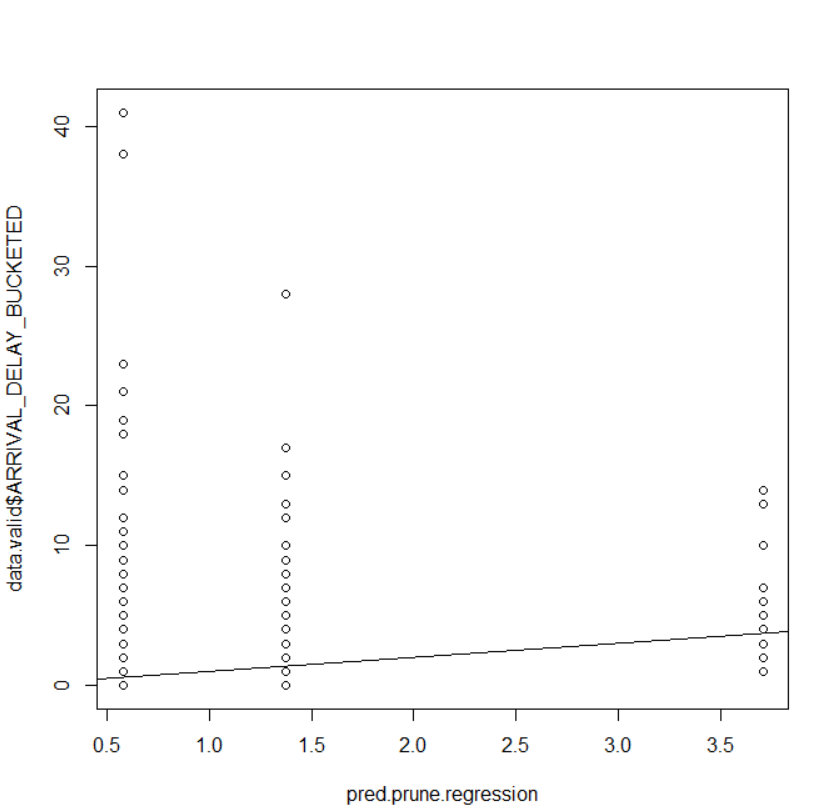
pred.prune.regression

plot(pred.prune.regression,data.valid$ARRIVAL\_DELAY\_BUCKETED)

abline(0,1)

mean((pred.prune.regression-data.valid$ARRIVAL\_DELAY\_BUCKETED)^2)





## Classification

### 13.1 Data cleaning for classification

file1=read.csv("flights1.csv")

head(file1)

data.work1=file1

data.work1=na.omit(data.work1)

data.work1$ARRIVAL\_DELAY\_BUCKETED <- ifelse(data.work1$ARRIVAL\_DELAY\_BUCKETED <= 0, 0, data.work1$ARRIVAL\_DELAY\_BUCKETED)

#We are performing 2 class classification with two hours delay and 4 hours delay.

data.work1$FINAL2=ifelse(data.work1$ARRIVAL\_DELAY\_BUCKETED <= 8, 0, 1)

data.work1$FINAL4=ifelse(data.work1$ARRIVAL\_DELAY\_BUCKETED <= 16, 0, 1)

data.work1$ARRIVAL\_DELAY\_BUCKETED\_TEMP=NULL

data.work1$ARRIVAL\_DELAY\_BUCKETED=NULL

data.work1$DIVERTED=NULL

data.work1$CANCELLED=NULL

data.work1$CANCELLATION\_REASON=NULL

data.work1$DAY\_OF\_WEEK <- as.factor(data.work1$DAY\_OF\_WEEK)

data.work1$AIRLINE <- as.factor(data.work1$AIRLINE)

data.work1$YEAR=NULL

data.work1$DAY=NULL

data.work1$MONTH=NULL

data.work1$FLIGHT\_NUMBER=NULL

data.work1$TAIL\_NUMBER=NULL

data.work1$SCHEDULED\_DEPARTURE=data.work1$SCHEDULED\_DEPARTURE/100

data.work1$SCHEDULED\_DEPARTURE=round(data.work1$SCHEDULED\_DEPARTURE)

data.work1$DEPARTURE\_TIME=NULL

data.work1$WHEELS\_OFF=NULL

data.work1$WHEELS\_ON=NULL

data.work1$SCHEDULED\_ARRIVAL=data.work1$SCHEDULED\_ARRIVAL/100

data.work1$SCHEDULED\_ARRIVAL=round(data.work1$SCHEDULED\_ARRIVAL)

data.work1$ARRIVAL\_TIME=NULL

data.work1$AIR\_SYSTEM\_DELAY=NULL

data.work1$SECURITY\_DELAY=NULL

data.work1$AIRLINE\_DELAY=NULL

data.work1$LATE\_AIRCRAFT\_DELAY=NULL

data.work1$WEATHER\_DELAY=NULL

data.work1$DEPARTURE\_DELAY=NULL

data.work1$ARRIVAL\_DELAY=NULL

data.work1$TAXI\_OUT=NULL

data.work1$TAXI\_IN=NULL

data.work1$ELAPSED\_TIME=NULL

data.work1$SCHEDULED\_DEPARTURE <- as.factor(data.work1$SCHEDULED\_DEPARTURE)

data.work1$SCHEDULED\_ARRIVAL <- as.factor(data.work1$SCHEDULED\_ARRIVAL)

data.work2=data.work1

data.work1$FINAL4=NULL

data.work2$FINAL2=NULL

data.work1$FINAL2=as.factor(data.work1$FINAL2)

data.work2$FINAL4=as.factor(data.work2$FINAL4)

### 13.2 Creating training and validation Dataset

set.seed(1000)

train=sample(nrow(data.work1),0.7\*nrow(data.work1))

data.train <- data.work1[train,]

data.valid <-data.work1[-train,]

data.validx <- data.work2[-train,]

data.validy <- data.work1[-train,]

### 13.3 Sampling for classification

set.seed(1000)

data.train <- data.work1[train,]

library("rpart")

library("ROSE")

data.over.train=ovun.sample(FINAL2~.,data.train, p=0.5, method ="over")$data

data.valid <-data.work1[-train,]

### 13.4 Data synthesis for 2 hours delay

data.rose <- ROSE(FINAL2 ~ ., data = data.train, seed = 1)$data

table(data.rose$FINAL2)

### Data Synthesis ##

library(tree)

tree.final1 = rpart(FINAL2~.,data.rose)

summary(tree.final1)

tree.final1

pred.full.final1 = predict(tree.final1,data.valid,type="class")

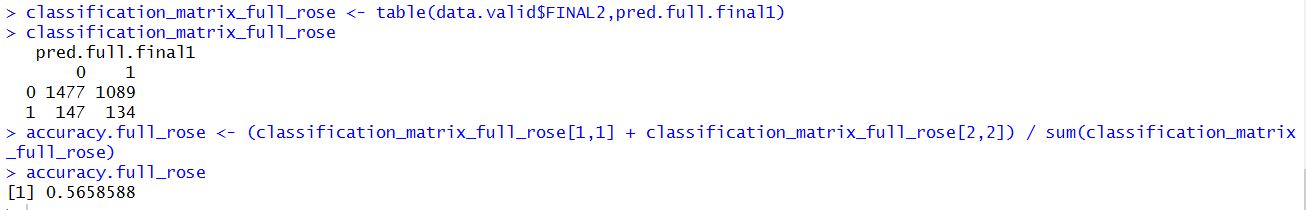
pred.full.final1

classification\_matrix\_full\_rose <- table(data.valid$FINAL2,pred.full.final1)

classification\_matrix\_full\_rose

accuracy.full\_rose <- (classification\_matrix\_full\_rose[1,1] + classification\_matrix\_full\_rose[2,2]) / sum(classification\_matrix\_full\_rose)

accuracy.full\_rose



### 13.4 Classification tree with oversampling for 2 hours delay

### Over Sampling ##

library(tree)

tree.final2 = rpart(FINAL2~.,data.over.train)

summary(tree.final2)

tree.final2

pred.full.final2 = predict(tree.final2,data.valid,type="class")

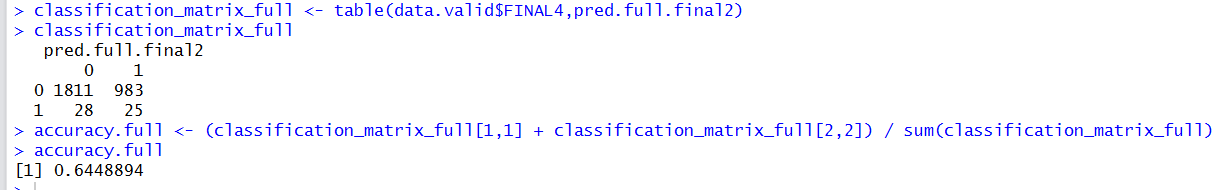
pred.full.final2

classification\_matrix\_full <- table(data.valid$FINAL2,pred.full.final2)

classification\_matrix\_full

accuracy.full <- (classification\_matrix\_full[1,1] + classification\_matrix\_full[2,2]) / sum(classification\_matrix\_full)

accuracy.full



### 13.5 Data synthesis for 4 hours delay

#4 hour delay

set.seed(1000)

train=sample(nrow(data.work1),0.7\*nrow(data.work1))

data.train4 <- data.work2[train,]

data.valid4 <-data.work2[-train,]

set.seed(1000)

data.train4 <- data.work2[train,]

library("rpart")

library("ROSE")

data.over.train4=ovun.sample(FINAL4~.,data.train4, p=0.3, method ="over")$data

data.valid <-data.work2[-train,]

data.rose <- ROSE(FINAL4 ~ ., data = data.over.train4, seed = 1)$data

table(data.rose$FINAL4)

### Data Synthesis ##

library(tree)

tree.final1 = rpart(FINAL4~.,data.rose)

summary(tree.final1)

tree.final1

pred.full.final14 = predict(tree.final1,data.valid,type="class")

pred.full.final14

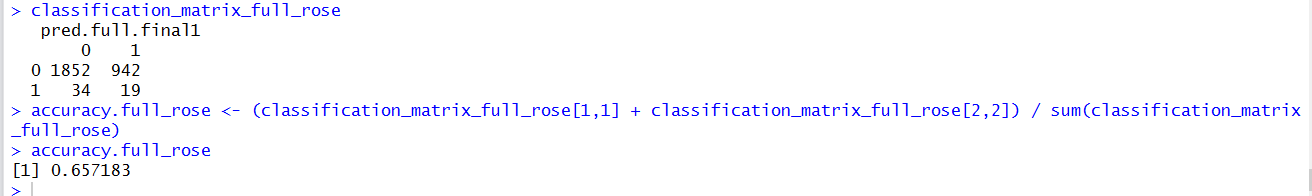
classification\_matrix\_full\_rose <- table(data.valid$FINAL4,pred.full.final14)

table(data.valid$FINAL4,pred.full.final14)

classification\_matrix\_full\_rose

accuracy.full\_rose <- (classification\_matrix\_full\_rose[1,1] + classification\_matrix\_full\_rose[2,2]) / sum(classification\_matrix\_full\_rose)

accuracy.full\_rose



13.6 Oversampling for 4 hours delay

### Over Sampling ##

library(tree)

tree.final24 = rpart(FINAL4~.,data.over.train4)

summary(tree.final24)

tree.final24

pred.full.final24 = predict(tree.final24,data.valid,type="class")

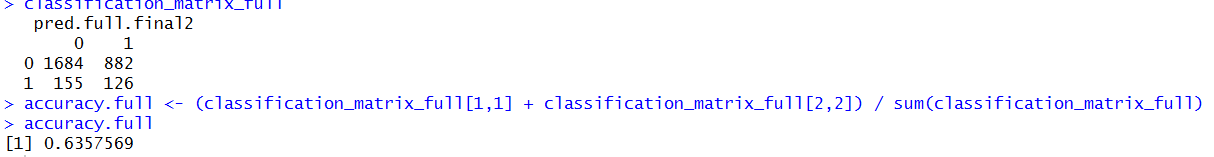
pred.full.final24

classification\_matrix\_full <- table(data.valid$FINAL4,pred.full.final2)

classification\_matrix\_full

accuracy.full <- (classification\_matrix\_full[1,1] + classification\_matrix\_full[2,2]) / sum(classification\_matrix\_full)

accuracy.full



## Association Rules

### 14.1 Data cleaning for association rules

file1=read.csv("flights.csv")

head(file1)

data.work1=file1

data.work1$DIVERTED=NULL

data.work1$CANCELLED=NULL

data.work1$CANCELLATION\_REASON=NULL

data.work1$YEAR=NULL

data.work1$TAIL\_NUMBER=NULL

data.work1$FLIGHT\_NUMBER=NULL

data.work1$DAY=NULL

data.work1[,7:12]=NULL

data.work1$AIR\_TIME=NULL

data.work1[,7:9]=NULL

data.work1[,10:14]=NULL

data.work1$ARRIVAL\_DELAY\_BUCKETED <- ifelse(data.work1$ARRIVAL\_DELAY<= 0, 0, data.work1$ARRIVAL\_DELAY)

data.work1$SCHEDULED\_ARRIVAL=data.work1$SCHEDULED\_ARRIVAL/100

data.work1$SCHEDULED\_ARRIVAL=round(data.work1$SCHEDULED\_ARRIVAL)

data.work1$SCHEDULED\_DEPARTURE=data.work1$SCHEDULED\_DEPARTURE/100

data.work1$SCHEDULED\_DEPARTURE=round(data.work1$SCHEDULED\_DEPARTURE)

data.work1$ARRIVAL\_DELAY=NULL

data.work1$ARRIVAL\_TIME=NULL

data.work1$FINAL2=data.work1$ARRIVAL\_DELAY\_BUCKETED/120

data.work1$FINAL2 <- ifelse(data.work1$FINAL2<= 1, 0, 1)

data.work1=na.omit(data.work1)

data.work1=data.frame(lapply(data.work1, as.factor))

library("dummies")

#data.work1= dummy.data.frame(data.work1, names=c("DAY\_OF\_WEEK", "AIRLINE", "ORIGIN\_AIRPORT","DESTINATION\_AIRPORT", "SCHEDULED\_DEPARTURE", "SCHEDULED\_ARRIVAL"), sep=",")

### 14.2 Running code

library("arules")

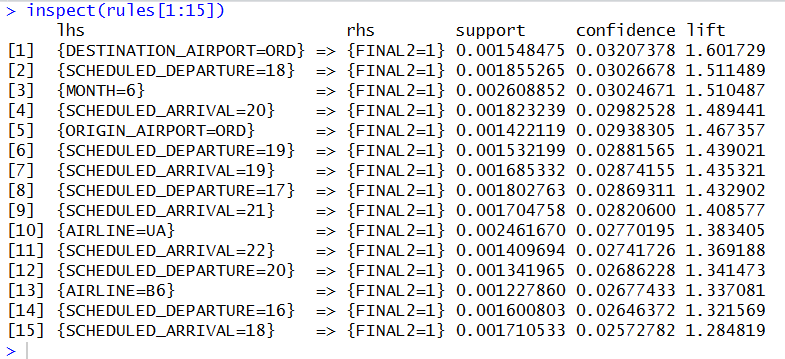
rules <- apriori(data=data.work1, parameter = list(supp=0.001, conf=0.01),

appearance = list(default="lhs", rhs="FINAL2=1"),

control = list(verbose=F))

rules <- sort(rules, decreasing = TRUE, by="confidence")

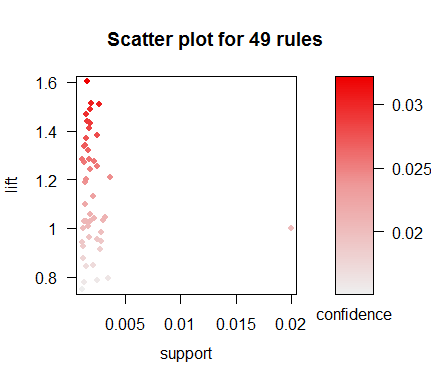
inspect(rules[1:15])



### 14.3 Plot of lift vs support with confidence shading

library("arulesViz")

plot(rules, measure=c("support","lift"), shading="confidence")



## KNN

file1=read.csv("flights1.csv")

head(file1)

data.work1=file1

data.work1=na.omit(data.work1)

data.work1$ARRIVAL\_DELAY\_BUCKETED <- ifelse(data.work1$ARRIVAL\_DELAY\_BUCKETED <= 0, 0, data.work1$ARRIVAL\_DELAY\_BUCKETED)

data.work1$FINAL2=ifelse(data.work1$ARRIVAL\_DELAY\_BUCKETED <= 8, 0, 1)

data.work1$FINAL4=ifelse(data.work1$ARRIVAL\_DELAY\_BUCKETED <= 16, 0, 1)

data.work1$ARRIVAL\_DELAY\_BUCKETED\_TEMP=NULL

data.work1$ARRIVAL\_DELAY\_BUCKETED=NULL

data.work1$DIVERTED=NULL

data.work1$CANCELLED=NULL

data.work1$CANCELLATION\_REASON=NULL

data.work1$DAY\_OF\_WEEK <- as.factor(data.work1$DAY\_OF\_WEEK)

data.work1$AIRLINE <- as.factor(data.work1$AIRLINE)

data.work1$YEAR=NULL

data.work1$DAY=NULL

data.work1$MONTH=NULL

data.work1$FLIGHT\_NUMBER=NULL

data.work1$TAIL\_NUMBER=NULL

data.work1$SCHEDULED\_DEPARTURE=data.work1$SCHEDULED\_DEPARTURE/100

data.work1$SCHEDULED\_DEPARTURE=round(data.work1$SCHEDULED\_DEPARTURE)

data.work1$DEPARTURE\_TIME=NULL

data.work1$WHEELS\_OFF=NULL

data.work1$WHEELS\_ON=NULL

data.work1$SCHEDULED\_ARRIVAL=data.work1$SCHEDULED\_ARRIVAL/100

data.work1$SCHEDULED\_ARRIVAL=round(data.work1$SCHEDULED\_ARRIVAL)

data.work1$ARRIVAL\_TIME=NULL

data.work1$AIR\_SYSTEM\_DELAY=NULL

data.work1$SECURITY\_DELAY=NULL

data.work1$AIRLINE\_DELAY=NULL

data.work1$LATE\_AIRCRAFT\_DELAY=NULL

data.work1$WEATHER\_DELAY=NULL

data.work1$DEPARTURE\_DELAY=NULL

data.work1$ARRIVAL\_DELAY=NULL

data.work1$TAXI\_OUT=NULL

data.work1$TAXI\_IN=NULL

data.work1$ELAPSED\_TIME=NULL

data.work1$SCHEDULED\_DEPARTURE <- as.factor(data.work1$SCHEDULED\_DEPARTURE)

data.work1$SCHEDULED\_ARRIVAL <- as.factor(data.work1$SCHEDULED\_ARRIVAL)

data.work2=data.work1

data.work1$FINAL4=NULL

data.work2$FINAL2=NULL

data.work1$FINAL2=as.factor(data.work1$FINAL2)

data.work2$FINAL4=as.factor(data.work2$FINAL4)

library("dummies")

data.work1= dummy.data.frame(data.work1, names=c("DAY\_OF\_WEEK", "AIRLINE", "DESTINATION\_AIRPORT", "SCHEDULED\_DEPARTURE", "SCHEDULED\_ARRIVAL"), sep=",")

data.work1= dummy.data.frame(data.work1, names=c("ORIGIN\_AIRPORT"), sep="\_")

fun <- function(x){

a <- mean(x)

b <- sd(x)

(x - a)/(b)

}

data.work1[,-93] <- apply(data.work1[,-93], 2, fun)

library("plyr")

names(data.work1)=gsub(",","",names(data.work1))

set.seed(1000)

train=sample(nrow(data.work1),0.7\*nrow(data.work1))

library("rpart")

library("ROSE")

data.train <- data.work1[train,]

data.over.train=ovun.sample(FINAL2~.,data=data.train, p=0.5, method ="over")$data

data.valid <-data.work1[-train,]

library(class)

#

train\_input <- as.matrix(data.train[,-93])

train\_output <- as.vector(data.train[,93])

validate\_input <- as.matrix(data.valid[,-93])

kmax <- 15

ER1 <- rep(0,kmax)

ER2 <- rep(0,kmax)

for (i in 1:kmax){

prediction <- knn(train\_input, train\_input,train\_output, k=i)

prediction2 <- knn(train\_input, validate\_input,train\_output, k=i)

# The confusion matrix for training data is:

CM1 <- table(prediction, data.train$FINAL2)

# The training error rate is:

ER1[i] <- (CM1[1,1]+CM1[2,2])/sum(CM1)

ER1[i]

# The confusion matrix for validation data is:

CM2 <- table(prediction2, data.valid$FINAL2)

ER2[i] <- (CM2[1,1]+CM2[2,2])/sum(CM2)

ER2[i]

}

ER1

ER2

plot(c(1,kmax),c(0,0.18),type="n", xlab="k",ylab="Error Rate")

lines(ER1,col="red")

lines(ER2,col="blue")

legend(9, 0.1, c("Training","Validation"),lty=c(1,1), col=c("red","blue"))

z <- which.max(ER2)

cat("Minimum Validation Error k:", z)

# Scoring at optimal k

prediction <- knn(train\_input, train\_input,train\_output, k=z)

prediction2 <- knn(train\_input, validate\_input,train\_output, k=z)

#prediction2

CM1 <- table(prediction, data.train$FINAL2,dnn=list('predicted','actual'))

CM2 <- table(prediction2, data.valid$FINAL2,dnn=list('predicted','actual'))

#CM1

CM2

#ER2 <- (CM2[1,2]+CM2[2,1])/sum(CM2)

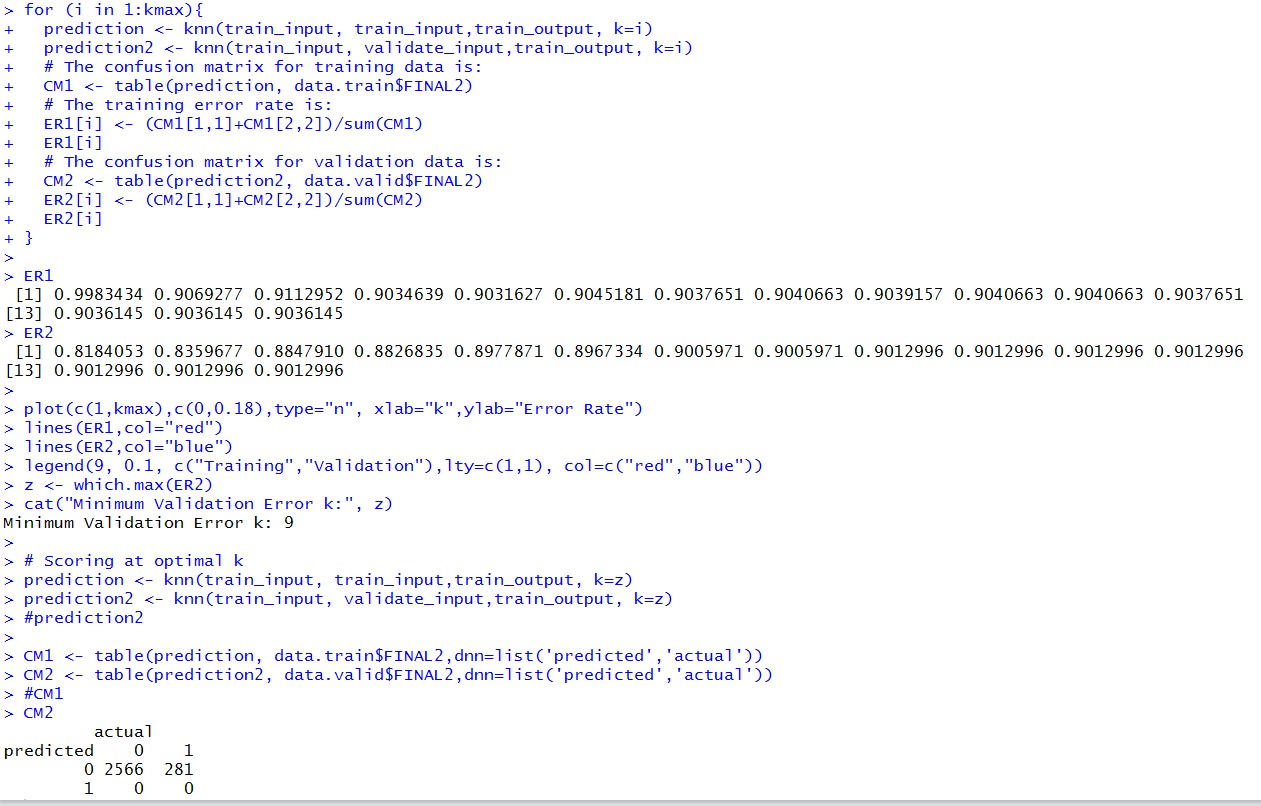
#ER2

ERClass0 <- (CM2[2,1]/(CM2[1,1]+CM2[2,1]))

ERClass1 <- (CM2[1,2]/(CM2[2,2]+CM2[1,2]))

ERClass0

ERClass1



# Note:

The association rules script needs to be run on flights.csv and all other R scripts are to be run using flights1.csv which is a subsetted dataset.