Logistic Regression

18BCE1104 - Ankita Duraphe

25/02/2021

### Logistic Regression on Smarket dataset

rm(list = ls())

library(ISLR)

names(Smarket)

## [1] "Year" "Lag1" "Lag2" "Lag3" "Lag4" "Lag5"   
## [7] "Volume" "Today" "Direction"

dim(Smarket)

## [1] 1250 9

summary(Smarket)

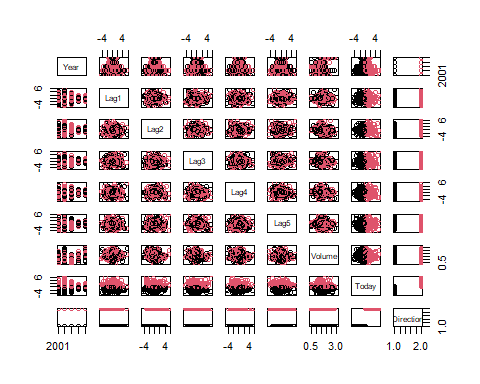
## Year Lag1 Lag2 Lag3   
## Min. :2001 Min. :-4.922000 Min. :-4.922000 Min. :-4.922000   
## 1st Qu.:2002 1st Qu.:-0.639500 1st Qu.:-0.639500 1st Qu.:-0.640000   
## Median :2003 Median : 0.039000 Median : 0.039000 Median : 0.038500   
## Mean :2003 Mean : 0.003834 Mean : 0.003919 Mean : 0.001716   
## 3rd Qu.:2004 3rd Qu.: 0.596750 3rd Qu.: 0.596750 3rd Qu.: 0.596750   
## Max. :2005 Max. : 5.733000 Max. : 5.733000 Max. : 5.733000   
## Lag4 Lag5 Volume Today   
## Min. :-4.922000 Min. :-4.92200 Min. :0.3561 Min. :-4.922000   
## 1st Qu.:-0.640000 1st Qu.:-0.64000 1st Qu.:1.2574 1st Qu.:-0.639500   
## Median : 0.038500 Median : 0.03850 Median :1.4229 Median : 0.038500   
## Mean : 0.001636 Mean : 0.00561 Mean :1.4783 Mean : 0.003138   
## 3rd Qu.: 0.596750 3rd Qu.: 0.59700 3rd Qu.:1.6417 3rd Qu.: 0.596750   
## Max. : 5.733000 Max. : 5.73300 Max. :3.1525 Max. : 5.733000   
## Direction   
## Down:602   
## Up :648   
##   
##   
##   
##

?Smarket  
str(Smarket)

## 'data.frame': 1250 obs. of 9 variables:  
## $ Year : num 2001 2001 2001 2001 2001 ...  
## $ Lag1 : num 0.381 0.959 1.032 -0.623 0.614 ...  
## $ Lag2 : num -0.192 0.381 0.959 1.032 -0.623 ...  
## $ Lag3 : num -2.624 -0.192 0.381 0.959 1.032 ...  
## $ Lag4 : num -1.055 -2.624 -0.192 0.381 0.959 ...  
## $ Lag5 : num 5.01 -1.055 -2.624 -0.192 0.381 ...  
## $ Volume : num 1.19 1.3 1.41 1.28 1.21 ...  
## $ Today : num 0.959 1.032 -0.623 0.614 0.213 ...  
## $ Direction: Factor w/ 2 levels "Down","Up": 2 2 1 2 2 2 1 2 2 2 ...

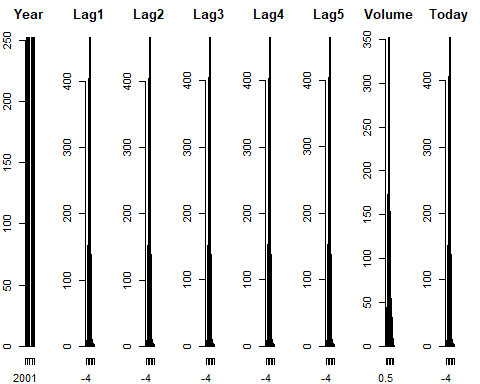
plot with pairs fn

pairs(Smarket,col=Smarket$Direction)



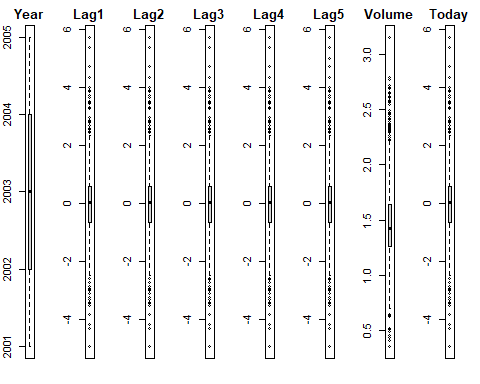
data analysis - Histogram plot

par(mar = rep(2, 4)) #To reduce the size of the margins  
par(mfrow=c(1,8))   
for(i in 1:8) {  
 hist(Smarket[,i], main=names(Smarket)[i])  
}



box plot

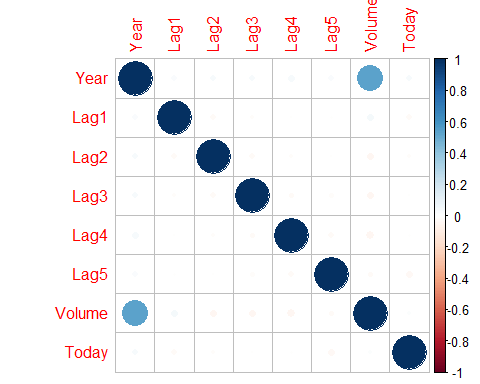
par(mar = rep(2, 4)) #To reduce the size of the margins  
par(mfrow=c(1,8))   
for(i in 1:8) {  
 boxplot(Smarket[,i], main=names(Smarket)[i])  
}



# cor(Smarket[,-9])

to find correlation between every variable

library(corrplot)  
correlations <- cor(Smarket[,1:8])  
corrplot(correlations, method="circle")



attach(Smarket)

Logistic Regression glm = Generalised linear models

glm.fits=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=binomial)  
summary(glm.fits)

##   
## Call:  
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +   
## Volume, family = binomial, data = Smarket)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.446 -1.203 1.065 1.145 1.326   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -0.126000 0.240736 -0.523 0.601  
## Lag1 -0.073074 0.050167 -1.457 0.145  
## Lag2 -0.042301 0.050086 -0.845 0.398  
## Lag3 0.011085 0.049939 0.222 0.824  
## Lag4 0.009359 0.049974 0.187 0.851  
## Lag5 0.010313 0.049511 0.208 0.835  
## Volume 0.135441 0.158360 0.855 0.392  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1731.2 on 1249 degrees of freedom  
## Residual deviance: 1727.6 on 1243 degrees of freedom  
## AIC: 1741.6  
##   
## Number of Fisher Scoring iterations: 3

coef(glm.fits)

## (Intercept) Lag1 Lag2 Lag3 Lag4 Lag5   
## -0.126000257 -0.073073746 -0.042301344 0.011085108 0.009358938 0.010313068   
## Volume   
## 0.135440659

summary(glm.fits)$coef

## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -0.126000257 0.24073574 -0.5233966 0.6006983  
## Lag1 -0.073073746 0.05016739 -1.4565986 0.1452272  
## Lag2 -0.042301344 0.05008605 -0.8445733 0.3983491  
## Lag3 0.011085108 0.04993854 0.2219750 0.8243333  
## Lag4 0.009358938 0.04997413 0.1872757 0.8514445  
## Lag5 0.010313068 0.04951146 0.2082966 0.8349974  
## Volume 0.135440659 0.15835970 0.8552723 0.3924004

summary(glm.fits)$coef[,4]

## (Intercept) Lag1 Lag2 Lag3 Lag4 Lag5   
## 0.6006983 0.1452272 0.3983491 0.8243333 0.8514445 0.8349974   
## Volume   
## 0.3924004

glm.probs=predict(glm.fits,type="response")  
glm.probs[1:10]

## 1 2 3 4 5 6 7 8   
## 0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565 0.4926509 0.5092292   
## 9 10   
## 0.5176135 0.4888378

glm.pred=rep("Down",1250)  
glm.pred[glm.probs>.5]="Up"  
table(glm.pred,Direction)

## Direction  
## glm.pred Down Up  
## Down 145 141  
## Up 457 507

(507+145)/1250

## [1] 0.5216

mean(glm.pred==Direction)

## [1] 0.5216

train=(Year<2005)  
Smarket.2005=Smarket[!train,]  
dim(Smarket.2005)

## [1] 252 9

Direction.2005=Direction[!train]  
glm.fits=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=binomial,subset=train)  
glm.probs=predict(glm.fits,Smarket.2005,type="response")  
glm.pred=rep("Down",252)  
glm.pred[glm.probs>.5]="Up"

other way

glm.pred=ifelse(glm.probs==.5, "Up", "Down")  
table(glm.pred,Direction.2005)

## Direction.2005  
## glm.pred Down Up  
## Down 111 141

mean(glm.pred==Direction.2005)

## [1] 0.4404762

mean(glm.pred!=Direction.2005)

## [1] 0.5595238

fit with two predictor variables

glm.fits=glm(Direction~Lag1+Lag2,data=Smarket,family=binomial,subset=train)  
glm.probs=predict(glm.fits,Smarket.2005,type="response")  
glm.pred=rep("Down",252)  
glm.pred[glm.probs>.5]="Up"  
table(glm.pred,Direction.2005)

## Direction.2005  
## glm.pred Down Up  
## Down 35 35  
## Up 76 106

mean(glm.pred==Direction.2005)

## [1] 0.5595238

106/(106+76)

## [1] 0.5824176

predict with a sample data

predict(glm.fits,newdata=data.frame(Lag1=c(1.2,1.5),Lag2=c(1.1,-0.8)),type="response")

## 1 2   
## 0.4791462 0.4960939

### Logistic Regression on Admission Predict dataset

Removing previously loaded objects from the environment

rm(list = ls())

Loading the required packages

library(tidyverse)  
library(plotly)  
library(GGally)  
library(ggplot2)  
library(readr)  
library(dplyr)

Importing the dataset

data <- read.csv("Admission\_Predict.csv")

The dataset contains several parameters which are considered important during the application for Masters Programs.

The parameters included are as follows:  
1. GRE Scores (out of 340)  
2. TOEFL Scores (out of 120)  
3. University Rating (out of 5)  
4. Statement of Purpose / SOP (out of 5)  
5. Letter of Recommendation Strength / LOR (out of 5)  
6. Undergraduate GPA (out of 10)  
7. Research Experience (either 0 or 1)  
8. Chance of Admit (ranging from 0 to 1)

Inspecting the dataset

names(data)

## [1] "Serial.No." "GRE.Score" "TOEFL.Score"   
## [4] "University.Rating" "SOP" "LOR"   
## [7] "CGPA" "Research" "Chance.of.Admit"

dim(data)

## [1] 400 9

summary(data)

## Serial.No. GRE.Score TOEFL.Score University.Rating  
## Min. : 1.0 Min. :290.0 Min. : 92.0 Min. :1.000   
## 1st Qu.:100.8 1st Qu.:308.0 1st Qu.:103.0 1st Qu.:2.000   
## Median :200.5 Median :317.0 Median :107.0 Median :3.000   
## Mean :200.5 Mean :316.8 Mean :107.4 Mean :3.087   
## 3rd Qu.:300.2 3rd Qu.:325.0 3rd Qu.:112.0 3rd Qu.:4.000   
## Max. :400.0 Max. :340.0 Max. :120.0 Max. :5.000   
## SOP LOR CGPA Research   
## Min. :1.0 Min. :1.000 Min. :6.800 Min. :0.0000   
## 1st Qu.:2.5 1st Qu.:3.000 1st Qu.:8.170 1st Qu.:0.0000   
## Median :3.5 Median :3.500 Median :8.610 Median :1.0000   
## Mean :3.4 Mean :3.453 Mean :8.599 Mean :0.5475   
## 3rd Qu.:4.0 3rd Qu.:4.000 3rd Qu.:9.062 3rd Qu.:1.0000   
## Max. :5.0 Max. :5.000 Max. :9.920 Max. :1.0000   
## Chance.of.Admit   
## Min. :0.3400   
## 1st Qu.:0.6400   
## Median :0.7300   
## Mean :0.7244   
## 3rd Qu.:0.8300   
## Max. :0.9700

View(data)  
str(data)

## 'data.frame': 400 obs. of 9 variables:  
## $ Serial.No. : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ GRE.Score : int 337 324 316 322 314 330 321 308 302 323 ...  
## $ TOEFL.Score : int 118 107 104 110 103 115 109 101 102 108 ...  
## $ University.Rating: int 4 4 3 3 2 5 3 2 1 3 ...  
## $ SOP : num 4.5 4 3 3.5 2 4.5 3 3 2 3.5 ...  
## $ LOR : num 4.5 4.5 3.5 2.5 3 3 4 4 1.5 3 ...  
## $ CGPA : num 9.65 8.87 8 8.67 8.21 9.34 8.2 7.9 8 8.6 ...  
## $ Research : int 1 1 1 1 0 1 1 0 0 0 ...  
## $ Chance.of.Admit : num 0.92 0.76 0.72 0.8 0.65 0.9 0.75 0.68 0.5 0.45 ...

Checking for missing values, if any, for the given dataset

data %>%   
 is.na() %>%   
 colSums(is.na(data))

## Serial.No. GRE.Score TOEFL.Score University.Rating   
## 0 0 0 0   
## SOP LOR CGPA Research   
## 0 0 0 0   
## Chance.of.Admit   
## 0

data\_new <- data %>%   
 mutate(Label.of.Admit = Chance.of.Admit)  
head(data\_new)

## Serial.No. GRE.Score TOEFL.Score University.Rating SOP LOR CGPA Research  
## 1 1 337 118 4 4.5 4.5 9.65 1  
## 2 2 324 107 4 4.0 4.5 8.87 1  
## 3 3 316 104 3 3.0 3.5 8.00 1  
## 4 4 322 110 3 3.5 2.5 8.67 1  
## 5 5 314 103 2 2.0 3.0 8.21 0  
## 6 6 330 115 5 4.5 3.0 9.34 1  
## Chance.of.Admit Label.of.Admit  
## 1 0.92 0.92  
## 2 0.76 0.76  
## 3 0.72 0.72  
## 4 0.80 0.80  
## 5 0.65 0.65  
## 6 0.90 0.90

The variable Chance.of.Admit is excluded from predictor variable due to correlationship with variable Label.of.Admit.

data\_new <- data\_new %>%   
 select(-Chance.of.Admit)  
head(data\_new)

## Serial.No. GRE.Score TOEFL.Score University.Rating SOP LOR CGPA Research  
## 1 1 337 118 4 4.5 4.5 9.65 1  
## 2 2 324 107 4 4.0 4.5 8.87 1  
## 3 3 316 104 3 3.0 3.5 8.00 1  
## 4 4 322 110 3 3.5 2.5 8.67 1  
## 5 5 314 103 2 2.0 3.0 8.21 0  
## 6 6 330 115 5 4.5 3.0 9.34 1  
## Label.of.Admit  
## 1 0.92  
## 2 0.76  
## 3 0.72  
## 4 0.80  
## 5 0.65  
## 6 0.90

The variable Label.of.Admit is divided into category “1” (admitted) and “0” (not admitted).

data\_new$Label.of.Admit <- ifelse(data\_new$Label.of.Admit > 0.5, "1", "0")  
head(data\_new)

## Serial.No. GRE.Score TOEFL.Score University.Rating SOP LOR CGPA Research  
## 1 1 337 118 4 4.5 4.5 9.65 1  
## 2 2 324 107 4 4.0 4.5 8.87 1  
## 3 3 316 104 3 3.0 3.5 8.00 1  
## 4 4 322 110 3 3.5 2.5 8.67 1  
## 5 5 314 103 2 2.0 3.0 8.21 0  
## 6 6 330 115 5 4.5 3.0 9.34 1  
## Label.of.Admit  
## 1 1  
## 2 1  
## 3 1  
## 4 1  
## 5 1  
## 6 1

Inspecting the new dataset

names(data\_new)

## [1] "Serial.No." "GRE.Score" "TOEFL.Score"   
## [4] "University.Rating" "SOP" "LOR"   
## [7] "CGPA" "Research" "Label.of.Admit"

dim(data\_new)

## [1] 400 9

summary(data\_new)

## Serial.No. GRE.Score TOEFL.Score University.Rating  
## Min. : 1.0 Min. :290.0 Min. : 92.0 Min. :1.000   
## 1st Qu.:100.8 1st Qu.:308.0 1st Qu.:103.0 1st Qu.:2.000   
## Median :200.5 Median :317.0 Median :107.0 Median :3.000   
## Mean :200.5 Mean :316.8 Mean :107.4 Mean :3.087   
## 3rd Qu.:300.2 3rd Qu.:325.0 3rd Qu.:112.0 3rd Qu.:4.000   
## Max. :400.0 Max. :340.0 Max. :120.0 Max. :5.000   
## SOP LOR CGPA Research   
## Min. :1.0 Min. :1.000 Min. :6.800 Min. :0.0000   
## 1st Qu.:2.5 1st Qu.:3.000 1st Qu.:8.170 1st Qu.:0.0000   
## Median :3.5 Median :3.500 Median :8.610 Median :1.0000   
## Mean :3.4 Mean :3.453 Mean :8.599 Mean :0.5475   
## 3rd Qu.:4.0 3rd Qu.:4.000 3rd Qu.:9.062 3rd Qu.:1.0000   
## Max. :5.0 Max. :5.000 Max. :9.920 Max. :1.0000   
## Label.of.Admit   
## Length:400   
## Class :character   
## Mode :character   
##   
##   
##

View(data\_new)  
str(data\_new)

## 'data.frame': 400 obs. of 9 variables:  
## $ Serial.No. : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ GRE.Score : int 337 324 316 322 314 330 321 308 302 323 ...  
## $ TOEFL.Score : int 118 107 104 110 103 115 109 101 102 108 ...  
## $ University.Rating: int 4 4 3 3 2 5 3 2 1 3 ...  
## $ SOP : num 4.5 4 3 3.5 2 4.5 3 3 2 3.5 ...  
## $ LOR : num 4.5 4.5 3.5 2.5 3 3 4 4 1.5 3 ...  
## $ CGPA : num 9.65 8.87 8 8.67 8.21 9.34 8.2 7.9 8 8.6 ...  
## $ Research : int 1 1 1 1 0 1 1 0 0 0 ...  
## $ Label.of.Admit : chr "1" "1" "1" "1" ...

Converting the response variable datatype from chr to factor

data\_new$Label.of.Admit <- as.factor(data\_new$Label.of.Admit)

Again inspecting the new dataset

names(data\_new)

## [1] "Serial.No." "GRE.Score" "TOEFL.Score"   
## [4] "University.Rating" "SOP" "LOR"   
## [7] "CGPA" "Research" "Label.of.Admit"

dim(data\_new)

## [1] 400 9

summary(data\_new)

## Serial.No. GRE.Score TOEFL.Score University.Rating  
## Min. : 1.0 Min. :290.0 Min. : 92.0 Min. :1.000   
## 1st Qu.:100.8 1st Qu.:308.0 1st Qu.:103.0 1st Qu.:2.000   
## Median :200.5 Median :317.0 Median :107.0 Median :3.000   
## Mean :200.5 Mean :316.8 Mean :107.4 Mean :3.087   
## 3rd Qu.:300.2 3rd Qu.:325.0 3rd Qu.:112.0 3rd Qu.:4.000   
## Max. :400.0 Max. :340.0 Max. :120.0 Max. :5.000   
## SOP LOR CGPA Research Label.of.Admit  
## Min. :1.0 Min. :1.000 Min. :6.800 Min. :0.0000 0: 35   
## 1st Qu.:2.5 1st Qu.:3.000 1st Qu.:8.170 1st Qu.:0.0000 1:365   
## Median :3.5 Median :3.500 Median :8.610 Median :1.0000   
## Mean :3.4 Mean :3.453 Mean :8.599 Mean :0.5475   
## 3rd Qu.:4.0 3rd Qu.:4.000 3rd Qu.:9.062 3rd Qu.:1.0000   
## Max. :5.0 Max. :5.000 Max. :9.920 Max. :1.0000

View(data\_new)  
str(data\_new)

## 'data.frame': 400 obs. of 9 variables:  
## $ Serial.No. : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ GRE.Score : int 337 324 316 322 314 330 321 308 302 323 ...  
## $ TOEFL.Score : int 118 107 104 110 103 115 109 101 102 108 ...  
## $ University.Rating: int 4 4 3 3 2 5 3 2 1 3 ...  
## $ SOP : num 4.5 4 3 3.5 2 4.5 3 3 2 3.5 ...  
## $ LOR : num 4.5 4.5 3.5 2.5 3 3 4 4 1.5 3 ...  
## $ CGPA : num 9.65 8.87 8 8.67 8.21 9.34 8.2 7.9 8 8.6 ...  
## $ Research : int 1 1 1 1 0 1 1 0 0 0 ...  
## $ Label.of.Admit : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 1 1 ...

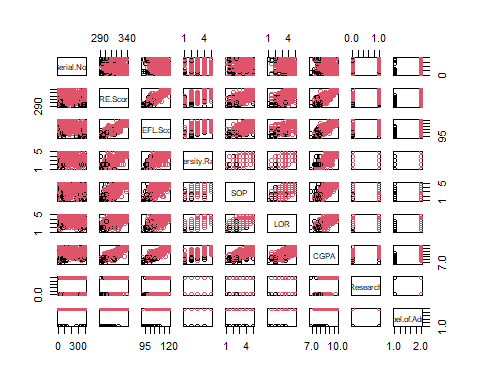
Checking for missing values, if any, for the new dataset

data\_new %>%   
 is.na() %>%   
 colSums(is.na(data\_new))

## Serial.No. GRE.Score TOEFL.Score University.Rating   
## 0 0 0 0   
## SOP LOR CGPA Research   
## 0 0 0 0   
## Label.of.Admit   
## 0

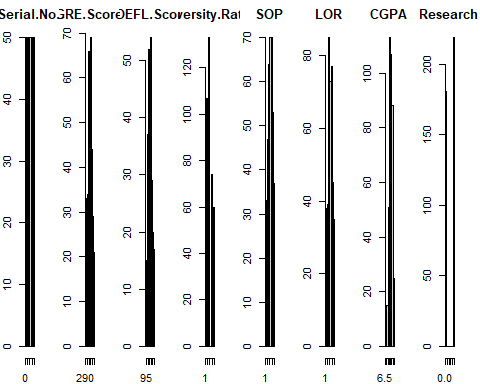
Plot with pairs fn

pairs(data\_new,col=data\_new$Label.of.Admit)



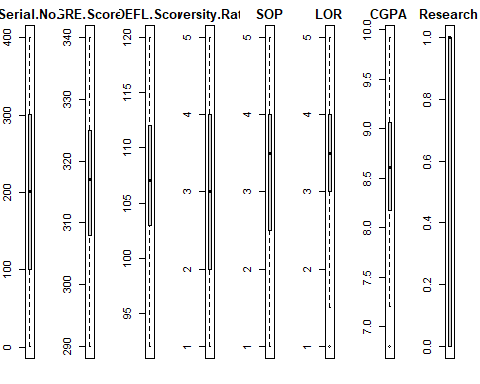
Data analysis: Histogram plot

par(mar = rep(2, 4)) #To reduce the size of the margins  
par(mfrow=c(1,8))   
for(i in 1:8) {  
 hist(data\_new[,i], main=names(data\_new)[i])  
}



Data analysis: Box plot

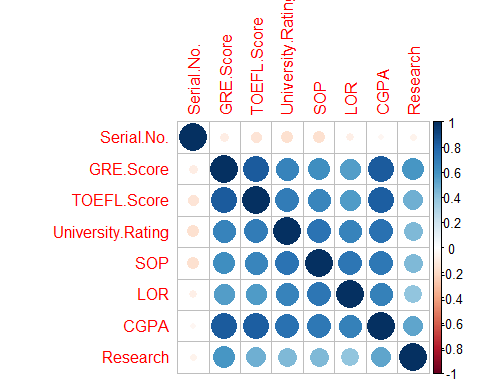
par(mar = rep(2, 4)) #To reduce the size of the margins  
par(mfrow=c(1,8))  
for(i in 1:8) {  
 boxplot(data\_new[,i], main=names(data\_new)[i])  
}



# cor(data\_new[,-9])

To find correlation between every variable

library(corrplot)  
correlations <- cor(data\_new[,1:8])  
corrplot(correlations, method="circle")



attach(data\_new)

Logistic Regression glm = Generalised linear models

glm.fits=glm(Label.of.Admit~GRE.Score+TOEFL.Score+University.Rating+SOP+LOR+CGPA+Research,data=data\_new,family=binomial)  
summary(glm.fits)

##   
## Call:  
## glm(formula = Label.of.Admit ~ GRE.Score + TOEFL.Score + University.Rating +   
## SOP + LOR + CGPA + Research, family = binomial, data = data\_new)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.07550 0.02612 0.09153 0.25583 1.75057   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -53.64616 10.76670 -4.983 6.27e-07 \*\*\*  
## GRE.Score 0.03567 0.03751 0.951 0.3417   
## TOEFL.Score 0.13754 0.08480 1.622 0.1048   
## University.Rating -0.63100 0.36018 -1.752 0.0798 .   
## SOP -0.61999 0.36321 -1.707 0.0878 .   
## LOR 1.04855 0.43932 2.387 0.0170 \*   
## CGPA 3.82655 0.89652 4.268 1.97e-05 \*\*\*  
## Research -0.12735 0.60877 -0.209 0.8343   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 237.37 on 399 degrees of freedom  
## Residual deviance: 120.66 on 392 degrees of freedom  
## AIC: 136.66  
##   
## Number of Fisher Scoring iterations: 7

coef(glm.fits)

## (Intercept) GRE.Score TOEFL.Score University.Rating   
## -53.64615909 0.03566925 0.13754157 -0.63100052   
## SOP LOR CGPA Research   
## -0.61999341 1.04854676 3.82654992 -0.12734621

summary(glm.fits)$coef

## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -53.64615909 10.76670298 -4.9825986 6.273601e-07  
## GRE.Score 0.03566925 0.03751175 0.9508822 3.416642e-01  
## TOEFL.Score 0.13754157 0.08480451 1.6218661 1.048320e-01  
## University.Rating -0.63100052 0.36018338 -1.7518868 7.979327e-02  
## SOP -0.61999341 0.36320521 -1.7070058 8.782097e-02  
## LOR 1.04854676 0.43932472 2.3867238 1.699926e-02  
## CGPA 3.82654992 0.89651828 4.2682341 1.970265e-05  
## Research -0.12734621 0.60877306 -0.2091850 8.343038e-01

summary(glm.fits)$coef[,4]

## (Intercept) GRE.Score TOEFL.Score University.Rating   
## 6.273601e-07 3.416642e-01 1.048320e-01 7.979327e-02   
## SOP LOR CGPA Research   
## 8.782097e-02 1.699926e-02 1.970265e-05 8.343038e-01

glm.probs=predict(glm.fits,type="response")  
glm.probs[1:10]

## 1 2 3 4 5 6 7 8   
## 0.9999798 0.9978843 0.9114631 0.9898104 0.9776952 0.9988366 0.9888739 0.9264653   
## 9 10   
## 0.8129702 0.9911679

glm.pred=rep("0",400)  
glm.pred[glm.probs>.5]="1"  
table(glm.pred,Label.of.Admit)

## Label.of.Admit  
## glm.pred 0 1  
## 0 17 4  
## 1 18 361

(17+361)/400 #Calculated from the Confusion Matrix

## [1] 0.945

mean(glm.pred==Label.of.Admit)

## [1] 0.945

train=(Serial.No.<320)  
data\_new.320=data\_new[!train,]  
dim(data\_new.320)

## [1] 81 9

Label.of.Admit.320=Label.of.Admit[!train]  
glm.fits=glm(Label.of.Admit~GRE.Score+TOEFL.Score+University.Rating+SOP+LOR+CGPA+Research,data=data\_new,family=binomial,subset=train) #Going to eliminate the variables that has little value, - GRE.Score, TOEFL.Score and Research.  
glm.probs=predict(glm.fits,data\_new.320,type="response")  
glm.pred=rep("0",81)   
glm.pred[glm.probs>.5]="1"

Aliter

glm.pred=ifelse(glm.probs==.5, "1", "0")  
table(glm.pred,Label.of.Admit.320)

## Label.of.Admit.320  
## glm.pred 0 1  
## 0 11 70

mean(glm.pred==Label.of.Admit.320)

## [1] 0.1358025

mean(glm.pred!=Label.of.Admit.320)

## [1] 0.8641975

Fit with four predictor variables

glm.fits=glm(Label.of.Admit~University.Rating+SOP+LOR+CGPA,data=data\_new,family=binomial,subset=train) #Going to eliminate the variables that has little value, - GRE.Score, TOEFL.Score and Research.  
glm.probs=predict(glm.fits,data\_new.320,type="response")  
glm.pred=rep("0",81)  
glm.pred[glm.probs>.5]="1"  
table(glm.pred,Label.of.Admit.320)

## Label.of.Admit.320  
## glm.pred 0 1  
## 0 4 2  
## 1 7 68

mean(glm.pred==Label.of.Admit.320)

## [1] 0.8888889

68/(68+7) #Calculated from the Confusion Matrix

## [1] 0.9066667

Predict with a sample data

predict(glm.fits,newdata=data.frame(University.Rating=c(4,5,5,5),SOP=c(4,5,5,4),LOR=c(3.5,4.5,5,4.5),CGPA=c(9.8,9.78,9.76,9.76)),type="response")

## 1 2 3 4   
## 0.9998902 0.9998980 0.9999357 0.9999447

### Logistic Regression on binary\_Admit dataset

Removing previously loaded objects from the environment

rm(list = ls())

Loading the required packages

library(tidyverse)  
library(plotly)  
library(GGally)  
library(ggplot2)  
library(readr)  
library(dplyr)

Importing the dataset

data <- read.csv("binary\_Admit.csv")

Inspecting the dataset

names(data)

## [1] "admit" "gre" "gpa" "rank"

dim(data)

## [1] 400 4

summary(data)

## admit gre gpa rank   
## Min. :0.0000 Min. :220.0 Min. :2.260 Min. :1.000   
## 1st Qu.:0.0000 1st Qu.:520.0 1st Qu.:3.130 1st Qu.:2.000   
## Median :0.0000 Median :580.0 Median :3.395 Median :2.000   
## Mean :0.3175 Mean :587.7 Mean :3.390 Mean :2.485   
## 3rd Qu.:1.0000 3rd Qu.:660.0 3rd Qu.:3.670 3rd Qu.:3.000   
## Max. :1.0000 Max. :800.0 Max. :4.000 Max. :4.000

View(data)  
str(data)

## 'data.frame': 400 obs. of 4 variables:  
## $ admit: int 0 1 1 1 0 1 1 0 1 0 ...  
## $ gre : int 380 660 800 640 520 760 560 400 540 700 ...  
## $ gpa : num 3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...  
## $ rank : int 3 3 1 4 4 2 1 2 3 2 ...

Checking for missing values, if any, for the given dataset

data %>%   
 is.na() %>%   
 colSums(is.na(data))

## admit gre gpa rank   
## 0 0 0 0

Converting the response variable datatype from int to factor

#data$admit <- as.factor(data$admit)

Again inspecting the new dataset

names(data)

## [1] "admit" "gre" "gpa" "rank"

dim(data)

## [1] 400 4

summary(data)

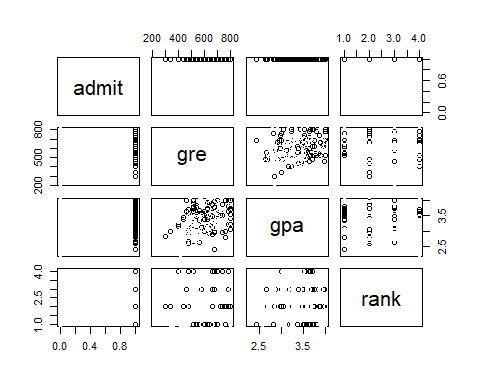
## admit gre gpa rank   
## Min. :0.0000 Min. :220.0 Min. :2.260 Min. :1.000   
## 1st Qu.:0.0000 1st Qu.:520.0 1st Qu.:3.130 1st Qu.:2.000   
## Median :0.0000 Median :580.0 Median :3.395 Median :2.000   
## Mean :0.3175 Mean :587.7 Mean :3.390 Mean :2.485   
## 3rd Qu.:1.0000 3rd Qu.:660.0 3rd Qu.:3.670 3rd Qu.:3.000   
## Max. :1.0000 Max. :800.0 Max. :4.000 Max. :4.000

View(data)  
str(data)

## 'data.frame': 400 obs. of 4 variables:  
## $ admit: int 0 1 1 1 0 1 1 0 1 0 ...  
## $ gre : int 380 660 800 640 520 760 560 400 540 700 ...  
## $ gpa : num 3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...  
## $ rank : int 3 3 1 4 4 2 1 2 3 2 ...

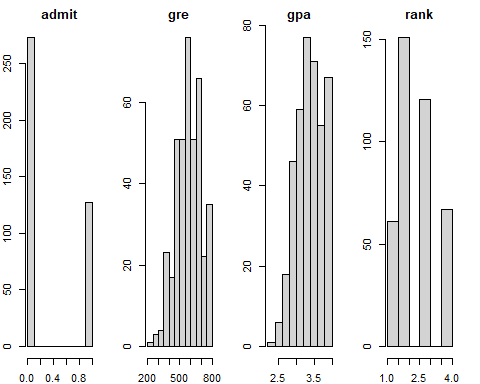
Plot with pairs fn

pairs(data,col=data$admit)



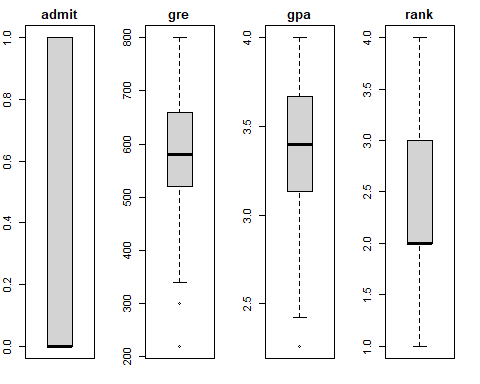
Data analysis: Histogram plot

par(mar = rep(2, 4)) #To reduce the size of the margins  
par(mfrow=c(1,4))  
for(i in 1:4) {  
 hist(data[,i], main=names(data)[i])  
}



Data analysis: Box plot

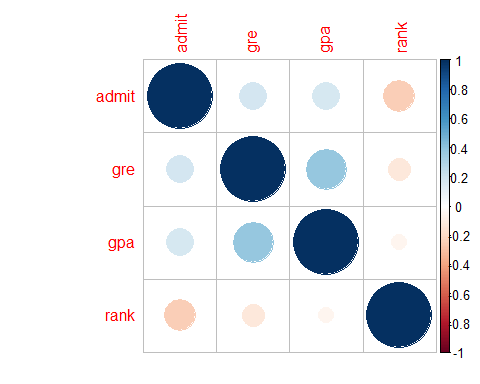
par(mar = rep(2, 4)) #To reduce the size of the margins  
par(mfrow=c(1,4))  
for(i in 1:4) {  
 boxplot(data[,i], main=names(data)[i])  
}



# cor(data[,-1])

To find correlation between every variable

library(corrplot)  
correlations <- cor(data[,1:4])  
corrplot(correlations, method="circle")



attach(data)

Logistic Regression glm = Generalised linear models

glm.fits=glm(admit~gre+gpa+rank,data=data,family=binomial)  
summary(glm.fits)

##   
## Call:  
## glm(formula = admit ~ gre + gpa + rank, family = binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.5802 -0.8848 -0.6382 1.1575 2.1732   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.449548 1.132846 -3.045 0.00233 \*\*   
## gre 0.002294 0.001092 2.101 0.03564 \*   
## gpa 0.777014 0.327484 2.373 0.01766 \*   
## rank -0.560031 0.127137 -4.405 1.06e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 499.98 on 399 degrees of freedom  
## Residual deviance: 459.44 on 396 degrees of freedom  
## AIC: 467.44  
##   
## Number of Fisher Scoring iterations: 4

coef(glm.fits)

## (Intercept) gre gpa rank   
## -3.44954840 0.00229396 0.77701357 -0.56003139

summary(glm.fits)$coef

## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -3.44954840 1.132846009 -3.045029 2.326583e-03  
## gre 0.00229396 0.001091839 2.101005 3.564052e-02  
## gpa 0.77701357 0.327483878 2.372677 1.765968e-02  
## rank -0.56003139 0.127136989 -4.404945 1.058109e-05

summary(glm.fits)$coef[,4]

## (Intercept) gre gpa rank   
## 2.326583e-03 3.564052e-02 1.765968e-02 1.058109e-05

glm.probs=predict(glm.fits,type="response")  
glm.probs[1:10]

## 1 2 3 4 5 6 7 8   
## 0.1895527 0.3177807 0.7178136 0.1489492 0.0979542 0.3786785 0.3990411 0.2211761   
## 9 10   
## 0.2215203 0.5205019

glm.pred=rep("0",400)  
glm.pred[glm.probs>.5]="1"  
table(glm.pred,admit)

## admit  
## glm.pred 0 1  
## 0 253 98  
## 1 20 29

(29+253)/400 #Calculated from the Confusion Matrix

## [1] 0.705

mean(glm.pred==admit)

## [1] 0.705

train=(rank<4)  
data.4=data[!train,]  
dim(data.4)

## [1] 67 4

admit.4=admit[!train]  
glm.fits=glm(admit~gre+gpa+rank,data=data,family=binomial,subset=train)   
glm.probs=predict(glm.fits,data.4,type="response")  
glm.pred=rep("0",67)   
glm.pred[glm.probs>.5]="1"

Aliter

glm.pred=ifelse(glm.probs==.5, "1", "0")  
table(glm.pred,admit.4)

## admit.4  
## glm.pred 0 1  
## 0 55 12

mean(glm.pred==admit.4)

## [1] 0.8208955

mean(glm.pred!=admit.4)

## [1] 0.1791045

Fit with four predictor variables

glm.fits=glm(admit~gre+gpa,data=data,family=binomial,subset=train) #Going to eliminate the variable - rank.  
glm.probs=predict(glm.fits,data.4,type="response")  
glm.pred=rep("0",67)  
glm.pred[glm.probs>.5]="1"  
table(glm.pred,admit.4)

## admit.4  
## glm.pred 0 1  
## 0 53 12  
## 1 2 0

mean(glm.pred==admit.4)

## [1] 0.7910448

### Logistic Regression on Heart dataset

Removing previously loaded objects from the environment

rm(list = ls())

Loading the required packages

library(tidyverse)  
library(plotly)  
library(GGally)  
library(ggplot2)  
library(readr)  
library(dplyr)

Importing the dataset

data <- read.csv("heart.csv")

Inspecting the dataset

names(data)

## [1] "ï..age" "sex" "cp" "trestbps" "chol" "fbs"   
## [7] "restecg" "thalach" "exang" "oldpeak" "slope" "ca"   
## [13] "thal" "target"

dim(data)

## [1] 303 14

summary(data)

## ï..age sex cp trestbps   
## Min. :29.00 Min. :0.0000 Min. :0.000 Min. : 94.0   
## 1st Qu.:47.50 1st Qu.:0.0000 1st Qu.:0.000 1st Qu.:120.0   
## Median :55.00 Median :1.0000 Median :1.000 Median :130.0   
## Mean :54.37 Mean :0.6832 Mean :0.967 Mean :131.6   
## 3rd Qu.:61.00 3rd Qu.:1.0000 3rd Qu.:2.000 3rd Qu.:140.0   
## Max. :77.00 Max. :1.0000 Max. :3.000 Max. :200.0   
## chol fbs restecg thalach   
## Min. :126.0 Min. :0.0000 Min. :0.0000 Min. : 71.0   
## 1st Qu.:211.0 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:133.5   
## Median :240.0 Median :0.0000 Median :1.0000 Median :153.0   
## Mean :246.3 Mean :0.1485 Mean :0.5281 Mean :149.6   
## 3rd Qu.:274.5 3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:166.0   
## Max. :564.0 Max. :1.0000 Max. :2.0000 Max. :202.0   
## exang oldpeak slope ca   
## Min. :0.0000 Min. :0.00 Min. :0.000 Min. :0.0000   
## 1st Qu.:0.0000 1st Qu.:0.00 1st Qu.:1.000 1st Qu.:0.0000   
## Median :0.0000 Median :0.80 Median :1.000 Median :0.0000   
## Mean :0.3267 Mean :1.04 Mean :1.399 Mean :0.7294   
## 3rd Qu.:1.0000 3rd Qu.:1.60 3rd Qu.:2.000 3rd Qu.:1.0000   
## Max. :1.0000 Max. :6.20 Max. :2.000 Max. :4.0000   
## thal target   
## Min. :0.000 Min. :0.0000   
## 1st Qu.:2.000 1st Qu.:0.0000   
## Median :2.000 Median :1.0000   
## Mean :2.314 Mean :0.5446   
## 3rd Qu.:3.000 3rd Qu.:1.0000   
## Max. :3.000 Max. :1.0000

View(data)  
str(data)

## 'data.frame': 303 obs. of 14 variables:  
## $ ï..age : int 63 37 41 56 57 57 56 44 52 57 ...  
## $ sex : int 1 1 0 1 0 1 0 1 1 1 ...  
## $ cp : int 3 2 1 1 0 0 1 1 2 2 ...  
## $ trestbps: int 145 130 130 120 120 140 140 120 172 150 ...  
## $ chol : int 233 250 204 236 354 192 294 263 199 168 ...  
## $ fbs : int 1 0 0 0 0 0 0 0 1 0 ...  
## $ restecg : int 0 1 0 1 1 1 0 1 1 1 ...  
## $ thalach : int 150 187 172 178 163 148 153 173 162 174 ...  
## $ exang : int 0 0 0 0 1 0 0 0 0 0 ...  
## $ oldpeak : num 2.3 3.5 1.4 0.8 0.6 0.4 1.3 0 0.5 1.6 ...  
## $ slope : int 0 0 2 2 2 1 1 2 2 2 ...  
## $ ca : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ thal : int 1 2 2 2 2 1 2 3 3 2 ...  
## $ target : int 1 1 1 1 1 1 1 1 1 1 ...

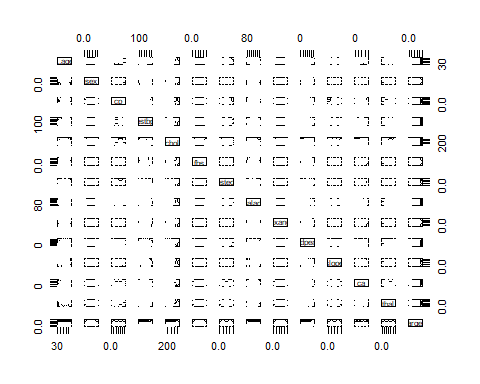
Checking for missing values, if any, for the given dataset

data %>%   
 is.na() %>%   
 colSums(is.na(data))

## ï..age sex cp trestbps chol fbs restecg thalach   
## 0 0 0 0 0 0 0 0   
## exang oldpeak slope ca thal target   
## 0 0 0 0 0 0

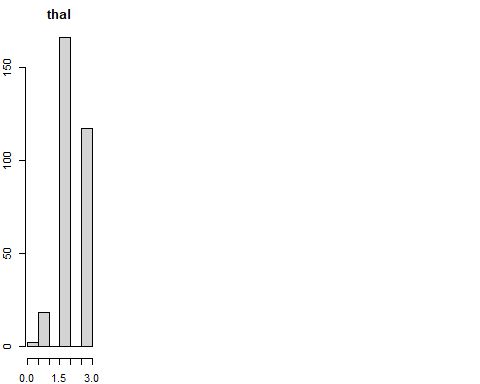
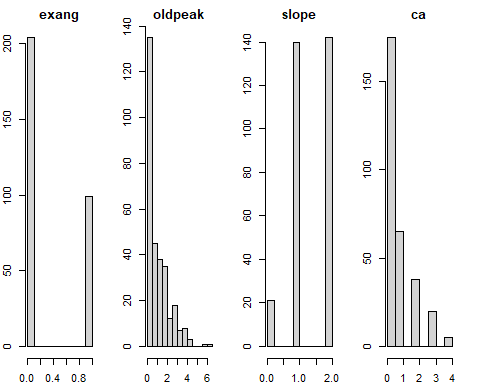
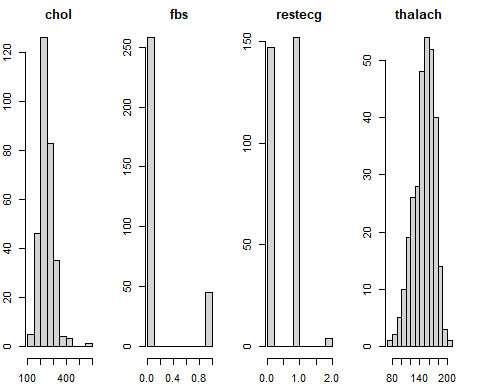
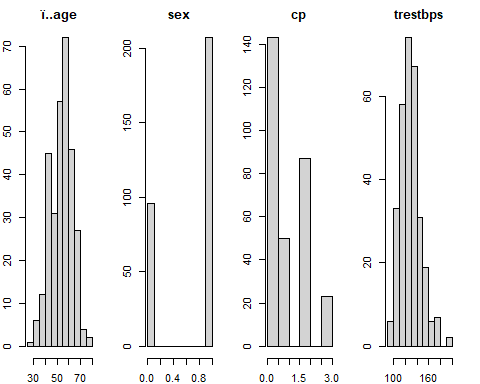
Plot with pairs fn

pairs(data,col=data$target)



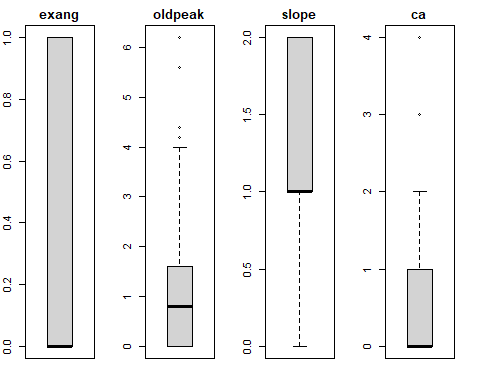
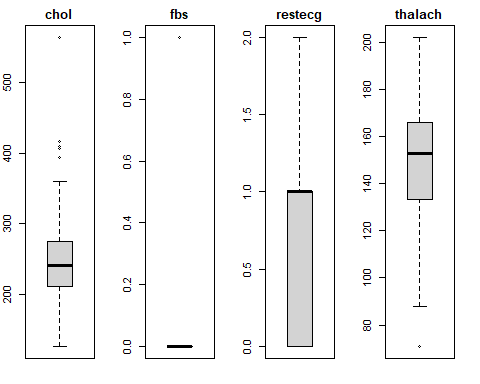
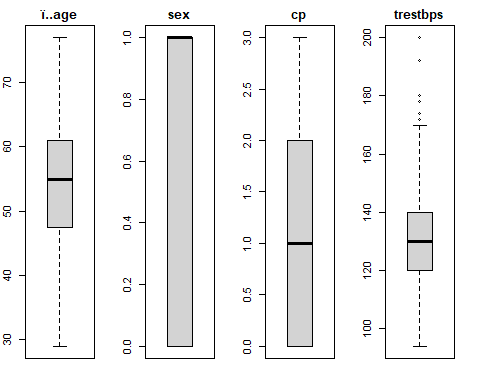
Data analysis: Histogram plot

par(mar = rep(2, 4)) #To reduce the size of the margins  
par(mfrow=c(1,4))  
for(i in 1:13) {  
 hist(data[,i], main=names(data)[i])  
}

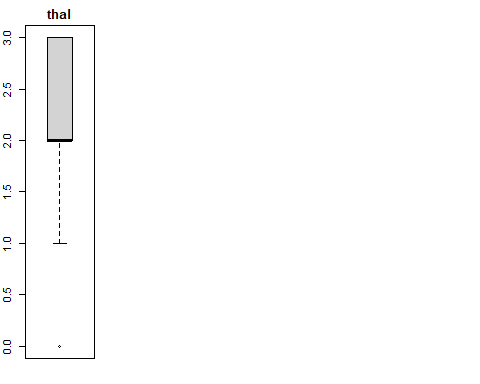


Data analysis: Box plot

par(mar = rep(2, 4)) #To reduce the size of the margins  
par(mfrow=c(1,4))  
for(i in 1:13) {  
 boxplot(data[,i], main=names(data)[i])  
}

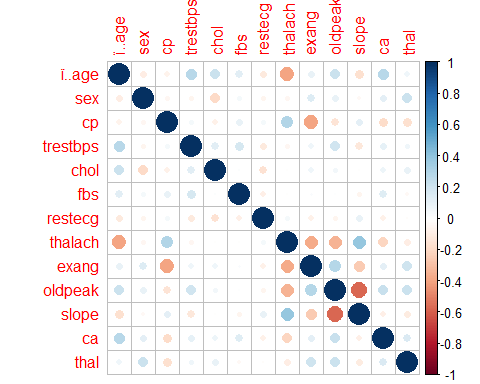


# cor(data[,-14])



To find correlation between every variable

library(corrplot)  
correlations <- cor(data[,1:13])  
corrplot(correlations, method="circle")



attach(data)

Logistic Regression glm = Generalised linear models

# baseline model  
table(target) # do not need data$target because we have used attach(data)

## target  
## 0 1   
## 138 165

165/303

## [1] 0.5445545

Splitting the dataset into train and test

library(caTools)  
#randomly split data  
set.seed(123)  
split=sample.split(target, SplitRatio = 0.75)  
# split

qualityTrain=subset(data,split == TRUE)  
qualityTest=subset(data,split == FALSE)  
nrow(qualityTrain)

## [1] 228

nrow(qualityTest)

## [1] 75

Logistic regression model

datasetlog=glm(target ~ .,data=qualityTrain,family = binomial)  
summary(datasetlog)

##   
## Call:  
## glm(formula = target ~ ., family = binomial, data = qualityTrain)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5396 -0.3303 0.1122 0.5589 2.8234   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 4.927799 3.144534 1.567 0.117091   
## ï..age -0.029667 0.028221 -1.051 0.293154   
## sex -2.232320 0.578096 -3.862 0.000113 \*\*\*  
## cp 0.926967 0.227764 4.070 4.7e-05 \*\*\*  
## trestbps -0.025049 0.013133 -1.907 0.056484 .   
## chol -0.007444 0.004485 -1.660 0.096933 .   
## fbs -0.389890 0.613586 -0.635 0.525150   
## restecg 0.085075 0.422699 0.201 0.840491   
## thalach 0.031284 0.013307 2.351 0.018726 \*   
## exang -0.691772 0.510946 -1.354 0.175767   
## oldpeak -0.478801 0.269531 -1.776 0.075663 .   
## slope 0.523543 0.440047 1.190 0.234147   
## ca -0.689768 0.220906 -3.122 0.001794 \*\*   
## thal -0.671945 0.346003 -1.942 0.052134 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 314.32 on 227 degrees of freedom  
## Residual deviance: 150.27 on 214 degrees of freedom  
## AIC: 178.27  
##   
## Number of Fisher Scoring iterations: 6

Removing variables based on Significance Level using the backward method i.e. removing the least significant variables one by one. In this case, from the above significant codes we see that the least significant variables are ‘ï..age’, ‘fbs’, ‘restecg’, ‘exang’, and ‘slope’

datasetlog2=glm(target ~ sex+cp+trestbps+chol+fbs+restecg+thalach+exang+oldpeak+slope+ca+thal,data=qualityTrain,family = binomial)  
summary(datasetlog2)

##   
## Call:  
## glm(formula = target ~ sex + cp + trestbps + chol + fbs + restecg +   
## thalach + exang + oldpeak + slope + ca + thal, family = binomial,   
## data = qualityTrain)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.4960 -0.3521 0.1221 0.5700 2.7963   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 3.198502 2.656704 1.204 0.228614   
## sex -2.180515 0.573779 -3.800 0.000145 \*\*\*  
## cp 0.919071 0.227208 4.045 5.23e-05 \*\*\*  
## trestbps -0.027641 0.013056 -2.117 0.034253 \*   
## chol -0.008492 0.004424 -1.919 0.054927 .   
## fbs -0.450683 0.613252 -0.735 0.462396   
## restecg 0.133113 0.418634 0.318 0.750508   
## thalach 0.035627 0.012664 2.813 0.004905 \*\*   
## exang -0.692668 0.509519 -1.359 0.174002   
## oldpeak -0.455187 0.267860 -1.699 0.089253 .   
## slope 0.539636 0.439552 1.228 0.219562   
## ca -0.717876 0.215169 -3.336 0.000849 \*\*\*  
## thal -0.682695 0.346326 -1.971 0.048695 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 314.32 on 227 degrees of freedom  
## Residual deviance: 151.38 on 215 degrees of freedom  
## AIC: 177.38  
##   
## Number of Fisher Scoring iterations: 6

Removing variables based on Significance Level using the backward method i.e. removing the least significant variables one by one. In this case, from the above significant codes we see that the least significant variables are ‘ï..age’, ‘fbs’, ‘restecg’, ‘exang’, and ‘slope’

datasetlog3=glm(target ~ sex+cp+trestbps+chol+restecg+thalach+exang+oldpeak+slope+ca+thal,data=qualityTrain,family = binomial)  
summary(datasetlog3)

##   
## Call:  
## glm(formula = target ~ sex + cp + trestbps + chol + restecg +   
## thalach + exang + oldpeak + slope + ca + thal, family = binomial,   
## data = qualityTrain)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.4794 -0.3462 0.1272 0.5784 2.7990   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 3.257919 2.653248 1.228 0.219485   
## sex -2.204628 0.572058 -3.854 0.000116 \*\*\*  
## cp 0.894491 0.221970 4.030 5.58e-05 \*\*\*  
## trestbps -0.028303 0.012963 -2.183 0.029014 \*   
## chol -0.008661 0.004436 -1.953 0.050873 .   
## restecg 0.141881 0.417357 0.340 0.733894   
## thalach 0.034794 0.012520 2.779 0.005449 \*\*   
## exang -0.736888 0.503031 -1.465 0.142950   
## oldpeak -0.462083 0.267569 -1.727 0.084174 .   
## slope 0.574260 0.435518 1.319 0.187314   
## ca -0.739510 0.212744 -3.476 0.000509 \*\*\*  
## thal -0.616816 0.334793 -1.842 0.065419 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 314.32 on 227 degrees of freedom  
## Residual deviance: 151.91 on 216 degrees of freedom  
## AIC: 175.91  
##   
## Number of Fisher Scoring iterations: 6

Removing variables based on Significance Level using the backward method i.e. removing the least significant variables one by one. In this case, from the above significant codes we see that the least significant variables are ‘ï..age’, ‘fbs’, ‘restecg’, ‘exang’, and ‘slope’

datasetlog4=glm(target ~ sex+cp+trestbps+chol+thalach+exang+oldpeak+slope+ca+thal,data=qualityTrain,family = binomial)  
summary(datasetlog4)

##   
## Call:  
## glm(formula = target ~ sex + cp + trestbps + chol + thalach +   
## exang + oldpeak + slope + ca + thal, family = binomial, data = qualityTrain)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.4519 -0.3601 0.1280 0.5745 2.8315   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 3.381470 2.628384 1.287 0.198261   
## sex -2.221353 0.570374 -3.895 9.84e-05 \*\*\*  
## cp 0.886272 0.220099 4.027 5.66e-05 \*\*\*  
## trestbps -0.028277 0.012941 -2.185 0.028885 \*   
## chol -0.009059 0.004273 -2.120 0.034001 \*   
## thalach 0.034871 0.012515 2.786 0.005330 \*\*   
## exang -0.755209 0.500013 -1.510 0.130947   
## oldpeak -0.455166 0.265982 -1.711 0.087032 .   
## slope 0.594020 0.430875 1.379 0.168006   
## ca -0.741934 0.212321 -3.494 0.000475 \*\*\*  
## thal -0.605269 0.331742 -1.825 0.068074 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 314.32 on 227 degrees of freedom  
## Residual deviance: 152.03 on 217 degrees of freedom  
## AIC: 174.03  
##   
## Number of Fisher Scoring iterations: 6

Removing variables based on Significance Level using the backward method i.e. removing the least significant variables one by one. In this case, from the above significant codes we see that the least significant variables are ‘ï..age’, ‘fbs’, ‘restecg’, ‘exang’, and ‘slope’

datasetlog5=glm(target ~ sex+cp+trestbps+chol+thalach+oldpeak+slope+ca+thal,data=qualityTrain,family = binomial)  
summary(datasetlog5)

##   
## Call:  
## glm(formula = target ~ sex + cp + trestbps + chol + thalach +   
## oldpeak + slope + ca + thal, family = binomial, data = qualityTrain)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.4366 -0.3581 0.1220 0.5927 2.7349   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.728046 2.520595 1.082 0.279118   
## sex -2.221351 0.557240 -3.986 6.71e-05 \*\*\*  
## cp 0.990788 0.215473 4.598 4.26e-06 \*\*\*  
## trestbps -0.030222 0.012780 -2.365 0.018043 \*   
## chol -0.008606 0.004161 -2.068 0.038604 \*   
## thalach 0.038719 0.012333 3.139 0.001693 \*\*   
## oldpeak -0.486890 0.261975 -1.859 0.063093 .   
## slope 0.594077 0.427188 1.391 0.164326   
## ca -0.716905 0.211387 -3.391 0.000695 \*\*\*  
## thal -0.645267 0.333063 -1.937 0.052700 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 314.32 on 227 degrees of freedom  
## Residual deviance: 154.27 on 218 degrees of freedom  
## AIC: 174.27  
##   
## Number of Fisher Scoring iterations: 6

Removing variables based on Significance Level using the backward method i.e. removing the least significant variables one by one. In this case, from the above significant codes we see that the least significant variables are ‘ï..age’, ‘fbs’, ‘restecg’, ‘exang’, and ‘slope’

datasetlog6=glm(target ~ sex+cp+trestbps+chol+thalach+oldpeak+ca+thal,data=qualityTrain,family = binomial)  
summary(datasetlog6)

##   
## Call:  
## glm(formula = target ~ sex + cp + trestbps + chol + thalach +   
## oldpeak + ca + thal, family = binomial, data = qualityTrain)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.3608 -0.3908 0.1185 0.5595 2.6328   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.955652 2.488644 1.188 0.234969   
## sex -2.144325 0.543428 -3.946 7.95e-05 \*\*\*  
## cp 0.996208 0.214774 4.638 3.51e-06 \*\*\*  
## trestbps -0.029643 0.012747 -2.325 0.020049 \*   
## chol -0.008259 0.004100 -2.015 0.043953 \*   
## thalach 0.042138 0.011933 3.531 0.000414 \*\*\*  
## oldpeak -0.669039 0.228538 -2.927 0.003417 \*\*   
## ca -0.654016 0.201596 -3.244 0.001178 \*\*   
## thal -0.638586 0.336643 -1.897 0.057838 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 314.32 on 227 degrees of freedom  
## Residual deviance: 156.17 on 219 degrees of freedom  
## AIC: 174.17  
##   
## Number of Fisher Scoring iterations: 6

Applying Model after removing least significant variables. A general rule in machine learning is that the more features you have, the more likely your model will suffer from overfitting. Making predictions on training sets using datasetlog6

predictTrain=predict(datasetlog6,type="response")  
# predictTrain

Accuracy

#Accuracy using a threshold of 0.7  
predictTest=predict(datasetlog6, newdata = qualityTest,type = "response")  
table(qualityTest$target,predictTest >=0.7)

##   
## FALSE TRUE  
## 0 28 6  
## 1 12 29

#accuracy  
(28+29)/75

## [1] 0.76

Logistic regression model with all the variables and logistic regression model after removing less significant attributes performed best with an accuracy of testing 76%

We can use the confint function to obtain confidence intervals for the coefficient estimates.

## CIs using profiled log-likelihood  
confint(datasetlog6)

## 2.5 % 97.5 %  
## (Intercept) -1.82970008 7.9833987313  
## sex -3.27802474 -1.1331027240  
## cp 0.59228777 1.4394981289  
## trestbps -0.05564081 -0.0053281873  
## chol -0.01647006 -0.0001178326  
## thalach 0.02004360 0.0670369080  
## oldpeak -1.14412760 -0.2422861843  
## ca -1.06303396 -0.2657695216  
## thal -1.31492295 0.0136751579

## CIs using standard errors  
confint.default(datasetlog6)

## 2.5 % 97.5 %  
## (Intercept) -1.92200131 7.8333057487  
## sex -3.20942471 -1.0792257059  
## cp 0.57525944 1.4171575002  
## trestbps -0.05462633 -0.0046586886  
## chol -0.01629420 -0.0002237449  
## thalach 0.01874979 0.0655252127  
## oldpeak -1.11696530 -0.2211121288  
## ca -1.04913766 -0.2588946055  
## thal -1.29839445 0.0212221189

We may also wish to see measures of how well our model fits. This can be particularly useful when comparing competing models. The output produced by summary(datasetlog6) included indices of fit (shown below the coefficients), including the null and deviance residuals and the AIC. One measure of model fit is the significance of the overall model. This test asks whether the model with predictors fits significantly better than a model with just an intercept (i.e., a null model). The test statistic is the difference between the residual deviance for the model with predictors and the null model. The test statistic is distributed chi-squared with degrees of freedom equal to the differences in degrees of freedom between the current and the null model (i.e., the number of predictor variables in the model). To find the difference in deviance for the two models (i.e., the test statistic) we can use the command:

with(datasetlog6, null.deviance - deviance)

## [1] 158.1511

The degrees of freedom for the difference between the two models is equal to the number of predictor variables in the mode, and can be obtained using:

with(datasetlog6, df.null - df.residual)

## [1] 8

Finally, the p-value can be obtained using:

with(datasetlog6, pchisq(null.deviance - deviance, df.null - df.residual, lower.tail = FALSE))

## [1] 3.89473e-30

The chi-square of 158.1511 with 8 degrees of freedom and an associated p-value of less than 0.001 tells us that our model as a whole fits significantly better than an empty model. This is sometimes called a likelihood ratio test (the deviance residual is -2\*log likelihood). To see the model’s log likelihood, we type:

logLik(datasetlog6)

## 'log Lik.' -78.08368 (df=9)

**Conclusion:**  
Logistic Regression has been successfully performed on Smarket, Admission Predict, binary\_Admit and Heart dataset.