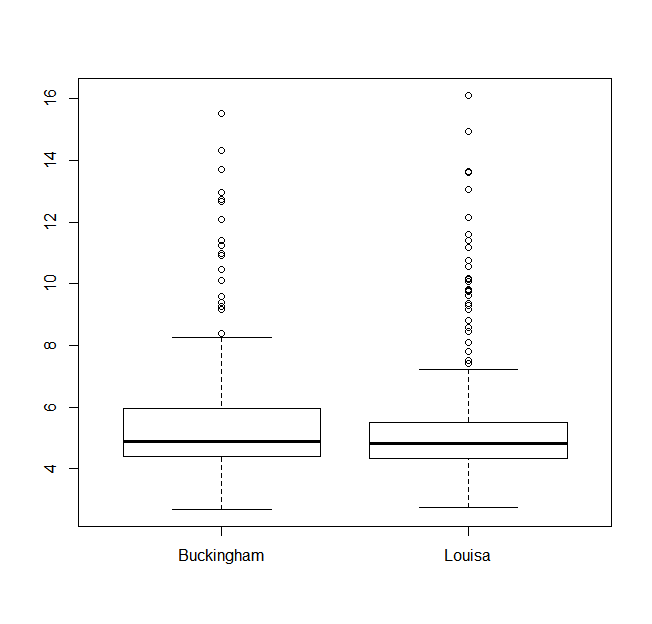
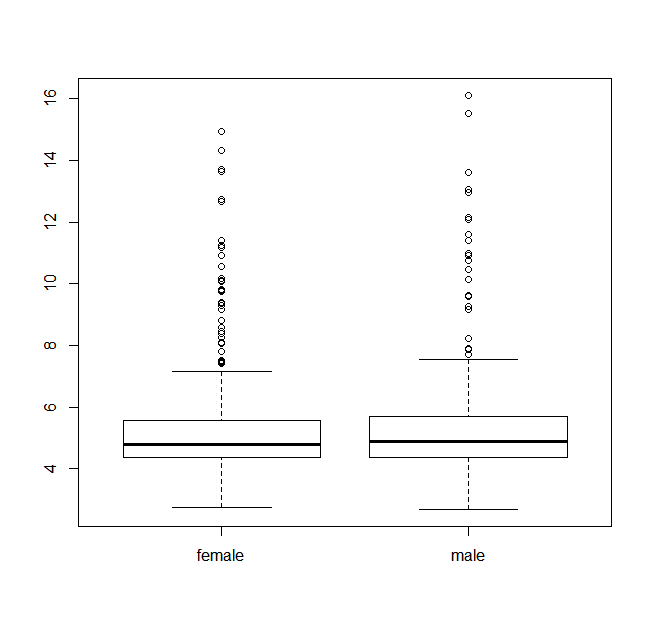
### Appendix for Final Draft for the Project.

### R codes

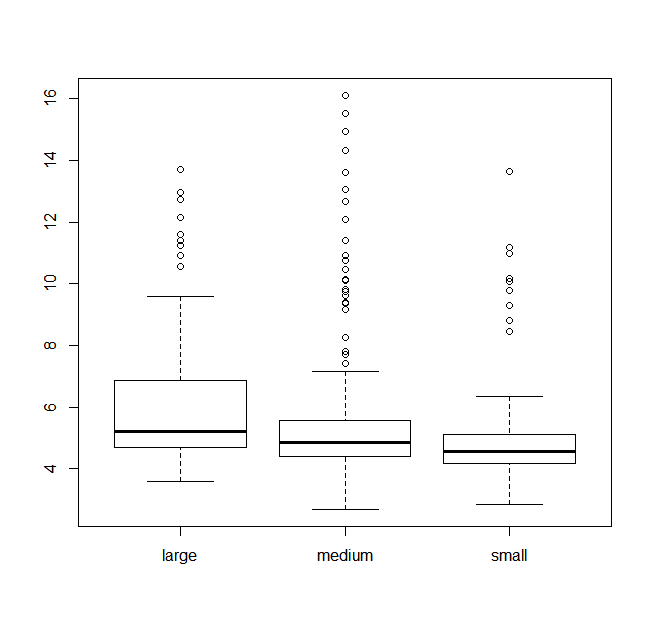
file <- "C:/Users/Administrator/Desktop/diabetes2.csv"  
data <- read.csv(file)  
data$diabetes <- ifelse(data$glyhb > 7, 1, 0)  
  
# bivariate analysis for glyhb  
# use `pairwise.complete.obs` to remove NAs  
  
#boxplots  
boxplot(glyhb ~ location, data = data)



boxplot(glyhb ~ gender, data = data)



subdata <- data[which(data$frame !=""),]  
subdata$frame <- factor(subdata$frame, levels =c("large","medium","small"))  
boxplot(glyhb ~ frame, data = subdata)



res <- tapply(data[ ,"glyhb"], data[ ,"location"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## Buckingham 5.676684 2.224481  
## Louisa 5.507200 2.262143

Buckingham <- data[ ,"glyhb"][which(data[ ,"location"] == "Buckingham")]  
Louisa <- data[ ,"glyhb"][which(data[ ,"location"] == "Louisa")]  
var.test(Buckingham, Louisa)

##   
## F test to compare two variances  
##   
## data: Buckingham and Louisa  
## F = 0.967, num df = 189, denom df = 199, p-value = 0.8164  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.7294856 1.2831270  
## sample estimates:  
## ratio of variances   
## 0.9669794

t.test(Buckingham, Louisa, var.equal = TRUE, alternative = "two.sided", conf.level = 0.95)

##   
## Two Sample t-test  
##   
## data: Buckingham and Louisa  
## t = 0.7456, df = 388, p-value = 0.4564  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -0.2774507 0.6164191  
## sample estimates:  
## mean of x mean of y   
## 5.676684 5.507200

res <- tapply(data[ ,"glyhb"], data[ ,"gender"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## female 5.494342 2.133675  
## male 5.724074 2.387779

female <- data[ ,"glyhb"][which(data[ ,"gender"] == "female")]  
male<- data[ ,"glyhb"][which(data[ ,"gender"] == "male")]  
var.test(female, male)

##   
## F test to compare two variances  
##   
## data: female and male  
## F = 0.7985, num df = 227, denom df = 161, p-value = 0.119  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.5974696 1.0596196  
## sample estimates:  
## ratio of variances   
## 0.7984873

t.test(female, male, var.equal = TRUE, alternative = "two.sided", conf.level = 0.95)

##   
## Two Sample t-test  
##   
## data: female and male  
## t = -0.9969, df = 388, p-value = 0.3194  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -0.6828033 0.2233393  
## sample estimates:  
## mean of x mean of y   
## 5.494342 5.724074

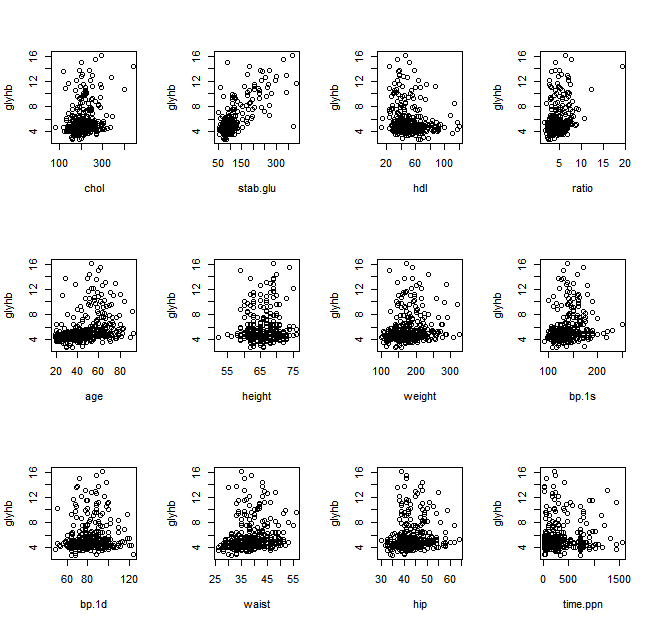
res <- tapply(data[ ,"glyhb"], data[ ,"frame"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
#romove NAs  
df[-1, ]

## mean sd  
## large 6.105657 2.245535  
## medium 5.640281 2.438113  
## small 5.040882 1.802382

subdata <- data[data$frame != "", ]  
anova(lm(glyhb ~ frame, data = subdata))

## Analysis of Variance Table  
##   
## Response: glyhb  
## Df Sum Sq Mean Sq F value Pr(>F)   
## frame 2 57.49 28.7442 5.7659 0.003416 \*\*  
## Residuals 376 1874.42 4.9852   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

#scatter plots   
par(mfrow = c(3,4))  
vars <- c("chol","stab.glu","hdl","ratio","age","height","weight","bp.1s","bp.1d","waist", "hip","time.ppn")  
for(i in vars)  
 plot(data[ ,i], data[ ,"glyhb"], xlab = i, ylab = "glyhb")



cor.chol <- cor(data[ ,"chol"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.chol <- round(cor.chol, 3)  
cor.chol

## [1] 0.247

cor.test(data[ ,"chol"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "chol"] and data[, "glyhb"]  
## t = 5.0166, df = 387, p-value = 8.032e-07  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.1513882 0.3382189  
## sample estimates:  
## cor   
## 0.2470987

cor.stab.glu <- cor(data[ ,"stab.glu"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.stab.glu <- round(cor.stab.glu, 3)  
cor.stab.glu

## [1] 0.749

cor.test(data[ ,"stab.glu"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "stab.glu"] and data[, "glyhb"]  
## t = 22.2832, df = 388, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.7021757 0.7897777  
## sample estimates:  
## cor   
## 0.7492355

cor.hdl <- cor(data[ ,"hdl"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.hdl <- round(cor.hdl, 3)  
cor.hdl

## [1] -0.149

cor.test(data[ ,"hdl"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "hdl"] and data[, "glyhb"]  
## t = -2.9672, df = 387, p-value = 0.003192  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.2449421 -0.0504630  
## sample estimates:  
## cor   
## -0.1491446

cor.ratio <- cor(data[ ,"ratio"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.ratio <- round(cor.ratio, 3)  
cor.ratio

## [1] 0.329

cor.test(data[ ,"ratio"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "ratio"] and data[, "glyhb"]  
## t = 6.8459, df = 387, p-value = 2.992e-11  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.2369789 0.4145475  
## sample estimates:  
## cor   
## 0.3286646

cor.age <- cor(data[ ,"age"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.age <- round(cor.age, 3)  
cor.age

## [1] 0.339

cor.test(data[ ,"age"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "age"] and data[, "glyhb"]  
## t = 7.0961, df = 388, p-value = 6.138e-12  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.2479728 0.4239623  
## sample estimates:  
## cor   
## 0.3389292

cor.height <- cor(data[ ,"height"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.height <- round(cor.height, 3)  
cor.height

## [1] 0.063

cor.test(data[ ,"height"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "height"] and data[, "glyhb"]  
## t = 1.2358, df = 383, p-value = 0.2173  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.03715689 0.16194843  
## sample estimates:  
## cor   
## 0.06302284

cor.weight <- cor(data[ ,"weight"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.weight <- round(cor.weight, 3)  
cor.weight

## [1] 0.172

cor.test(data[ ,"weight"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "weight"] and data[, "glyhb"]  
## t = 3.4324, df = 387, p-value = 0.0006628  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.07371196 0.26675317  
## sample estimates:  
## cor   
## 0.1718821

cor.bp.1s <- cor(data[ ,"bp.1s"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.bp.1s <- round(cor.bp.1s, 3)  
cor.bp.1s

## [1] 0.198

cor.test(data[ ,"bp.1s"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "bp.1s"] and data[, "glyhb"]  
## t = 3.9519, df = 383, p-value = 9.233e-05  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.09996788 0.29210297  
## sample estimates:  
## cor   
## 0.1979359

cor.bp.1d <- cor(data[ ,"bp.1d"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.bp.1d <- round(cor.bp.1d, 3)  
cor.bp.1d

## [1] 0.032

cor.test(data[ ,"bp.1d"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "bp.1d"] and data[, "glyhb"]  
## t = 0.6339, df = 383, p-value = 0.5265  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.06779048 0.13189346  
## sample estimates:  
## cor   
## 0.03237455

cor.waist <- cor(data[ ,"waist"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.waist <- round(cor.waist, 3)  
cor.waist

## [1] 0.226

cor.test(data[ ,"waist"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "waist"] and data[, "glyhb"]  
## t = 4.562, df = 386, p-value = 6.815e-06  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.1295430 0.3185685  
## sample estimates:  
## cor   
## 0.226184

cor.hip <- cor(data[ ,"hip"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.hip <- round(cor.hip, 3)  
cor.hip

## [1] 0.141

cor.test(data[ ,"hip"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "hip"] and data[, "glyhb"]  
## t = 2.8063, df = 386, p-value = 0.005266  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.04244043 0.23761415  
## sample estimates:  
## cor   
## 0.1414011

cor.time.ppn <- cor(data[ ,"time.ppn"], data[ ,"glyhb"], use = "pairwise.complete.obs")  
cor.time.ppn <- round(cor.time.ppn , 3)  
cor.time.ppn

## [1] 0.033

cor.test(data[ ,"time.ppn"], data[ ,"glyhb"])

##   
## Pearson's product-moment correlation  
##   
## data: data[, "time.ppn"] and data[, "glyhb"]  
## t = 0.6385, df = 385, p-value = 0.5235  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.06738282 0.13178200  
## sample estimates:  
## cor   
## 0.03252244

# bivariate analysis for diabetes  
# use `pairwise.complete.obs` to remove NAs  
  
tab1 <- table(data[ ,"location"], data[ ,"diabetes"])  
tab1

##   
## 0 1  
## Buckingham 159 31  
## Louisa 171 29

chisq.test(tab1)

##   
## Pearson's Chi-squared test with Yates' continuity correction  
##   
## data: tab1  
## X-squared = 0.127, df = 1, p-value = 0.7216

tab2 <- table(data[ ,"gender"], data[ ,"diabetes"])  
tab2

##   
## 0 1  
## female 194 34  
## male 136 26

chisq.test(tab2)

##   
## Pearson's Chi-squared test with Yates' continuity correction  
##   
## data: tab2  
## X-squared = 0.027, df = 1, p-value = 0.8695

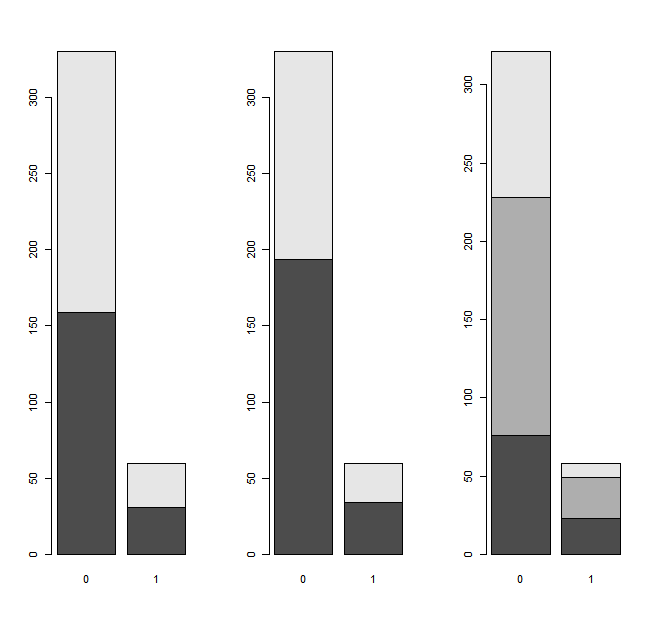
#remove NAs  
tab3 <- table(data[ ,"frame"], data[ ,"diabetes"])[-1, ]  
tab3

##   
## 0 1  
## large 76 23  
## medium 152 26  
## small 93 9

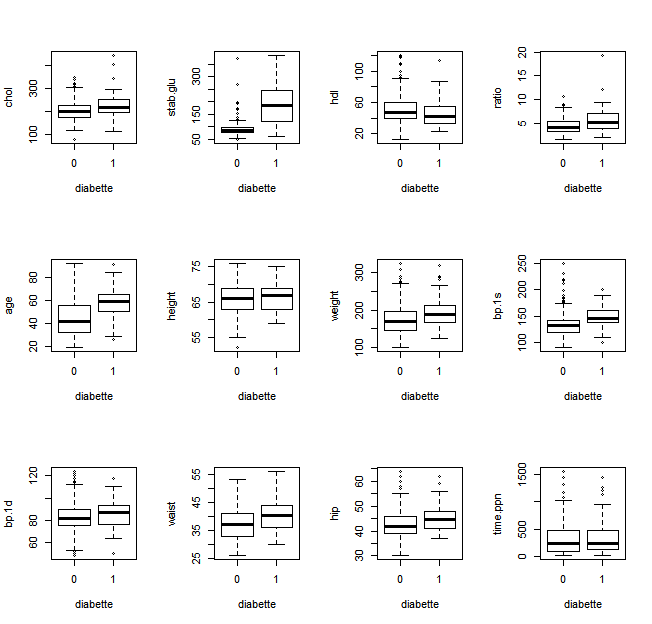
chisq.test(tab3)

##   
## Pearson's Chi-squared test  
##   
## data: tab3  
## X-squared = 8.1728, df = 2, p-value = 0.0168

#barplots  
par(mfrow = c(1,3))  
barplot(tab1)  
barplot(tab2)  
barplot(tab3)



#boxplots  
par(mfrow = c(3,4))  
vars <- c("chol","stab.glu","hdl","ratio","age","height","weight","bp.1s","bp.1d","waist","hip","time.ppn")  
for(i in vars)  
 plot(factor(data[ ,"diabetes"]), data[ ,i], ylab = i, xlab = "diabette")



res <- tapply(data[ ,"chol"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 203.386 41.13504  
## 1 228.600 56.52514

var.test(chol ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: chol by diabetes  
## F = 0.5296, num df = 328, denom df = 59, p-value = 0.000555  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.3471553 0.7649288  
## sample estimates:  
## ratio of variances   
## 0.5295909

t.test(chol ~ diabetes, data = data, var.equal = FALSE, alternative = "two.sided")

##   
## Welch Two Sample t-test  
##   
## data: chol by diabetes  
## t = -3.2996, df = 70.828, p-value = 0.001518  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -40.451620 -9.976344  
## sample estimates:  
## mean in group 0 mean in group 1   
## 203.386 228.600

res <- tapply(data[ ,"stab.glu"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 91.55152 26.86585  
## 1 194.16667 77.43826

var.test(stab.glu ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: stab.glu by diabetes  
## F = 0.1204, num df = 329, denom df = 59, p-value < 2.2e-16  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.07890509 0.17382934  
## sample estimates:  
## ratio of variances   
## 0.1203622

t.test(stab.glu ~ diabetes, data = data, var.equal = FALSE, alternative = "two.sided")

##   
## Welch Two Sample t-test  
##   
## data: stab.glu by diabetes  
## t = -10.1538, df = 61.605, p-value = 8.863e-15  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -122.8194 -82.4109  
## sample estimates:  
## mean in group 0 mean in group 1   
## 91.55152 194.16667

res <- tapply(data[ ,"hdl"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 51.17629 17.25188  
## 1 45.28333 16.85340

var.test(hdl ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: hdl by diabetes  
## F = 1.0478, num df = 328, denom df = 59, p-value = 0.8527  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.6868805 1.5134860  
## sample estimates:  
## ratio of variances   
## 1.047847

t.test(hdl ~ diabetes, data = data, var.equal = FALSE, alternative = "two.sided")

##   
## Welch Two Sample t-test  
##   
## data: hdl by diabetes  
## t = 2.4817, df = 83.158, p-value = 0.01509  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## 1.170163 10.615754  
## sample estimates:  
## mean in group 0 mean in group 1   
## 51.17629 45.28333

res <- tapply(data[ ,"ratio"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 4.324316 1.435488  
## 1 5.635000 2.634827

var.test(ratio ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: ratio by diabetes  
## F = 0.2968, num df = 328, denom df = 59, p-value = 3.985e-12  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.1945711 0.4287218  
## sample estimates:  
## ratio of variances   
## 0.2968213

t.test(ratio ~ diabetes, data = data, var.equal = FALSE, alternative = "two.sided")

##   
## Welch Two Sample t-test  
##   
## data: ratio by diabetes  
## t = -3.753, df = 65.526, p-value = 0.000373  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -2.0080581 -0.6133097  
## sample estimates:  
## mean in group 0 mean in group 1   
## 4.324316 5.635000

res <- tapply(data[ ,"age"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 44.66061 16.10584  
## 1 58.40000 13.12160

var.test(age ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: age by diabetes  
## F = 1.5066, num df = 329, denom df = 59, p-value = 0.05693  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.9876624 2.1758382  
## sample estimates:  
## ratio of variances   
## 1.506585

t.test(age ~ diabetes, data = data, var.equal = TRUE, alternative = "two.sided")

##   
## Two Sample t-test  
##   
## data: age by diabetes  
## t = -6.24, df = 388, p-value = 1.147e-09  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -18.068431 -9.410357  
## sample estimates:  
## mean in group 0 mean in group 1   
## 44.66061 58.40000

res <- tapply(data[ ,"height"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 65.95092 3.943140  
## 1 66.16949 3.855818

var.test(height ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: height by diabetes  
## F = 1.0458, num df = 325, denom df = 58, p-value = 0.8621  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.6829914 1.5145957  
## sample estimates:  
## ratio of variances   
## 1.045807

t.test(height ~ diabetes, data = data, var.equal = TRUE, alternative = "two.sided")

##   
## Two Sample t-test  
##   
## data: height by diabetes  
## t = -0.3931, df = 383, p-value = 0.6945  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -1.3118107 0.8746681  
## sample estimates:  
## mean in group 0 mean in group 1   
## 65.95092 66.16949

res <- tapply(data[ ,"weight"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 174.6030 39.84213  
## 1 192.7119 40.67917

var.test(weight ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: weight by diabetes  
## F = 0.9593, num df = 329, denom df = 58, p-value = 0.7994  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.6266595 1.3886597  
## sample estimates:  
## ratio of variances   
## 0.95927

t.test(weight ~ diabetes, data = data, var.equal = TRUE, alternative = "two.sided")

##   
## Two Sample t-test  
##   
## data: weight by diabetes  
## t = -3.2054, df = 387, p-value = 0.001461  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -29.216433 -7.001235  
## sample estimates:  
## mean in group 0 mean in group 1   
## 174.6030 192.7119

res <- tapply(data[ ,"bp.1s"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 135.1877 22.92578  
## 1 147.7667 20.49586

var.test(bp.1s ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: bp.1s by diabetes  
## F = 1.2512, num df = 324, denom df = 59, p-value = 0.2972  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.8199177 1.8079606  
## sample estimates:  
## ratio of variances   
## 1.251169

t.test(bp.1s ~ diabetes, data = data, var.equal = TRUE, alternative = "two.sided")

##   
## Two Sample t-test  
##   
## data: bp.1s by diabetes  
## t = -3.9667, df = 383, p-value = 8.699e-05  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -18.814005 -6.343944  
## sample estimates:  
## mean in group 0 mean in group 1   
## 135.1877 147.7667

res <- tapply(data[ ,"bp.1d"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 82.99692 13.70174  
## 1 84.85000 12.91409

var.test(bp.1d ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: bp.1d by diabetes  
## F = 1.1257, num df = 324, denom df = 59, p-value = 0.5918  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.7376972 1.6266601  
## sample estimates:  
## ratio of variances   
## 1.125703

t.test(bp.1d ~ diabetes, data = data, var.equal = TRUE, alternative = "two.sided")

##   
## Two Sample t-test  
##   
## data: bp.1d by diabetes  
## t = -0.9709, df = 383, p-value = 0.3322  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -5.605775 1.899621  
## sample estimates:  
## mean in group 0 mean in group 1   
## 82.99692 84.85000

res <- tapply(data[ ,"waist"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 37.35061 5.604356  
## 1 40.88333 5.746308

var.test(waist ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: waist by diabetes  
## F = 0.9512, num df = 327, denom df = 59, p-value = 0.7661  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.6234835 1.3740478  
## sample estimates:  
## ratio of variances   
## 0.9512039

t.test(waist ~ diabetes, data = data, var.equal = TRUE, alternative = "two.sided")

##   
## Two Sample t-test  
##   
## data: waist by diabetes  
## t = -4.4718, df = 386, p-value = 1.021e-05  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -5.085958 -1.979489  
## sample estimates:  
## mean in group 0 mean in group 1   
## 37.35061 40.88333

res <- tapply(data[ ,"hip"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 42.69207 5.621578  
## 1 44.90000 5.476297

var.test(hip ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: hip by diabetes  
## F = 1.0538, num df = 327, denom df = 59, p-value = 0.831  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.6907069 1.5221963  
## sample estimates:  
## ratio of variances   
## 1.053762

t.test(hip ~ diabetes, data = data, var.equal = TRUE, alternative = "two.sided")

##   
## Two Sample t-test  
##   
## data: hip by diabetes  
## t = -2.8082, df = 386, p-value = 0.005235  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -3.7537987 -0.6620549  
## sample estimates:  
## mean in group 0 mean in group 1   
## 42.69207 44.90000

res <- tapply(data[ ,"time.ppn"], data[ ,"diabetes"], function(x) {  
 c(mean(x, na.rm = T),sd(x, na.rm = T))   
 })  
df <- do.call('rbind',res)  
colnames(df) <- c("mean","sd")  
df

## mean sd  
## 0 330.1835 302.6452  
## 1 368.5000 341.9464

var.test(time.ppn ~ diabetes, data = data)

##   
## F test to compare two variances  
##   
## data: time.ppn by diabetes  
## F = 0.7833, num df = 326, denom df = 59, p-value = 0.194  
## alternative hypothesis: true ratio of variances is not equal to 1  
## 95 percent confidence interval:  
## 0.5134175 1.1316906  
## sample estimates:  
## ratio of variances   
## 0.7833421

t.test(time.ppn ~ diabetes, data = data, var.equal = TRUE, alternative = "two.sided")

##   
## Two Sample t-test  
##   
## data: time.ppn by diabetes  
## t = -0.8829, df = 385, p-value = 0.3778  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -123.64017 47.00714  
## sample estimates:  
## mean in group 0 mean in group 1   
## 330.1835 368.5000

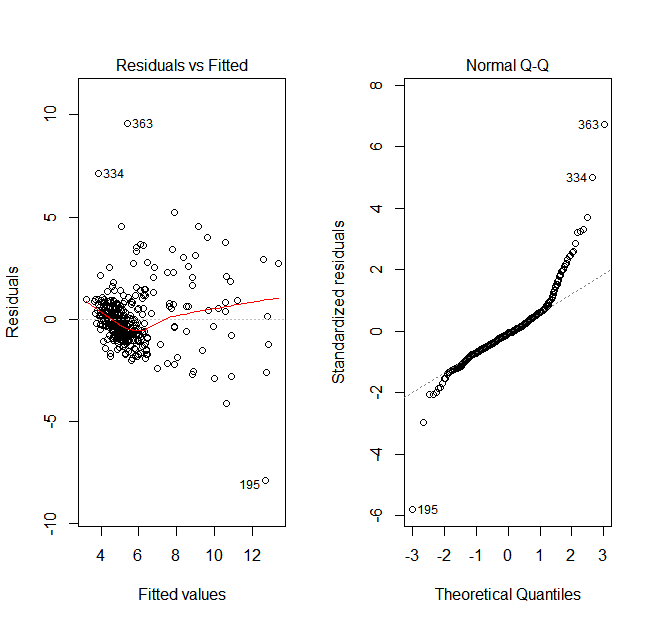
#linear regression  
  
#romove unnecessary variables and NAs  
subdata <- data[ ,-c(1,18)]  
subdata <- subdata[complete.cases(subdata), ]  
lm.fit.full <- lm(glyhb ~ ., data = subdata)  
  
lm.fit.red <- lm(glyhb ~ . - height - bp.1d - time.ppn - location - gender, data = subdata)  
  
anova(lm.fit.red, lm.fit.full)

## Analysis of Variance Table  
##   
## Model 1: glyhb ~ (chol + stab.glu + hdl + ratio + location + age + gender +   
## height + weight + frame + bp.1s + bp.1d + waist + hip + time.ppn) -   
## height - bp.1d - time.ppn - location - gender  
## Model 2: glyhb ~ chol + stab.glu + hdl + ratio + location + age + gender +   
## height + weight + frame + bp.1s + bp.1d + waist + hip + time.ppn  
## Res.Df RSS Df Sum of Sq F Pr(>F)  
## 1 362 746.07   
## 2 357 727.25 5 18.814 1.8471 0.103

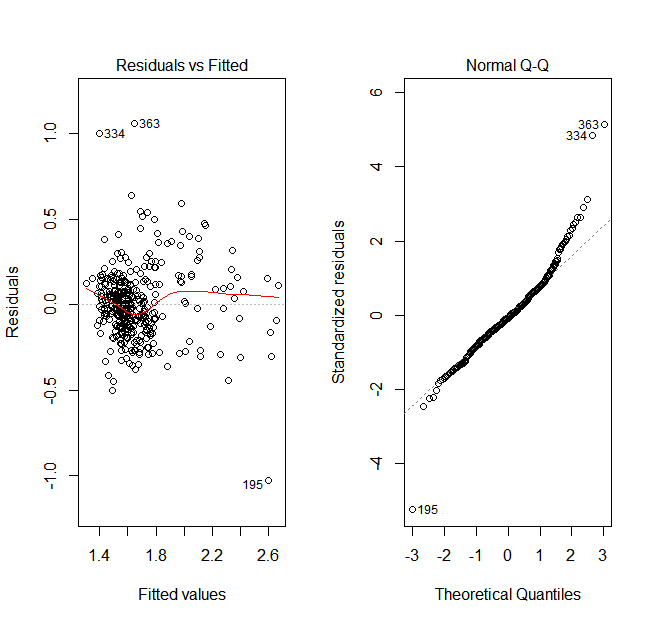
summary(lm.fit.red)

##   
## Call:  
## lm(formula = glyhb ~ . - height - bp.1d - time.ppn - location -   
## gender, data = subdata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7.9010 -0.7122 -0.1245 0.5084 9.5536   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.002147 1.108577 -0.904 0.3666   
## chol 0.003216 0.003232 0.995 0.3203   
## stab.glu 0.027409 0.001505 18.208 <2e-16 \*\*\*  
## hdl 0.002282 0.010024 0.228 0.8201   
## ratio 0.132205 0.111430 1.186 0.2362   
## age 0.012448 0.005726 2.174 0.0304 \*   
## weight -0.003156 0.004203 -0.751 0.4531   
## framelarge 0.483445 0.512941 0.942 0.3466   
## framemedium 0.706728 0.498550 1.418 0.1572   
## framesmall 0.612486 0.512477 1.195 0.2328   
## bp.1s 0.001331 0.003676 0.362 0.7174   
## waist 0.035703 0.030172 1.183 0.2375   
## hip 0.002436 0.027178 0.090 0.9286   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.436 on 362 degrees of freedom  
## Multiple R-squared: 0.5941, Adjusted R-squared: 0.5806   
## F-statistic: 44.15 on 12 and 362 DF, p-value: < 2.2e-16

par(mfrow = c(1,2))  
plot(lm.fit.red, which = c(1,2))



lm.fit.red.log <- lm(log(glyhb) ~ . - height - bp.1d - time.ppn - location - gender, data = subdata)  
  
par(mfrow = c(1,2))  
plot(lm.fit.red.log, which = c(1,2))



summary(lm.fit.red.log)

##   
## Call:  
## lm(formula = log(glyhb) ~ . - height - bp.1d - time.ppn - location -   
## gender, data = subdata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.02954 -0.11588 -0.01587 0.10707 1.05572   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.6567400 0.1600369 4.104 5.03e-05 \*\*\*  
## chol 0.0003777 0.0004666 0.809 0.41882   
## stab.glu 0.0035354 0.0002173 16.269 < 2e-16 \*\*\*  
## hdl 0.0002537 0.0014472 0.175 0.86093   
## ratio 0.0190061 0.0160864 1.182 0.23818   
## age 0.0026961 0.0008267 3.261 0.00121 \*\*   
## weight -0.0004584 0.0006067 -0.756 0.45040   
## framelarge 0.0732339 0.0740494 0.989 0.32333   
## framemedium 0.0930937 0.0719719 1.293 0.19667   
## framesmall 0.0767085 0.0739824 1.037 0.30050   
## bp.1s 0.0002568 0.0005307 0.484 0.62876   
## waist 0.0068159 0.0043557 1.565 0.11850   
## hip 0.0007446 0.0039234 0.190 0.84958   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2072 on 362 degrees of freedom  
## Multiple R-squared: 0.5729, Adjusted R-squared: 0.5587   
## F-statistic: 40.46 on 12 and 362 DF, p-value: < 2.2e-16

#logistic regression  
  
#romove unnecessary variables and NAs  
subdata2 <- data[ ,-c(1,6)]  
subdata2 <- subdata2[complete.cases(subdata2), ]  
  
glm.fit.red <- glm(diabetes~ . - height - bp.1d - time.ppn - location - gender, data = subdata2, family = binomial(link = "logit"))  
  
summary(glm.fit.red)

##   
## Call:  
## glm(formula = diabetes ~ . - height - bp.1d - time.ppn - location -   
## gender, family = binomial(link = "logit"), data = subdata2)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.7022 -0.3524 -0.2188 -0.1278 3.1848   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -12.401456 3.182345 -3.897 9.74e-05 \*\*\*  
## chol 0.012521 0.008325 1.504 0.1326   
## stab.glu 0.034459 0.005218 6.604 3.99e-11 \*\*\*  
## hdl -0.026463 0.027162 -0.974 0.3299   
## ratio -0.154444 0.264802 -0.583 0.5597   
## age 0.031154 0.016961 1.837 0.0662 .   
## weight -0.003069 0.011925 -0.257 0.7969   
## framelarge 0.186822 1.410883 0.132 0.8947   
## framemedium 0.521393 1.370695 0.380 0.7037   
## framesmall 0.651211 1.443999 0.451 0.6520   
## bp.1s 0.009747 0.008966 1.087 0.2770   
## waist 0.081689 0.079562 1.027 0.3045   
## hip -0.006461 0.069490 -0.093 0.9259   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 323.04 on 374 degrees of freedom  
## Residual deviance: 161.97 on 362 degrees of freedom  
## AIC: 187.97  
##   
## Number of Fisher Scoring iterations: 6

preds <- predict(glm.fit.red, type = "response")  
pred\_class <- ifelse(preds > 0.5, 1, 0)  
#confuse matrix  
tab <- table(pred\_class, subdata2[ ,"diabetes"])  
tab

##   
## pred\_class 0 1  
## 0 310 20  
## 1 7 38

accuracy <- sum(diag(tab))/sum(tab)  
accuracy

## [1] 0.928

Support Vector Machine (SVM)

#SVM Applies all variable（Eliminate id、frame、location、gender）

options(digits=3)#

head(data)

datas <- data[,-c(16,17,18,19)]#Eliminate Categorical variable

head(datas)

temp<-datas[,16]#diabetestemp

tempdata <- scale(datas,center=T,scale=T)

# Data centralization and standardization have the same meaning, in order to eliminate the impact of the dimension on the data structure. Two parameters in the #scale method: 1.center and scale default to true, ie T or TRUE

# 2.center is true for data centering, which refers to the data in the data set minus the mean of the data set.

# 3.scale is true to indicate data standardization, which means that the data after the centralization is divided by the standard deviation of the data set, that is, the data in the data set minus the mean of the data set divided by the standard deviation of the data set.

tempdata

tempdata[,16]<- temp

tempdata[,16]

Bc<-tempdata[,1:16]

Bc

set.seed(12345)

ind <- sample(2,nrow(Bc),replace = TRUE,prob = c(0.8,0.2))#

train.Bc <- Bc[ind==1,]

test.Bc <- Bc[ind==2,];

table(train.Bc[,16])

table(test.Bc[,16])

model.tune <- tune.svm(train.Bc[,1:15],train.Bc[,16],cost = 2^(-3:3),gamma = 2^(-4:4))

plot(model.tune,xlab=expression(gamma),ylab="reduction parameters C")#

# plot(model.tune)

model.tune#

bestsvm <- model.tune$best.model

summary(bestsvm)

n\_points\_in\_grid = 60 # num grid points in a line

x\_axis\_range <- range (datas[, 2]) # range of X axis

y\_axis\_range <- range (datas[, 1]) # range of Y axis

X\_grid\_points <- seq (from=x\_axis\_range[1], to=x\_axis\_range[2], length=n\_points\_in\_grid) # grid points along x-axis

Y\_grid\_points <- seq (from=y\_axis\_range[1], to=y\_axis\_range[2], length=n\_points\_in\_grid) # grid points along y-axis

all\_grid\_points <- expand.grid (X\_grid\_points, Y\_grid\_points) # generate all grid points

names (all\_grid\_points) <- c("Hwt", "Bwt") # rename

all\_points\_predited <- predict(svmfit, all\_grid\_points) # predict for all points in grid

color\_array <- c("red", "blue")[as.numeric(all\_points\_predited)] # colors for all points based on predictions

plot (all\_grid\_points, col=color\_array, pch=20, cex=0.25) # plot all grid points

points (x=trainingData$Hwt, y=trainingData$Bwt, col=c("red", "blue")[as.numeric(trainingData$response)], pch=19) # plot data points

points (trainingData[svmfit$index, c (2, 1)], pch=5, cex=2) # plot support vectors

svm<-svm(train.Bc[,1:15],train.Bc[,16],cost=4,gamma=0.0625,cross=10,probability=TRUE,type="C-classification",kernel="radial",scale=FALSE)#最初的

# svm<-svm(train.Bc[,1:15],train.Bc[,16])

summary(svm)

pre.model <- fitted(svm)#Check training dataset result

test.pred <- predict(svm,type="prob",newdata=test.Bc[,1:15],probability=TRUE,decision.values=TRUE)

table(True=train.Bc[,16],Prediction=pre.model)

table(True=test.Bc[,16],prediction=test.pred)

#ROCR

test.pred <- predict(svm,type="prob",newdata = test.Bc[,1:15],probability = T)

prob.rocr <- prediction(attr(test.pred,"probabilities")[,2],test.Bc[,16])

test.pref <- performance(prob.rocr,"tpr","fpr")

plot(test.pref,col=3)

# #Support Vector Machine SVM removes 5 independent variables

# head(data)

# datas <- data[,-c(7,9,13,16,17,18,19)]#

Remove five unrelated variables and id and frame categorical variables

# head(datas)

# temp<-datas[,13]#diabetes

# temp

# tempdata <- scale(datas)

# tempdata

# tempdata[,13]<- temp

# tempdata[,13]

# Bc<-tempdata[,1:13]

# Bc

# set.seed(2)

# ind <- sample(2,nrow(Bc),replace = TRUE,prob = c(0.8,0.2))

# train.Bc <- Bc[ind==1,]

# test.Bc <- Bc[ind==2,];

# table(train.Bc[,13])# Training set 262 non-disease, 55 sick# table(test.Bc[,13])

# # Grid Search Method

# model.tune <- tune.svm(train.Bc[,1:12],train.Bc[,13],cost = 2^(-3:3),gamma = 2^(-4:4))

# plot(model.tune)# Visual optimization process, if there are parameters located at the boundary of the given range, such as cost=2^4, then the range needs to be relaxed, continue to optimize

# model.tune

# # Find the optimal parameters to build the SVM model

# svm<-svm(train.Bc[,1:12],train.Bc[,13],cost=4,gamma=0.0625,cross=10,probability=TRUE,type="C-classification",kernel="radial",scale=FALSE)#Original

# summary(svm)

# pre.model <- fitted(svm)# View the classification effect of the training set

# test.pred <- predict(svm,type="prob",newdata=test.Bc[,1:12],probability=TRUE,decision.values=TRUE)

# table(True=test.Bc[,13],prediction=test.pred)

# #ROCR

# test.pred <- predict(svm,type="prob",newdata = test.Bc[,1:12],probability = T)

# prob.rocr <- prediction(attr(test.pred,"probabilities")[,2],test.Bc[,13])

# test.pref <- performance(prob.rocr,"tpr","fpr")

# plot(test.pref,col=3)

# # 6 correlated variable SVM

# head(data)

# datas <- data[,c(1,2,3,4,6,11,20)]#chol、stab.glu、hdl、age、ratio、waist、diabetes

# head(datas)

# temp<-datas[,7]#diabetes一行

# temp

# tempdata <- scale(datas)

# tempdata

# tempdata[,7]<- temp

# tempdata[,7]

# Bc<-tempdata[,1:7]

# Bc

# set.seed(2)

# ind <- sample(2,nrow(Bc),replace = TRUE,prob = c(0.8,0.2))#

# train.Bc <- Bc[ind==1,]

# test.Bc <- Bc[ind==2,];

# table(train.Bc[,7])

# table(test.Bc[,7])

# model.tune <- tune.svm(train.Bc[,1:6],train.Bc[,7],cost = 2^(-3:3),gamma = 2^(-4:4))

# plot(model.tune)

#Visualization optimization process, if there are parameters located at the boundary of the given range, such as cost=2^4, then the range needs to be relaxed, continue to optimize

# model.tune#Optimal parameters

## Find the optimal parameters to build the SVM model

# svm<-svm(train.Bc[,1:6],train.Bc[,7],cost=2,gamma=0.125,cross=10,probability=TRUE,type="C-classification",kernel="radial",scale=FALSE)#

# summary(svm)#

The result shows the accuracy of the 10-fold cross-validation

# pre.model <- fitted(svm)

# test.pred <- predict(svm,type="prob",newdata=test.Bc[,1:6],probability=TRUE,decision.values=TRUE)

# table(True=train.Bc[,7],Prediction=pre.model)

# table(True=test.Bc[,7],prediction=test.pred)

# #ROCR

# test.pred <- predict(svm,type="prob",newdata = test.Bc[,1:6],probability = T)

# prob.rocr <- prediction(attr(test.pred,"probabilities")[,2],test.Bc[,7])

# test.pref <- performance(prob.rocr,"tpr","fpr")

# plot(test.pref,col=3)

# D <- read.csv("Table.csv",header = T)#Read the date

# pre <- predict(svm,D)# Use the model just built to predict

# write.csv(pre,"Pre.csv")#store the result

# C-Support vector machine, the program is correct, but the prediction result is completely correct, can not be used

#Support vector machine uses all variables (remove id, frame, location, gender)

# head(data)

# datas <- data[,-c(16,17,18,19)]

# head(datas)

# datas

# temp<-datas[,16]#diabetes一行

# temp

# tempdata <- scale(datas)

# tempdata

# tempdata[,16]<- temp

# tempdata[,16]

# Bc<-tempdata[,1:16]

# Bc

# svm<-svm(Bc[,1:15],Bc[,16],type="C-classification")

# summary(svm)

# # View the effects of this hyperplane prediction

# pre.model <- fitted(svm)#

# table(True=Bc[,16],pre.model)

# model.tune <- tune.svm(Bc[,1:15],Bc[,16],cost = 10^(-1:4),gamma = 10^(-3:1))

# # plot(model.tune)

# model.tune

# # #

Find the optimal parameters and reconstruct the SVM model

# svm1<-svm(Bc[,1:15],Bc[,16],type="C-classification",cost=10,gamma=0.1,cross=10)

# summary(svm1)

# pre.svm1 <- fitted(svm1)

# table(True=Bc[,16],Prediction=pre.svm1)

# datas <- data[,c(1,2,4,6,14,20)]#chol、stab.glu、age、BMI、ratio、diabetes

# head(datas)

# temp<-datas[,6]#diabetes一行

# temp

# tempdata <- scale(datas)

# tempdata

# tempdata[,6]<- temp

# tempdata[,6]

# Bc<-tempdata[,1:6]

# set.seed(2)

# ind <- sample(2,nrow(Bc),replace = TRUE,prob = c(0.8,0.2))# Define sequence ind, 80% training set and 20% test set

# ind

# train.Bc <- Bc[ind==1,]

# test.Bc <- Bc[ind==2,];

# svm<-svm(train.Bc[,1:5],train.Bc[,6],type="C-classification",cost=10,kernel="radial",probability=TRUE,scale=FALSE)

# svm

# summary(svm)

# pred<-predict(svm,train.Bc[,1:5],decision.values=TRUE)

# pred

# train.Bc[,1:5]

# table(pred,train.Bc[,6])

# train.Bc

# pred<-predict(svm,test.Bc[,1:5],decision.values=TRUE)

# pred

# test.Bc[,1:5]

# table(pred,test.Bc[,6])