Homework3_MSA8150

Anutida Sangkla

2/13/2021

Question 1

part (a)

```
binary <- read.csv("BinaryData.csv", header=TRUE,  sep=",")
str(binary)

## 'data.frame':    60 obs. of  2 variables:
##  $ x: num  -0.9692 0.0957 0.5893 1.1097 1.5309 ...
##  $ y: int  0 0 0 0 0 0 0 0 0 0 ...

lr.fit <- glm(y~x,data = binary,family=binomial)
summary(lr.fit)

##
## Call:
## glm(formula = y ~ x, family = binomial, data = binary)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0617  -0.5460   0.1472   0.4474   1.9844
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.7776     0.4545  -1.711 0.087069 .
## x             1.2088     0.3175   3.807 0.000141 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 76.382  on 59  degrees of freedom
## Residual deviance: 42.824  on 58  degrees of freedom
## AIC: 46.824
##
## Number of Fisher Scoring iterations: 6

beta0<-lr.fit$coefficient[1]
beta1<-lr.fit$coefficient[2]
sprintf('%s = %3.6f', c('beta0', 'beta1'),c(beta0,beta1))

## [1] "beta0 = -0.777599" "beta1 = 1.208808"
```

From the summary, $\beta_0 = -0.777599$ and $\beta_1 = 1.208808$

part (b)

From

$$y = sigmoid(\beta_0 + \beta_1 x)$$

Where

$$sigmoid = \frac{1}{1 + e^{-z}}$$

$$y = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

Then, the negative log likelihood loss function is

$$L(\beta_0, \beta_1) = \sum_{i=1}^{n} y_i \ log(1 + e^{-\beta_0 + \beta_1 x_i}) + (1 - y_i) \ log(1 + e^{-\beta_0 + \beta_1 x_i})$$

$$\sum_{i=1}^{n} - log(1 + e^{\beta_0 + \beta_1 x_i}) + \sum_{i=1}^{n} y_i \log(\beta_0 + \beta_1 x_i)$$

$$\frac{\partial L(\beta_0, \beta_1)}{\beta_0}(\beta_0, \ \beta_1) = -\sum_{i=1}^{n} \frac{1}{1 + e^{\beta_0 + \beta 1 x_i}} e^{\beta_0 + \beta_1 x_i} \ + \ \sum_{i=1}^{n} y_i$$

$$\frac{\partial L(\beta_0, \beta_1)}{\beta_0}(\beta_0, \ \beta_1) = \sum_{i=1}^{n} s\,igmoid(\beta_0 + \beta_1 x_i) \ - \ y_i$$

$$\frac{\partial L(\beta_0, \beta_1)}{\beta_1}(\beta_0, \ \beta_1) = -\sum_{i=1}^{n} \frac{1}{1 + e^{\beta_0 + \beta 1 x_i}} e^{\beta_0 + \beta_1 x_i}(x_i) \ + \ \sum_{i=1}^{n} y_i \, x_i$$

$$\frac{\partial L(\beta_0, \beta_1)}{\beta_1}(\beta_0, \ \beta_1) = \sum_{i=1}^{n} s\,igmoid(\beta_0 + \beta_1 x_i) x_i \ - \ y_i x_i$$

part (c)

```
library(dplyr)

##
## Attaching package: 'dplyr'

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

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

```r
X <- binary$x
y <- binary$y
beta0<- rep(0,2)
beta1<- rep(0,2)


gradfunc <- function(X,y,beta0,beta1){
   sigmoid<- 1/(1+exp(-beta0 -beta1*X))
   grad0 <- sum((sigmoid - y))
   grad1 <- sum(X*sigmoid - X*y)
   result <- c(grad0,grad1)
   return(result)
}

gradDescent <-function(X,y, beta0, beta1, alpha, num_iters){
  for(i in 1:num_iters){
  grad<- function(){c(grad0,grad1)}
  grad<- gradfunc(X,y,beta0,beta1)
  beta0 <- beta0 - alpha*grad[1]
  beta1 <- beta1 - alpha*grad[2]
  }
  result<- list(beta0,beta1)
  return(result)
}

alpha <- 0.01
num_iters <- 500
results <- gradDescent(X, y, beta0, beta1, alpha, num_iters)
beta0 <- results[[1]][-1]
beta1 <- results[[2]][-1]
sprintf('%s = %3.6f', c('beta0', 'beta1'),c(beta0,beta1))

## [1] "beta0 = -0.777599" "beta1 = 1.208808"

#### Comparing the results to part (a)

lr.fit$coefficients

## (Intercept)           x
##  -0.7775995   1.2088080
```

Therefore, $\beta_0 = -0.777599$ and $\beta_1 = 1.208808$ from the gradient descent scheme, which means they are identical results as part (a)

Question 2

part (a)

```r
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.0.3

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

binary <- read.csv("BinaryData.csv", header=TRUE,  sep=",")
x0<- binary$x[1:20]
x1<-binary$x[21:60]
model <- qda(y~x, data = binary)
pi_0<- model$prior[1]
print(pi_0)

##         0
## 0.3333333

mu_0 <- model$means[1]
print(mu_0)

## [1] -1.181442e-17

var_0<- sum((x0-mu_0)^2)/(length(x0) - 1)
print(var_0)

## [1] 1
```

Therefore,

$$\pi_0 = 0.3333333 \ \ or \ \ 33\%$$

$$\mu_0 = \ 0$$

$$\sigma_0^2 = 1$$

```
pi_1<- model$prior[2]
print(pi_1)

##         1
## 0.6666667

mu_1 <- model$means[2]
print(mu_1)

## [1] 3

var_1<- sum((x1-mu_1)^2)/(length(x1) - 1)
print(var_1)

## [1] 4
```

Therefore,

$$\pi_1 = 0.6666667 \;\; or \;\; 67\%$$

$$\mu_1 = 3$$

$$\sigma_1^2 = 4$$

part (b)

```
delta0<- -0.5*log(var_0) -0.5*(x0-mu_0)*var_0*(x0-mu_0) + log(pi_0)
delta1<- -0.5*log(var_1) -0.5*(x1-mu_1)*var_1*(x1-mu_1) + log(pi_1)
delta = c(delta0,delta1)
decision = max(delta)
sprintf('%s = %3.6f', 'The decision point of this QDA model', decision)

## [1] "The decision point of this QDA model = -1.103189"
```

Therefore, the decision point is -1.103189.

Question 3

part (a)

Read data

```
titanic <- read.csv("Titanic.csv", header=TRUE,  sep=",")
str(titanic)

## 'data.frame':    891 obs. of  9 variables:
##  $ Survived: int  0 0 1 0 1 0 0 0 1 1 ...
##  $ Pclass  : int  3 3 3 2 2 3 3 3 3 3 ...
##  $ Name    : chr  "Abbing, Mr. Anthony" "Abbott, Mr. Rossmore Edward"
"Abbott, Mrs. Stanton (Rosa Hunt)" "Abelson, Mr. Samuel" ...
##  $ Sex     : chr  "male" "male" "female" "male" ...
##  $ Age     : num  42 16 35 30 28 30 26 40 18 26 ...
##  $ SibSp   : int  0 1 1 1 1 0 0 1 0 0 ...
##  $ Parch   : int  0 1 1 0 0 0 0 0 1 0 ...
##  $ Fare    : num  7.55 20.25 20.25 24 24 ...
##  $ Embarked: chr  "S" "S" "S" "C" ...
```

Drop the column ‘Name’

```
titanic2<- titanic[, -3]
str(titanic2)

## 'data.frame':    891 obs. of  8 variables:
##  $ Survived: int  0 0 1 0 1 0 0 0 1 1 ...
##  $ Pclass  : int  3 3 3 2 2 3 3 3 3 3 ...
##  $ Sex     : chr  "male" "male" "female" "male" ...
##  $ Age     : num  42 16 35 30 28 30 26 40 18 26 ...
##  $ SibSp   : int  0 1 1 1 1 0 0 1 0 0 ...
##  $ Parch   : int  0 1 1 0 0 0 0 0 1 0 ...
```

```
##  $ Fare    : num  7.55 20.25 20.25 24 24 ...
##  $ Embarked: chr  "S" "S" "S" "C" ...
```

Check missing values

```
Missingvalue = function (x) {sum(is.na(x)) }
apply(titanic2, 2, Missingvalue)

## Survived   Pclass      Sex      Age    SibSp    Parch     Fare Embarked 
##        0        0        0      177        0        0        0        0
```

Fill missing values for the column Age with the mean age

```
titanic2$Age[is.na(titanic2$Age)] = mean(titanic2$Age, na.rm=TRUE)
apply(titanic2, 2, Missingvalue)

## Survived   Pclass      Sex      Age    SibSp    Parch     Fare Embarked 
##        0        0        0        0        0        0        0        0

dim(titanic2)

## [1] 891   8
```

part (b)

```
train <- titanic2[1:750,]
test <- titanic2[751:891,]
dim(train)

## [1] 750   8

dim(test)

## [1] 141   8
```

part (c)

Convert variables types

```
titanic2$Pclass <- as.factor(titanic2$Pclass)
titanic2$Sex <- as.factor(titanic2$Sex)
titanic2$Embarked <- as.factor(titanic2$Embarked)
str(titanic2)

## 'data.frame':    891 obs. of  8 variables:
##  $ Survived: int  0 0 1 0 1 0 0 0 1 1 ...
##  $ Pclass  : Factor w/ 3 levels "1","2","3": 3 3 3 2 2 3 3 3 3 3 ...
##  $ Sex     : Factor w/ 2 levels "female","male": 2 2 1 2 1 2 2 1 1 2 ...
##  $ Age     : num  42 16 35 30 28 30 26 40 18 26 ...
##  $ SibSp   : int  0 1 1 1 1 0 0 1 0 0 ...
##  $ Parch   : int  0 1 1 0 0 0 0 0 1 0 ...
##  $ Fare    : num  7.55 20.25 20.25 24 24 ...
##  $ Embarked: Factor w/ 3 levels "C","Q","S": 3 3 3 1 1 3 3 3 3 1 ...

train <- titanic2[1:750,]
test <- titanic2[751:891,]
```

Logistic regression model

```
lr.fit <- glm(Survived~., data = train, family = 'binomial')
summary(lr.fit)
```

```
##
## Call:
## glm(formula = Survived ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7120  -0.6071  -0.4269   0.6390   2.4424
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.300650   0.522622   8.229  < 2e-16 ***
## Pclass2     -1.053284   0.321508  -3.276  0.00105 **
## Pclass3     -2.160271   0.322171  -6.705 2.01e-11 ***
## Sexmale     -2.710843   0.218952 -12.381  < 2e-16 ***
## Age         -0.042720   0.008758  -4.878 1.07e-06 ***
## SibSp       -0.309982   0.110833  -2.797  0.00516 **
## Parch       -0.055166   0.126158  -0.437  0.66191
## Fare         0.002054   0.002584   0.795  0.42661
## EmbarkedQ   -0.115701   0.401344  -0.288  0.77313
## EmbarkedS   -0.454426   0.263245  -1.726  0.08430 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1005.32  on 749  degrees of freedom
## Residual deviance:  668.26  on 740  degrees of freedom
## AIC: 688.26
##
## Number of Fisher Scoring iterations: 5
```

From the results, p-values of intercept and all numerical variables are

```
p.valueIntercept = coef(summary(lr.fit))[1, 4]
p.valueAge = coef(summary(lr.fit))[5, 4]
p.valueSibSp = coef(summary(lr.fit))[6, 4]
p.valueParch = coef(summary(lr.fit))[7, 4]
p.valueFare = coef(summary(lr.fit))[8, 4]
sprintf('%s = %4.5f', c('P-value of intercept', 'P-value of Age','P-value of
Sibsp', 'P-value of Parch', 'P-value of Fare'),c(p.valueIntercept,p.valueAge
,p.valueSibSp, p.valueParch,p.valueFare))
```

```
## [1] "P-value of intercept = 0.00000" "P-value of Age = 0.00000"
## [3] "P-value of Sibsp = 0.00516"     "P-value of Parch = 0.66191"
## [5] "P-value of Fare = 0.42661"
```

P-value of intercept = 2e-16

P-value of Age = 1.07e-06
P-value of Sibsp = 0.00516
P-value of Parch = 0.66191
P-value of Fare = 0.42661

Therefore, Parch and Fare features are the numerical variables that have large p-values.

The accuracy of the logistic regression model

```
pred.prob = predict(lr.fit, newdata= test, type="response")
pred.prob = ifelse(pred.prob > 0.5, 1, 0)
confusion.matrix = table(pred.prob, test$Survived)
TP<- confusion.matrix['1','1']; FN<- confusion.matrix['1','0']; TN<-
confusion.matrix['0','0']; FP<-confusion.matrix['0','1']
accuracy<-(TN+TP)/(TP+FN+TN+FP)
sprintf('%s = %3.6f', 'The accuracy of the logistic regression model is',
accuracy)

## [1] "The accuracy of the logistic regression model is = 0.801418"
```

From the results, the accuracy of logistic regression model is 0.8014.

LDA

```
lda.fit<- lda(Survived~., data = train)
lda.fit

## Call:
## lda(Survived ~ ., data = train)
##
## Prior probabilities of groups:
##         0         1
## 0.6066667 0.3933333
##
## Group means:
##     Pclass2   Pclass3   Sexmale      Age     SibSp     Parch     Fare
## 0 0.1912088 0.6725275 0.8549451 30.00947 0.5978022 0.3252747 22.52202
## 1 0.2474576 0.3627119 0.3254237 27.92512 0.4983051 0.4779661 47.41647
##    EmbarkedQ EmbarkedS
## 0 0.09890110 0.7692308
## 1 0.09830508 0.6338983
##
## Coefficients of linear discriminants:
##                  LD1
## Pclass2   -0.679206533
## Pclass3   -1.402603828
## Sexmale   -2.107178941
## Age       -0.026170100
## SibSp     -0.168895926
```

```
## Parch     -0.043687044
## Fare       0.001450777
## EmbarkedQ -0.048133361
## EmbarkedS -0.292748259
```

Accuracy of the LDA model

```
pred.lda = predict(lda.fit, newdata=test)
confusion.matrix<-table(Predicted=pred.lda$class, Survived=test$Survived)
TP<- confusion.matrix['1','1']; FN<- confusion.matrix['1','0']; TN<-
confusion.matrix['0','0']; FP<-confusion.matrix['0','1']
accuracy<-(TN+TP)/(TP+FN+TN+FP)
sprintf('%s = %3.6f', 'The accuracy of LDA model is', accuracy)

## [1] "The accuracy of LDA model is = 0.787234"
```

From the results, the accuracy of LDA model is 0.7872.

QDA

```
qda.fit<- qda(Survived~.,data = train)
qda.fit

## Call:
## qda(Survived ~ ., data = train)
##
## Prior probabilities of groups:
##         0         1
## 0.6066667 0.3933333
##
## Group means:
##     Pclass2   Pclass3   Sexmale      Age     SibSp     Parch     Fare
## 0 0.1912088 0.6725275 0.8549451 30.00947 0.5978022 0.3252747 22.52202
## 1 0.2474576 0.3627119 0.3254237 27.92512 0.4983051 0.4779661 47.41647
##    EmbarkedQ EmbarkedS
## 0 0.09890110 0.7692308
## 1 0.09830508 0.6338983
```

Accuracy of the QDA model

```
##Predicting test results.
pred.qda = predict(qda.fit, newdata=test)
confusion.matrix<-table(Predicted=pred.qda$class, Survived=test$Survived)
TP<- confusion.matrix['1','1']; FN<- confusion.matrix['1','0']; TN<-
confusion.matrix['0','0']; FP<-confusion.matrix['0','1']
accuracy<-(TN+TP)/(TP+FN+TN+FP)
sprintf('%s = %3.6f', 'The accuracy of QDA model is', accuracy)

## [1] "The accuracy of QDA model is = 0.801418"
```

From the results, the accuracy of QDA model is 0.8014.

Since the model accuracy of logistic regression and QDA are 0.8014,and the LDA is 0.7872, the logistic regression and QDA model seem the most accurate for this classification.

part (d)

Convert variables types

```
titanic2$Pclass <- as.numeric(titanic2$Pclass)
str(titanic2)

## 'data.frame':    891 obs. of  8 variables:
##  $ Survived: int  0 0 1 0 1 0 0 0 1 1 ...
##  $ Pclass  : num  3 3 3 2 2 3 3 3 3 3 ...
##  $ Sex     : Factor w/ 2 levels "female","male": 2 2 1 2 1 2 2 1 1 2 ...
##  $ Age     : num  42 16 35 30 28 30 26 40 18 26 ...
##  $ SibSp   : int  0 1 1 1 1 0 0 1 0 0 ...
##  $ Parch   : int  0 1 1 0 0 0 0 0 1 0 ...
##  $ Fare    : num  7.55 20.25 20.25 24 24 ...
##  $ Embarked: Factor w/ 3 levels "C","Q","S": 3 3 3 1 1 3 3 3 3 1 ...

train <- titanic2[1:750,]
test <- titanic2[751:891,]
```

Logistic regression model

```
lr.fit <- glm(Survived~., data = train, family = 'binomial')
summary(lr.fit)

## 
## Call:
## glm(formula = Survived ~ ., family = "binomial", data = train)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -2.7157  -0.6067  -0.4279   0.6401   2.4409  
## 
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)    
## (Intercept)  5.400374   0.616272   8.763  < 2e-16 ***
## Pclass      -1.084684   0.155753  -6.964 3.30e-12 ***
## Sexmale     -2.712003   0.218694 -12.401  < 2e-16 ***
## Age         -0.042822   0.008713  -4.915 8.89e-07 ***
## SibSp       -0.310629   0.110676  -2.807  0.00501 ** 
## Parch       -0.054572   0.126006  -0.433  0.66495    
## Fare         0.001994   0.002521   0.791  0.42896    
## EmbarkedQ   -0.117197   0.401217  -0.292  0.77021    
## EmbarkedS   -0.450139   0.260520  -1.728  0.08402 .  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 1005.32  on 749  degrees of freedom
## Residual deviance:  668.28  on 741  degrees of freedom
## AIC: 686.28
```

```
##
## Number of Fisher Scoring iterations: 5
```

From the results, p-values of intercept and all numerical variables are

```
p.valueIntercept = coef(summary(lr.fit))[1, 4]
p.valuePclass = coef(summary(lr.fit))[2,4]
p.valueAge = coef(summary(lr.fit))[4, 4]
p.valueSibSp = coef(summary(lr.fit))[5, 4]
p.valueParch = coef(summary(lr.fit))[6, 4]
p.valueFare = coef(summary(lr.fit))[7, 4]
sprintf('%s = %4.5f', c('P-value of intercept', 'P-value of Pclass' ,'P-value
of Age','P-value of Sibsp', 'P-value of Parch', 'P-value of
Fare'),c(p.valueIntercept,p.valuePclass,p.valueAge ,p.valueSibSp,
p.valueParch,p.valueFare))

## [1] "P-value of intercept = 0.00000" "P-value of Pclass = 0.00000"
## [3] "P-value of Age = 0.00000"       "P-value of Sibsp = 0.00501"
## [5] "P-value of Parch = 0.66495"     "P-value of Fare = 0.42896"
```

P-value of intercept = 2e-16
P-value of Pclass = 3.30e-12
P-value of Age = 8.89e-07
P-value of Sibsp = 0.00501
P-value of Parch = 0.664950
P-value of Fare = 0.428959

Therefore, Parch and Fare features are the numerical variables that have large p-values.

The accuracy of the logistic regression model

```
pred.prob = predict(lr.fit, newdata= test, type="response")
pred.prob = ifelse(pred.prob > 0.5, 1, 0)
confusion.matrix = table(pred.prob, test$Survived)
TP<- confusion.matrix['1','1']; FN<- confusion.matrix['1','0']; TN<-
confusion.matrix['0','0']; FP<-confusion.matrix['0','1']
accuracy<-(TN+TP)/(TP+FN+TN+FP)
sprintf('%s = %3.6f', 'The accuracy of the logistic regression model is',
accuracy)

## [1] "The accuracy of the logistic regression model is = 0.801418"
```

From the results, the accuracy of logistic regression model is 0.8014.

LDA

```
##Predicting test results.
lda.fit<- lda(Survived~., data = train)
lda.fit
```

```
## Call:
## lda(Survived ~ ., data = train)
##
## Prior probabilities of groups:
##         0         1
## 0.6066667 0.3933333
##
## Group means:
##     Pclass   Sexmale      Age     SibSp     Parch      Fare  EmbarkedQ
EmbarkedS
## 0 2.536264 0.8549451 30.00947 0.5978022 0.3252747 22.52202 0.09890110
0.7692308
## 1 1.972881 0.3254237 27.92512 0.4983051 0.4779661 47.41647 0.09830508
0.6338983
##
## Coefficients of linear discriminants:
##                    LD1
## Pclass    -0.705665832
## Sexmale   -2.108025075
## Age       -0.026218985
## SibSp     -0.169151267
## Parch     -0.042940284
## Fare       0.001389868
## EmbarkedQ -0.048406137
## EmbarkedS -0.289449342
```

Accuracy of the QDA model

```
pred.lda = predict(lda.fit, newdata=test)
confusion.matrix<-table(Predicted=pred.lda$class, Survived=test$Survived)
TP<- confusion.matrix['1','1']; FN<- confusion.matrix['1','0']; TN<-
confusion.matrix['0','0']; FP<-confusion.matrix['0','1']
accuracy<-(TN+TP)/(TP+FN+TN+FP)
sprintf('%s = %3.6f', 'The accuracy of LDA model is', accuracy)

## [1] "The accuracy of LDA model is = 0.787234"
```

From the results, the accuracy of LDA model is 0.7872.

QDA

```
qda.fit<- qda(train$Survived~.,data = train)
qda.fit

## Call:
## qda(train$Survived ~ ., data = train)
##
## Prior probabilities of groups:
##         0         1
## 0.6066667 0.3933333
##
## Group means:
##     Pclass   Sexmale      Age     SibSp     Parch      Fare  EmbarkedQ
```

```
EmbarkedS
## 0 2.536264 0.8549451 30.00947 0.5978022 0.3252747 22.52202 0.09890110
0.7692308
## 1 1.972881 0.3254237 27.92512 0.4983051 0.4779661 47.41647 0.09830508
0.6338983
```

Accuracy of the QDA model

```
pred.qda = predict(qda.fit, newdata=test)
confusion.matrix<- table(Predicted=pred.qda$class, Survived=test$Survived)
TP<- confusion.matrix['1','1']; FN<- confusion.matrix['1','0']; TN<-
confusion.matrix['0','0']; FP<-confusion.matrix['0','1']
accuracy<-(TN+TP)/(TP+FN+TN+FP)
sprintf('%s = %3.6f', 'The accuracy of QDA model is', accuracy)

## [1] "The accuracy of QDA model is = 0.829787"
```

From the results, the accuracy of QDA model is 0.8297.

Since the model accuracy of logistic regression is 0.8014, LDA is 0.7872, and QDA is 0.8297, the QDA model seems the most accurate one.

Question 4

part (a)

```
x <- c(8.2344,4.4854,5.4821,1.0953,2.1565,2.5096,4.9772,2.4998,4.2628,0.6933)
mu<- mean(x)
var<- sum((x-mu)^2)/(length(x) - 1)
D<- var/mu
sprintf('%s = %3.6f', 'D for this sample set',D)

## [1] "D for this sample set = 1.444883"
```

part (b)

```
B = 10000
D = rep(NA,B)
for (i in 1:B) {
  bootstrap.sample = sample(x, replace = TRUE)
  D[i] <- var(bootstrap.sample)/mean(bootstrap.sample)
}
sd.D<- sd(D)
sprintf('%s = %3.4f', 'The standard deviation of D',sd.D)

## [1] "The standard deviation of D = 0.5126"
```

Therefore, the standard deviation of D is 0.51.

part (c)

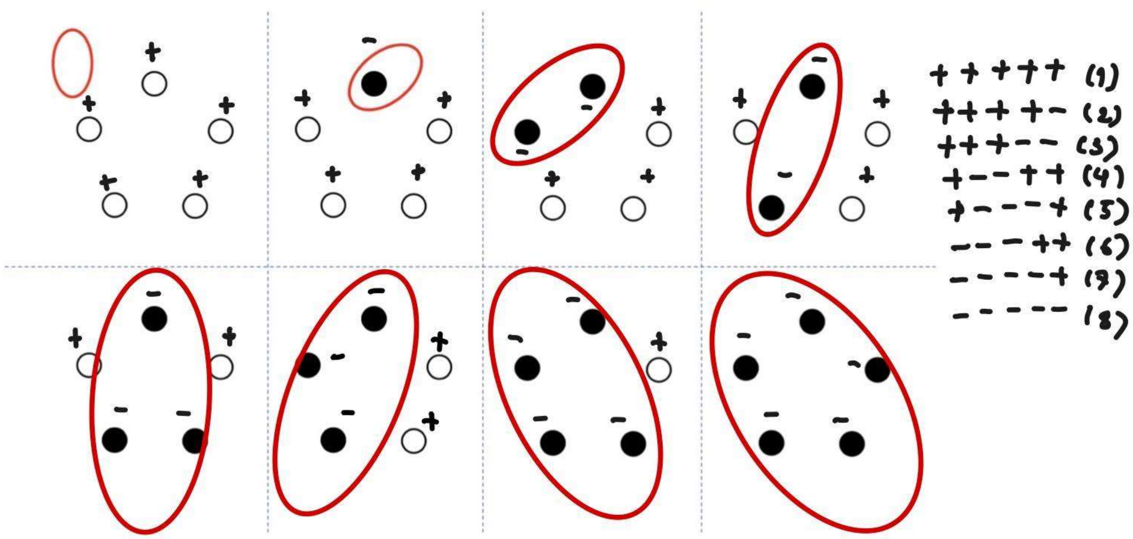
```r
B = 10000
mu = 3
sd = sqrt(3.24)
D = rep(NA, B)
for(i in 1:B) {
  bootstrap.sample = rnorm(10, mean = mu, sd = sd)
  D[i] <- var(bootstrap.sample)/mean(bootstrap.sample)
}
sd.D <- sd(D)
sprintf('%s = %3.3f', 'The standard deviation of D',sd.D)

## [1] "The standard deviation of D = 0.595"
```

Therefore, the standard deviation of D is 0.60.

Since the standard deviation of part (c) is equal to 0.51 and the standard deviation of part (d) is equal to 0.60, we will see that the standard deviation of D when we know that the reference distribution is normal distribution is a little bit difference to the case that we do not know the reference or we can say that they are equal if we round them up, the standard deviations of D in part (b) and part (c) equal to 1.

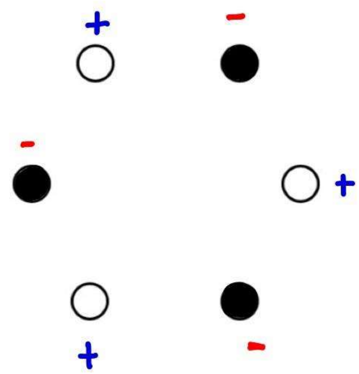
Question 5

part (a)



From the above figure, it indicates the a set of 5 points on the 2D plan can be shattered the set of ellipsoid classifier. We can conclude that the VCD of H is at least 5.

part (b)



If we have 6 points on the 2D plane, we label the negative points (black dots) and positive points (white dots) in the way that no ellipsoid can separate them. As a result, this figure shows an unrealizable dichotomy for the ellipsoid classifier.