1. (R and Python) For classification, assume that there may be more than two classes. You can assume that values of the class variable are integers starting with 1. Assume that a training dataset and a test dataset are available. Modify your program in Assignment #2 to do followings.

a. Prompt the user whether to run regression or classification.

(R code)

```
HW3 = function() {
   cat('Enter 1 to run Regression.','\n',
         'Enter 2 to run Classification.')
   ans = scan(n=1, quiet=TRUE)
   if(ans == 1){regression()} else {classification()}
}
```

(Python code)

```
def HW3() :
      method = input('Enter 1 to use Regression, Enter 2 to use Classification')
      if method == '1' :
            regression()
      else :
            classification()
```

- Prompt 창에 유저가 직접 Assignment2에서 만든 Regression과, 이번 과제에서 추가한 Classification 함수를 Regression은 1을 입력하고 Classification은 2를 입력하여 분석에 사용할 수 있는 HW3 함수를 생성했다.

b. If regression is chosen, perform the linear regression as you did in Assignment #2. (You have nothing to work on the regression algorithm in this assignment).

(R code)

```
HW3()
1
harris.dat
1
1
HW3KimDA_Reg_R_output
```

(Python code)

```
HW3()
1
harris.dat
1
1
HW3KimDA_Reg_ Python _output
```

(HW3KimDA_Reg_R_output)

```
Coefficients
-------------
Constant:     3526.422
Beta1:     90.02
Beta2:     1.269
Beta3:     23.406
Beta4:     722.461

ID, Actual values, Fitted values
--------------------------------
1, 3900, 4630.1
2, 4020, 4646.3
  (…..  중략  …..)
92, 6900, 6328.4
93, 8100, 6530.8

Model Summary
-------------
R-square = 0.5109
MSE = 254583.5835
```

(HW3KimDA_Reg_ Python _output)

```
Coefficients
-------------
Constant:     3526.422
Beta1:     90.02
Beta2:     1.269
Beta3:     23.406
Beta4:     722.461

ID, Actual values, Fitted values
--------------------------------
1, 3900, 4630.1
2, 4020, 4646.3
  (…..  중략  …..)
92, 6900, 6328.4
93, 8100, 6530.8

Model Summary
-------------
R-square = 0.5109
MSE = 254583.5835
```

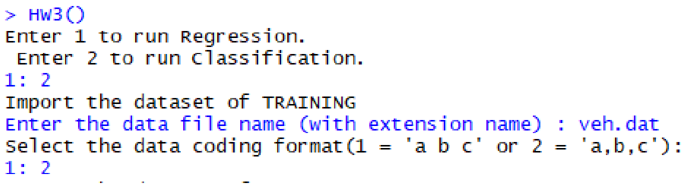
- HW3()을 이용하여 Regression을 실행하여 Assignment2와 동일한 결과로 성공적으로 회귀분석을 진행한 것을 확인하였다.

c. If classification is chosen, ask the user the filename of the training and test dataset.
(Assume the column location of the class variable is the same for both training and test dataset.)

(R code)

```
### importing dataset function ###
read = function(){
   data = readline("Enter the data file name (with extension name) : " )
   cat("Select the data coding format(1 = 'a b c' or 2 = 'a,b,c'): ")
   fm = scan(n=1, quiet=TRUE)
   if(fm == 1){form=""} else {form=","}
   read.table(data, sep=form)
}

# import training & test data file
   cat('Import the dataset of TRAINING','\n')
   train = read()
   cat('Import the dataset of TEST','\n')
   test = read()
```

(R console)

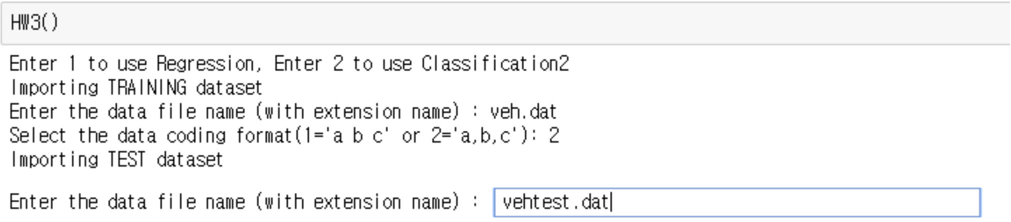


```
> HW3()
Enter 1 to run Regression.
 Enter 2 to run Classification.
1: 2
Import the dataset of TRAINING
Enter the data file name (with extension name) : veh.dat
Select the data coding format(1 = 'a b c' or 2 = 'a,b,c'):
1: 2
```

(Python code)

```
### importing data function ###
def read():
      import pandas as pd
      name = input("Enter the data file name (with extension name) : ")
      fm = input("Select the data coding format(1='a b c' or 2='a,b,c'): " )
      if fm == '1':
            form = " "
      elif fm == '2':
            form = ","
      return pd.read_csv(name, sep=form, header=None)

# prompt user to enter the data
   print('Importing TRAINING dataset')
   train = read()
   print('Importing TEST dataset')
   test = read()
```

(Python console)



```
HW3()

Enter 1 to use Regression, Enter 2 to use Classification2
Importing TRAINING dataset
Enter the data file name (with extension name) : veh.dat
Select the data coding format(1='a b c' or 2='a,b,c'): 2
Importing TEST dataset

Enter the data file name (with extension name) : vehtest.dat
```

- 가독성과 편의를 위해 Data를 import하는 read function을 따로 만들어, Classification내의 Training dataset과 Test dataset을 유저가 직접 파일 이름을 입력하여 import하게끔 만들었다.

d. Make the program to implement (i) LDA and (ii) QDA that can handle more than two classes.

(R 코드)

```
# Basic vectors
N = nrow(train)
n = nrow(test)
x = t(train[-num])
x.test = t(as.matrix(test[,-num]))
prior = as.matrix(sapply(prior,function(x){eval(parse(text=x))}))
means = t(as.matrix(aggregate(train[-num], train[num], mean)[,-1]))
cov = lapply(lapply(split(train,train[,num]), function(x){x[,-num]}), cov)
covs = lapply( cov, function(x){ (nrow(x)-1)*x /(N - nclass) })
sp = Reduce('+', covs)


# (i) LDA
if(choice==1) {
    d.resub = matrix(0,nrow=N,ncol=nclass)
    d.test = matrix(0,nrow=n,ncol=nclass)

    for(i in 1:nclass) {
    t0 = t(t(means[,i])%*%solve(sp)%*%x)
    t1 = t(t(means[,i])%*%solve(sp)%*%x.test)
    t2 = t((-0.5*t(means[,i])%*%solve(sp)%*%means[,i] + log(prior[i]))%*%matrix(1,nrow=1,ncol=N))
    t3 = t((-0.5*t(means[,i])%*%solve(sp)%*%means[,i] + log(prior[i]))%*%matrix(1,nrow=1,ncol=n))
    d.resub[,i]=t0+t2
    d.test[,i]=t1+t3 }

    result.resub = cbind(train[,num], max.col(d.resub))
    result.test = cbind(test[,num], max.col(d.test))

    } else if(choice==2){

# (ii) QDA
      d.resub = matrix(0,nrow=N,ncol=nclass)
      d.test = matrix(0,nrow=n,ncol=nclass)

      for(i in 1:nclass) {
        t0 = -0.5*log(det(cov[[i]]))
        for(j in 1:N) {
        t1 = -0.5*t(x[,j]-means[,i])%*%solve(cov[[i]])%*%(x[,j]-means[,i]) + log(prior[i])
        d.resub[j,i]=t0+t1 }
        for(j in 1:n) {
        t2 = -0.5*t(x.test[,j]-means[,i])%*%solve(cov[[i]])%*%(x.test[,j]-means[,i]) + log(prior[i])
        d.test[j,i]=t0+t2 }
        }

      result.resub = cbind(train[,num], max.col(d.resub))
      result.test = cbind(test[,num], max.col(d.test))
      } else warning ('Choose 1 for LDA or 2 for QDA')

# Confusion Matrix
  t1 = table(result.resub[,1], result.resub[,2], dnn=c("Actual Class","Predicted Class"))
  t2 = table(result.test[,1], result.test[,2], dnn=c("Actual Class","Predicted Class"))

  # Accuracy
  accuracy.resub = sum(result.resub[,1] == result.resub[,2])/N
  accuracy.test = sum(result.test[,1] == result.test[,2])/n
```

(Python code)

```
# Basic vectors
N = train.shape[0]
n = test.shape[0]
np = train[num].groupby(train[num]).count()
x = train.drop(num, axis=1).T
x_test = test.drop(num, axis=1).T
y = train[num]
y_test = test[num]
y_pred = []
y_test_pred = []
prior = pd.DataFrame(prior)

means = train.groupby(train[num]).mean().T
cov = train.groupby(train[num]).cov()
covs=[]
sp = 0
for i in range(1, nclass+1):
    covs.append(cov.loc[i])
    sp = sp + (np[i]-1)*cov.loc[i]/(N-nclass)

# (i) LDA
    if choice == 1 :
        dresub=[]
        dtest=[]

        for i in range(1, nclass+1):
            t0 = (means[i].T).dot(lin.inv(sp)).dot(x)
            t1 = (means[i].T).dot(lin.inv(sp)).dot(x_test)
            t2 = (-0.5)*means[i].T.dot(lin.inv(sp)).dot(means[i])+math.log(prior.T[i-1])
            t3 = (-0.5)*means[i].T.dot(lin.inv(sp)).dot(means[i])+math.log(prior.T[i-1])
            dresub.append(t0+t2)
            dtest.append(t1+t3)
        y_pred = pd.DataFrame(dresub).idxmax()+1
        y_test_pred = pd.DataFrame(dtest).idxmax()+1

# (ii) QDA
    elif choice == 2 :
        import numpy as np
        dresub = np.zeros((nclass,N))
        dtest = np.zeros((nclass,n))

        for i in range(0, nclass):
            for j in range(0, N):
                t0 = (-0.5)*math.log(lin.det(cov.loc[i+1]))
                t1 = (-0.5)*((x[j]-means[i+1]).T).dot(lin.inv(cov.loc[i+1])).dot(x[j]-means[i+1]) +
math.log(prior.T[i])
                dresub[i,j]= t0+t1
            for j in range(0, n):
                t2 = (-0.5)*((x_test[j]-means[i+1]).T).dot(lin.inv(cov.loc[i+1])).dot(x_test[j]-means[i+1]) +
math.log(prior.T[i])
                dtest[i,j]= t0+t2

        y_pred = pd.DataFrame(dresub).idxmax()+1
        y_test_pred = pd.DataFrame(dtest).idxmax()+1

# Crosstable
    import numpy as np
    cross_res = pd.crosstab(y, y_pred, colnames=[''])
    cross_res.index = np.array(["Actual","Class",""]).repeat([1,1,nclass-2])
    accuracy_res = np.trace(cross_res)/N

    cross_t = pd.crosstab(y_test, y_test_pred, colnames=[''])
    cross_t.index = np.array(["Actual","Class",""]).repeat([1,1,nclass-2])
    accuracy_t = np.trace(cross_t)/n
```

- Class의 size를 nclass에 저장하여, 2개 이상의 class의 개수에 맞춰 LDA와 QDA를 수행하는 R과 Python 코드이다.

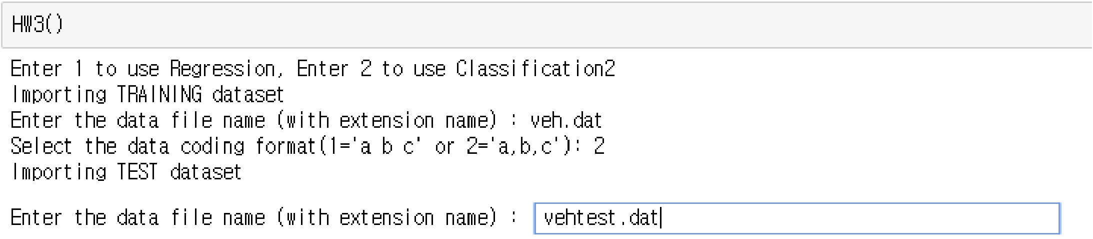
e. Perform (i) LDA or (ii) QDA depending on the choice by the user. Use a data file named 'veh.dat' for the training and 'vehtest.dat' as the test data.

(R code)

```
# for classification of veh.dat and vehtest.dat
hw3()
2       # classification
veh.dat       # training dataset
2       # form
vehtest.dat       # test dataset
2       # form
19       # response var
2       # QDA
1/4,1/4,1/4,1/4       # arbitrary set equal prior
HW3KimDA_R_output       # ouput file name
```

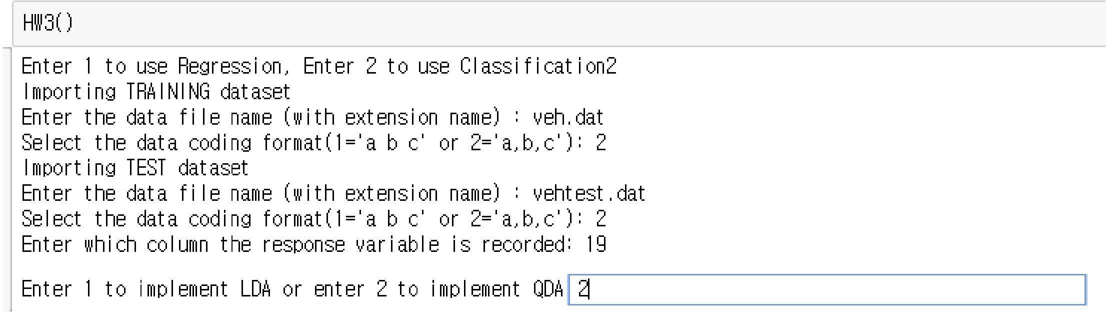
(Python console)

# importing datasets



```
HW3()

Enter 1 to use Regression, Enter 2 to use Classification2
Importing TRAINING dataset
Enter the data file name (with extension name) : veh.dat
Select the data coding format(1='a b c' or 2='a,b,c'): 2
Importing TEST dataset

Enter the data file name (with extension name) : vehtest.dat
```

# choose QDA



```
HW3()

Enter 1 to use Regression, Enter 2 to use Classification2
Importing TRAINING dataset
Enter the data file name (with extension name) : veh.dat
Select the data coding format(1='a b c' or 2='a,b,c'): 2
Importing TEST dataset
Enter the data file name (with extension name) : vehtest.dat
Select the data coding format(1='a b c' or 2='a,b,c'): 2
Enter which column the response variable is recorded: 19

Enter 1 to implement LDA or enter 2 to implement QDA 2
```

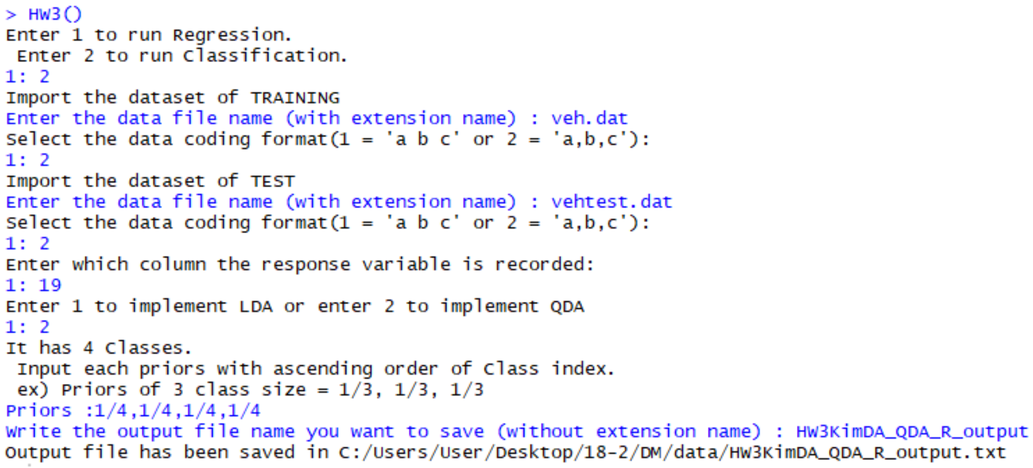
- R과 Python 유저가 직접 LDA는 1을, QDA는 2을 입력하여 Classification을 수행하도록 prompt창을 통해 구성하였다. Training dataset은 veh.dat, Test dataset은 vehtest.dat으로 Classification 한 결과이다.

f. The output file for classification generated by the program must look like below. (The numbers are fictitious).

(R code)

```
# make output file
   outputname = readline("Write the output file name you want to save (without extension name) : ")
   outputname = paste(outputname,".txt",sep="")


   cat("ID, Actual class, Resub pred", "\n", "-------------------------", "\n", file = outputname, sep="")
   write.table(cbind(c(1:N), result.resub), outputname, sep= ", ", row.names=FALSE, col.names=FALSE,
append=TRUE, quote=FALSE)

   cat("\n", "Confusion Matrix (Resubstitution)", "\n", "-------------------------------", "\n",file =
outputname,sep="", append=TRUE)
   capture.output(print(t1),file=outputname,append=TRUE)

   cat("\n", "Model Summary (Resubstitution)", "\n", "-------------------------------", "\n",file =
outputname,sep="", append=TRUE)
   cat("Overall accuracy: ", round(accuracy.resub,3), "\n\n",file = outputname, sep="", append=TRUE)


   cat("ID, Actual class, Test pred", "\n", "-------------------------", "\n",file = outputname,sep="",
append=TRUE)
   write.table(cbind(c(1:n), result.test), file = outputname, sep= ", ", row.names=FALSE,
col.names=FALSE, append=TRUE, quote=FALSE)

   cat("\n", "Confusion Matrix (Test)", "\n", "-------------------------------", "\n",file = outputname,sep="",
append=TRUE)
   capture.output(print(t2),file=outputname,append=TRUE)

   cat("\n", "Model Summary (Test)", "\n", "-------------------------------", "\n",file = outputname,sep="",
append=TRUE)
   cat("Overall accuracy: ", round(accuracy.test,3), "\n" ,file = outputname,sep="", append=TRUE)

   cat("Output file has been saved in ",getwd(),"/",outputname,sep="")
}
```

(R console)



```
> HW3()
Enter 1 to run Regression.
 Enter 2 to run Classification.
1: 2
Import the dataset of TRAINING
Enter the data file name (with extension name) : veh.dat
Select the data coding format(1 = 'a b c' or 2 = 'a,b,c'):
1: 2
Import the dataset of TEST
Enter the data file name (with extension name) : vehtest.dat
Select the data coding format(1 = 'a b c' or 2 = 'a,b,c'):
1: 2
Enter which column the response variable is recorded:
1: 19
Enter 1 to implement LDA or enter 2 to implement QDA
1: 2
It has 4 classes.
 Input each priors with ascending order of class index.
 ex) Priors of 3 class size = 1/3, 1/3, 1/3
Priors :1/4,1/4,1/4,1/4
Write the output file name you want to save (without extension name) : HW3KimDA_QDA_R_output
Output file has been saved in C:/Users/User/Desktop/18-2/DM/data/HW3KimDA_QDA_R_output.txt
```

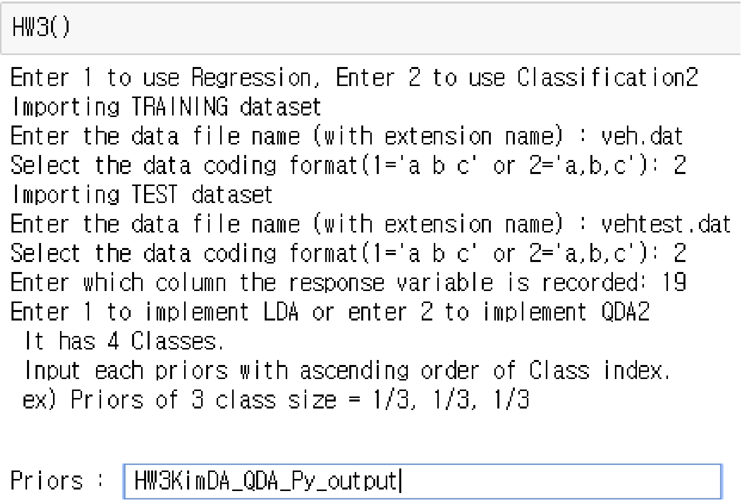
(R Output)

```
ID, Actual class, Resub pred
-----------------------------
1, 1, 1
2, 1, 1
3, 1, 1
4, 1, 1
5, 1, 1
(생략)
422, 4, 4
423, 4, 4
424, 4, 4
425, 4, 4

Confusion Matrix (Resubstitution)
----------------------------------
           1    2     3     4
Actual   91   13     0     1
Class     9   99     0     2
          0    0   108     2
          0    0     0   100

Model Summary (Resubstitution)
------------------------------
Overall accuracy = 0.936

ID, Actual class, Test pred
-----------------------------
1, 1, 1
2, 1, 1
3, 1, 2
4, 1, 4
5, 1, 1
(생략)
333, 4, 4
334, 4, 4
335, 4, 4
336, 4, 4

Confusion Matrix (Test)
----------------------------------
           1    2    3    4
Actual   64   17    1    4
Class    26   51    1    7
          0    0   84    2
          0    0    1   78

Model Summary (Test)
------------------------------
Overall accuracy = 0.824
```

(Python code)

```
# output file name
outputname = input("Write the output file name you want to save (without extension name) : ")
outputname = outputname+'.txt'

# outport the result
with open(outputname,"w") as text_file:

    print('ID, Actual class, Resub pred', file=text_file)
    print('----------------------------', file=text_file)
    for i in range(N):
        print(i+1, y[i], y_pred[i], sep=', ', file=text_file)
    print('',file=text_file)
    print('Confusion Matrix (Resubstitution)', file=text_file)
    print('---------------------------------', file=text_file)
    print(cross_res, file=text_file)
    print("",file=text_file)
    print("Model Summary (Resubstitution)", file=text_file)
    print('-----------------------------', file=text_file)
    print("Overall accuracy = ", accuracy_res.round(3), sep='', file=text_file)
    print('', file=text_file)

    print('ID, Actual class, Test pred', file=text_file)
    print('----------------------------', file=text_file)
    for i in range(n):
        print(i+1, y_test[i], y_test_pred[i], sep=', ', file=text_file)
    print('',file=text_file)
    print('Confusion Matrix (Test)', file=text_file)
    print('---------------------------------', file=text_file)
    print(cross_t, file=text_file)
    print("",file=text_file)
    print("Model Summary (Test)", file=text_file)
    print('-----------------------------', file=text_file)
    print("Overall accuracy = ", accuracy_t.round(3), sep='', file=text_file)
    print('', file=text_file)
```

(Python console)



```
HW3()

Enter 1 to use Regression, Enter 2 to use Classification2
Importing TRAINING dataset
Enter the data file name (with extension name) : veh.dat
Select the data coding format(1='a b c' or 2='a,b,c'): 2
Importing TEST dataset
Enter the data file name (with extension name) : vehtest.dat
Select the data coding format(1='a b c' or 2='a,b,c'): 2
Enter which column the response variable is recorded: 19
Enter 1 to implement LDA or enter 2 to implement QDA2
 It has 4 Classes.
 Input each priors with ascending order of Class index.
 ex) Priors of 3 class size = 1/3, 1/3, 1/3


Priors :  HW3KimDA_QDA_Py_output
```

(Python Output)

```
ID, Actual class, Resub pred
-----------------------------
1, 1, 1
2, 1, 1
3, 1, 1
4, 1, 1
5, 1, 1
(생략)
422, 4, 4
423, 4, 4
424, 4, 4
425, 4, 4

Confusion Matrix (Resubstitution)
----------------------------------
            1    2     3     4
Actual    91   13     0     1
Class      9   99     0     2
           0    0   108     2
           0    0     0   100

Model Summary (Resubstitution)
------------------------------
Overall accuracy = 0.936

ID, Actual class, Test pred
-----------------------------
1, 1, 1
2, 1, 1
3, 1, 2
4, 1, 4
5, 1, 1
(생략)
333, 4, 4
334, 4, 4
335, 4, 4
336, 4, 4

Confusion Matrix (Test)
----------------------------------
            1    2    3    4
Actual    64   17    1    4
Class     26   51    1    7
           0    0   84    2
           0    0    1   78

Model Summary (Test)
------------------------------
Overall accuracy = 0.824
```

- R과 Python으로 만든 HW3 함수로 Training dataset은 veh.dat, Test dataset은 vehtest.dat으로 Classification 한 결과이다. QDA를 사용하고, prior은 임의로 equal prior인 1/4씩 주었다. R과 Python 모두 동일한 결과를 확인할 수 있으며, 또한 Resubstitution Accuracy rate보다 Test Accuracy rate이 낮아지는 것을 알 수 있다. LDA 결과는 Classification method 선택 prompt에서 1을 선택하면 된다. 그 결과 또한 R과 Python이 동일하다.