# Homework 3

## Q1

1. Suppose that we wish to predict whether a given stock will issue a dividend this year ("Yes" or "No") based on X, last year's percent profit. We examine a large number of companies and discover that the mean value of X for companies that issued a dividend was $\bar{x} = 10$, while the mean for those that didn't was $\bar{x} = 0$. In addition, the variance of X for these two sets of companies was $\sigma^2 = 36$. Finally, 80% of companies issued dividends. Assuming that X follows a normal distribution, predict the probability that a company will issue a dividend this year given that its percentage profit was X = 4 last year.

   (a) Write down what is P(X | Dividend = Yes) (3 marks)
   The probability density function for last year's percent profit that comes from a company who will issue a dividend this year.

   (b) Write down what is P(X | Dividend = No) (3 marks)
   The probability density function for laset year's percent profit that comes from a company who will not issue a dividend this year.

   (c) Use dnorm() function in R to calculate conditional probabilities in a) and b) when X = 4 (4 marks, 2 marks each)

```r
dnorm(4, mean=10, sd= 6) #conditional probability in (a)
```

```
## [1] 0.04032845
```

```r
dnorm(4, mean=0, sd= 6) #conditional probability in (b)
```

```
## [1] 0.05324133
```

(d) What is the value of P(Dividend = Yes)? (2 marks)
80%

(e) What is the value of P( Dividend = No)? (2 marks)
20%

(f) Now predict the probability that a company will issue a dividend this year given that its percentage profit was X = 4 last year. Hint: Use Bayes' rule as we discussed in the class. (6 marks)
By Bayes Theorm,

$Pr(Dividend = Yes|X = 4) = \frac{\pi_1 f_1(x)}{\pi_1 f_1(x)+\pi_2 f_2(x)} = \frac{0.8*0.0403}{0.8*0.0403+0.2*0.0532} = 75.14\%$

## Q2

In this problem, you will develop a model to predict whether a given car gets high or low gas mileage based on the Auto data set.

(a) Create a binary variable, mpg01, that contains a 1 if mpg contains a value above its median, and a 0 if mpg contains a value below its median. You can compute the median using the median() function in R. Note you may find it helpful to use the data.frame() function to create a single data set containing both mpg01 and the other Auto variables. (2 marks)

```r
library(ISLR)
summary(Auto)
```

```
##       mpg          cylinders      displacement     horsepower
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0
##
##      weight      acceleration        year           origin
##  Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000
##  1st Qu.:2225   1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000
##  Median :2804   Median :15.50   Median :76.00   Median :1.000
##  Mean   :2978   Mean   :15.54   Mean   :75.98   Mean   :1.577
##  3rd Qu.:3615   3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000
##  Max.   :5140   Max.   :24.80   Max.   :82.00   Max.   :3.000
##
##                  name
##  amc matador       :  5
##  ford pinto        :  5
##  toyota corolla    :  5
##  amc gremlin       :  4
##  amc hornet        :  4
##  chevrolet chevette:  4
##  (Other)           :365
```

```r
attach(Auto)
```

```r
median(mpg)
```

```
## [1] 22.75
```

```r
mpg01 = rep(0, length(mpg))
mpg01[mpg>median(mpg)]=1
Auto_2 <- data.frame(Auto,mpg01)
```

(b) Which of the continuous features seem most likely to be useful in predicting mpg? Use cor() function in R and consider features with correlation coefficients > 0.6 as useful in predicting. (2 marks)

```r
cor(Auto_2[, -9])
```

```
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
## horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
## weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
## acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
```

```
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
## mpg01         0.8369392 -0.7591939   -0.7534766 -0.6670526 -0.7577566
##              acceleration       year     origin      mpg01
## mpg             0.4233285  0.5805410  0.5652088  0.8369392
## cylinders      -0.5046834 -0.3456474 -0.5689316 -0.7591939
## displacement   -0.5438005 -0.3698552 -0.6145351 -0.7534766
## horsepower     -0.6891955 -0.4163615 -0.4551715 -0.6670526
## weight         -0.4168392 -0.3091199 -0.5850054 -0.7577566
## acceleration    1.0000000  0.2903161  0.2127458  0.3468215
## year            0.2903161  1.0000000  0.1815277  0.4299042
## origin          0.2127458  0.1815277  1.0000000  0.5136984
## mpg01           0.3468215  0.4299042  0.5136984  1.0000000
```

As we can see, the useful feasures are displacement, horsepower, weight which have correlations with |mpg01| > 0.6.

(c) Split the data into a training set and a test set holding 30% of data for testing. Use sample.split() function in the library 'caTools' in R to split the data with the random seed 101. Use set.seed() function in R to assign the random seed. (3 marks)

```r
library(caTools)
set.seed(101)
sample = sample.split(1:nrow(Auto_2), SplitRatio=0.3)
test = subset(Auto_2, sample==TRUE)
train = subset(Auto_2, sample==FALSE)
```

(d) Perform LDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg in (b). You may not include mpg as it was used to derive mpg01. What is the test error of the model obtained? Use lda() function in the library 'MASS' in R.(3 marks)

```r
library(MASS)
lda.fit=lda(mpg01~weight+displacement+horsepower,data=train)
lda.pred = predict(lda.fit, test)
mean(lda.pred$class != test$mpg01)
```

```
## [1] 0.1367521
```

The test error by LDA is 13.67%.

(e) Perform QDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg in (b). You may not include mpg as it was used to derive mpg01. What is the test error of the model obtained? Use qda() function in the library 'MASS' in R (3 marks)

```r
qda.fit=qda(mpg01~weight+displacement+horsepower,data=train)
qda.pred = predict(qda.fit, test)
mean(qda.pred$class != test$mpg01)
```

```
## [1] 0.1282051
```

The test error by QDA is 12.82%.

(f) Perform logistic regression on the training data in order to predict mpg01 using the variables that seemed most associated with mpg in (b). You may not include mpg as it was used to derive mpg01. What is the test error of the model obtained? (3 marks)

```
glm.fit=glm(mpg01~weight+displacement+horsepower,family=binomial,data=train)
glm.probs=predict(glm.fit, test, type="response")
glm.pred=rep(0,length(glm.probs))
glm.pred[glm.probs > 0.5] = 1
mean(glm.pred != test$mpg01)
```

```
## [1] 0.1452991
```

The test error by logistic regression is 14.53%.

(g) Perform KNN on the training data, with several values of K (use K=1, K=5, K=10, K=15, K=20, K=30, K=50, K=100, K=150, K=200) in order to predict mpg01. Use only the variables that seemed most associated with mpg in (b). You may not include mpg as it was used to derive mpg01. Obtain test errors corresponds to each K. Which value of K seems to perform the best on this data set? Use knn() function in the library (4 marks)

```
library(class)
train2=cbind(train$weight,train$displacement,train$horsepower)
test2=cbind(test$weight,test$displacement,test$horsepower)
knn.1=mean(knn(train2,test2, train$mpg01,k=1) != test$mpg01)
knn.5=mean(knn(train2,test2, train$mpg01,k=5) != test$mpg01)
knn.10=mean(knn(train2,test2, train$mpg01,k=10) != test$mpg01)
knn.15=mean(knn(train2,test2, train$mpg01,k=15) != test$mpg01)
knn.20=mean(knn(train2,test2, train$mpg01,k=20) != test$mpg01)
knn.30=mean(knn(train2,test2, train$mpg01,k=30) != test$mpg01)
knn.50=mean(knn(train2,test2, train$mpg01,k=50) != test$mpg01)
knn.100=mean(knn(train2,test2, train$mpg01,k=100) != test$mpg01)
knn.150=mean(knn(train2,test2, train$mpg01,k=150) != test$mpg01)
knn.200=mean(knn(train2,test2, train$mpg01,k=200) != test$mpg01)

knn.1
```

```
## [1] 0.1623932
```

```
knn.5
```

```
## [1] 0.1709402
```

```
knn.10
```

```
## [1] 0.1709402
```

```
knn.15
```

```
## [1] 0.1794872
```

```r
knn.20
```

```
## [1] 0.1452991
```

```r
knn.30
```

```
## [1] 0.1538462
```

```r
knn.50
```

```
## [1] 0.1538462
```

```r
knn.100
```

```
## [1] 0.1538462
```

```r
knn.150
```

```
## [1] 0.1709402
```

```r
knn.200
```

```
## [1] 0.1965812
```

Thus, K=20 performs better than other values of K.