By Michael Ekubay Negassi & Lu Yang

Full Name	Student ID
Michael Ekubay Negassi	002238528
Lu Yang	002238352

Statistical Learning Summer Project

1.

Prerequisites

Install these packages for and libraries for Question 1.

```
install.packages("questionr")
library(questionr)
```

Load **lowbwt.txt** file from saved Directory

```
temp<-read.table("C:/Challenge2/lowbwt.txt",header = TRUE)
```

#a)

Write the equation and interpret β1, then estimated coefficient of Apgar score.

```
apgar5<-temp$apgar5
germ.hem<-temp$germ.hem
apgar5.glm = glm(formula=germ.hem ~ apgar5, data=temp, family=binomial
(link="logit"))
summary(apgar5.glm)
```

```
Call:
glm(formula = germ.hem ~ apgar5, family = binomial(link = "logit"
),
    data = temp)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0514  -0.5528  -0.4645  -0.3877   2.1891

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.3037     0.6191  -0.491   0.6237
apgar5       -0.2496     0.1044  -2.392   0.0168 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 84.542  on 99  degrees of freedom
Residual deviance: 78.927  on 98  degrees of freedom
AIC: 82.927
Number of Fisher Scoring iterations: 4
```

Equation : ln (p/1-p) = -0.3037-0.2496x

For each one unit of increase in the apgar score, the log odds of exper iencing a germinal matrix hemorrhage decrease by 0.2496

#b)

confint(apgar5.glm)

```
                 2.5 %      97.5 %
(Intercept) -1.5832887   0.89613184
apgar5      -0.4599101  -0.04420464
```

exp(coef(apgar5.glm))

```
(Intercept)     apgar5
  0.7380593   0.7791066
```

exp(cbind(OR=coef(apgar5.glm),confint(apgar5.glm)))

```
                   OR     2.5 %    97.5 %
(Intercept) 0.7380593 0.2052988 2.4501073
apgar5      0.7791066 0.6313404 0.9567581
```

When a logistic regression is calculated, the regression coefficient (β1) is the estimated increase in the log odds of the *outcome per unit increase* in the value of the *exposure.* In other words, the exponential function of the regression coefficient (*eb1*) is the odds ratio associated with a one-unit increase in the exposure. Odds ratio could also be obtained with exp(coef(x)) and confidence intervals with exp(confint(x)).

#c)

```
H0: Î²1=0 Logit graham apgar5

Iteration 0: log likelihood = -42.270909
Iteration 1: log likelihood = -39.727053
Iteration 2: log likelihood = -39.463638
Iteration 3: log likelihood = -39.463411

Logistic regression Log likelihood = -39.463411
```

#d)

```
new.apgar<-data.frame(apgar5=3)
prob<-predict(apgar5.glm,new.apgar,type = "response")
prob
```

0.2587351

```
new.apgar2<-data.frame(apgar5=7)
prob2<-predict(apgar5.glm,new.apgar2,type = "response")
prob2
```

0.1139531

2.

__Prerequisites__

#Install these packages for and libraries for Question 2.

```
install.packages("ISLR")
library(ISLR)
install.packages("corrplot")
```

Load Auto CSV file to variable AUTO and
Clearing data with "?" values from source horsepower column

```
Auto<-read.csv("C:/Challenge2/Auto.csv",header = TRUE,sep = ',')
Auto$horsepower <- as.numeric(as.character(Auto$horsepower))
Auto <- na.omit(Auto)
```

#a)

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

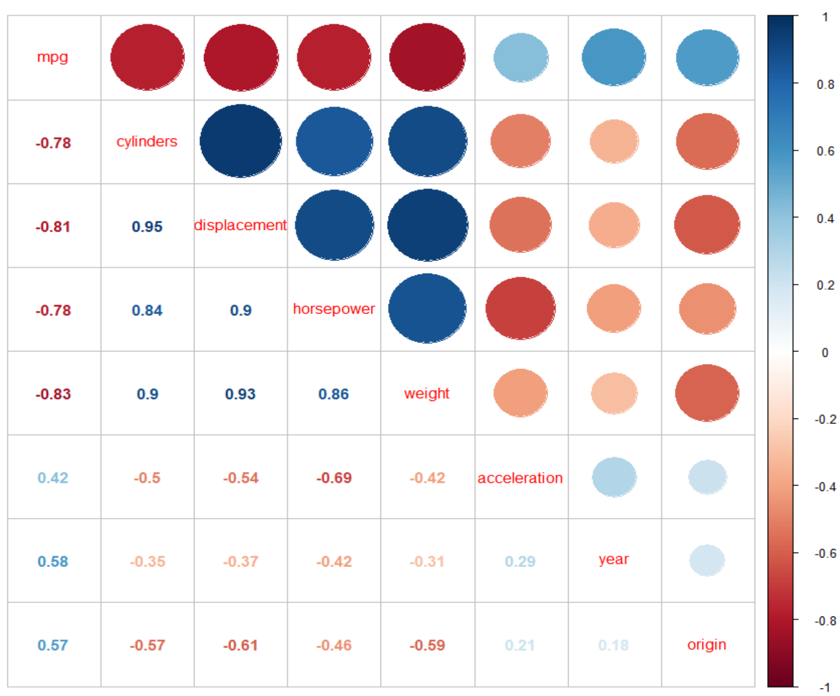
```
      mpg          cylinders      displacement     horsepower        weight      acceleration        year           origin                      name    
 Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador       :  5  
 1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225   1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto        :  5  
 Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804   Median :15.50   Median :76.00   Median :1.000   toyota corolla    :  5  
 Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978   Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin       :  4  
 3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615   3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet        :  4  
 Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140   Max.   :24.80   Max.   :82.00   Max.   :3.000   chevrolet chevette:  4  
                                                                                                                                (Other)           :365  
     mpg01    
 Min.   :0.0  
 1st Qu.:0.0  
 Median :0.5  
 Mean   :0.5  
 3rd Qu.:1.0  
 Max.   :1.0  
```

#b) This depends on the mpg01 column created above

cor(Auto[, -9])

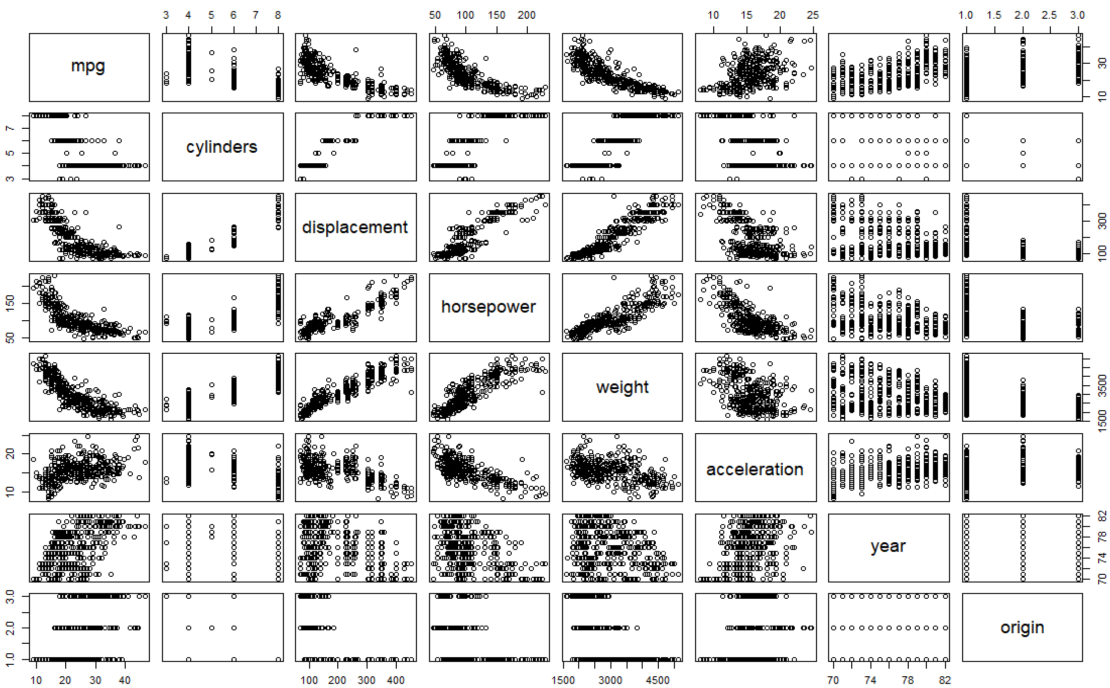
	mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	mpg01
mpg	1.0000000	-0.7776175	-0.8051269	-0.7784268	-0.8322442	0.4233285	0.5805410	0.5652088	0.8369392
cylinders	-0.7776175	1.0000000	0.9508233	0.8429834	0.8975273	-0.5046834	-0.3456474	-0.5689316	-0.7591939
displacement	-0.8051269	0.9508233	1.0000000	0.8972570	0.9329944	-0.5438005	-0.3698552	-0.6145351	-0.7534766
horsepower	-0.7784268	0.8429834	0.8972570	1.0000000	0.8645377	-0.6891955	-0.4163615	-0.4551715	-0.6670526
weight	-0.8322442	0.8975273	0.9329944	0.8645377	1.0000000	-0.4168392	-0.3091199	-0.5850054	-0.7577566
acceleration	0.4233285	-0.5046834	-0.5438005	-0.6891955	-0.4168392	1.0000000	0.2903161	0.2127458	0.3468215
year	0.5805410	-0.3456474	-0.3698552	-0.4163615	-0.3091199	0.2903161	1.0000000	0.1815277	0.4299042
origin	0.5652088	-0.5689316	-0.6145351	-0.4551715	-0.5850054	0.2127458	0.1815277	1.0000000	0.5136984
mpg01	0.8369392	-0.7591939	-0.7534766	-0.6670526	-0.7577566	0.3468215	0.4299042	0.5136984	1.0000000

library(corrplot)
corrplot::corrplot.mixed(cor(Auto[, -9]), upper="circle")



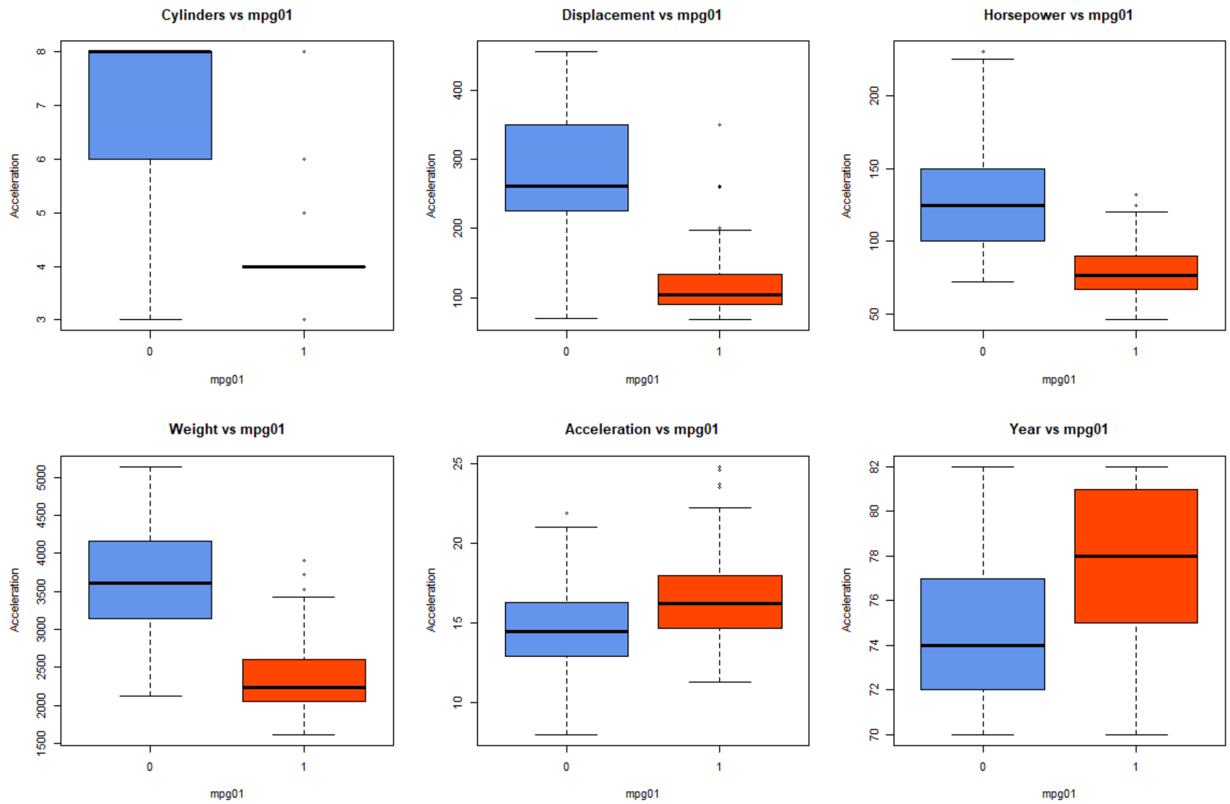
 # Scatterplot matrix

pairs(Auto[, -9])



 # BoxPlots

par(mfrow=c(2,3))
boxplot(cylinders ~ mpg01, data = Auto, main = "Cylinders vs mpg01" ,
xlab = "mpg01", ylab = "Acceleration",col = c("cornflowerblue", "orangered"))
boxplot(displacement ~ mpg01, data = Auto, main = "Displacement vs mpg01" ,
xlab = "mpg01", ylab = "Acceleration",col = c("cornflowerblue", "orangered"))
boxplot(horsepower ~ mpg01, data = Auto, main = "Horsepower vs mpg01" ,
xlab = "mpg01", ylab = "Acceleration",col = c("cornflowerblue", "orangered"))
boxplot(weight ~ mpg01, data = Auto, main = "Weight vs mpg01" , xlab = "mpg01",
ylab = "Acceleration",col = c("cornflowerblue", "orangered"))
boxplot(acceleration ~ mpg01, data = Auto, main = "Acceleration vs mpg01" ,
xlab = "mpg01", ylab = "Acceleration",col = c("cornflowerblue", "orangered"))
boxplot(year ~ mpg01, data = Auto, main = "Year vs mpg01" , xlab = "mpg01",
ylab = "Acceleration",col = c("cornflowerblue", "orangered"))



Based on the above plots, mpg01 has a negative association with Weight, Cylinders, Displacement, and Horsepower.

#c) Split the data into a training set and a test set.

```
set.seed(123)
train <- sample(1:dim(Auto)[1], dim(Auto)[1]*.7, rep=FALSE)
test <- -train
Training_dataSet<- Auto[train, ]
Testing_dataSet= Auto[test, ]
mpg01.test <- mpg01[test]
```

#d) LDA on the training data to predict mpg01 using the variables that seemed most associated with mpg01

```
library(MASS)
lda_model <- lda(mpg01 ~ cylinders + weight + displacement + horsepower, dat
a = Training_dataSet)
lda_model
```

```
Call:
lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Training_d
ataSet)

Prior probabilities of groups:
        0         1 
0.4817518 0.5182482 

Group means:
  cylinders   weight displacement horsepower
0  6.795455 3659.008     274.3333  130.88636
1  4.218310 2343.599     117.0528   79.71127

Coefficients of linear discriminants:
                       LD1
cylinders    -0.4483062756
weight       -0.0012201696
displacement  0.0001078142
horsepower    0.0046253024
```

lda_pred = predict(lda_model, Testing_dataSet)

names(lda_pred)

[1] "class"   "posterior" "x"

The confusion matrix

pred.lda <- predict(lda_model, Testing_dataSet)
table(Testing_dataSet$mpg01, pred.lda$class)
mpg01.test

```
   0  1
  0 53 11
  1  3 51
> mpg01.test
  [1] 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 1 0 0 1
0 0 1 1 0 0 0 1 1 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 0 0 0 0 1 1 0 0 1 1 0 0
0 0 0 0
 [80] 1 0 0 0 0 0 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
0 1 1
```

The Mean Value

mean(pred.lda$class != Testing_dataSet$mpg01)

```
[1] 0.1186441
```

Using all 4 predictors (cylinders, weight, displacement, and horsepower) with a Linear Discriminant Analysis the test error rate is 11.86441%*.*

#f)

```
glm_model <- glm(mpg01 ~ cylinders + weight + displacement + horsepower, d
ata = Training_dataSet, family = binomial)
summary(glm_model)
```

```
Call:
glm(formula = mpg01 ~ cylinders + weight + displacement + horsepower, 
    family = binomial, data = Training_dataSet)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.4870  -0.1763   0.1231   0.3633   3.2024  

Coefficients:
               Estimate Std. Error z value Pr(>|z|)    
(Intercept)  11.6661375  1.9885822   5.867 4.45e-09 ***
cylinders     0.0365940  0.3817545   0.096  0.92363    
weight       -0.0023706  0.0008319  -2.850  0.00438 ** 
displacement -0.0106969  0.0096956  -1.103  0.26991    
horsepower   -0.0327332  0.0154688  -2.116  0.03434 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 379.48  on 273  degrees of freedom
Residual deviance: 147.50  on 269  degrees of freedom
AIC: 157.5

Number of Fisher Scoring iterations: 7
```

```
probs <- predict(glm_model, Testing_dataSet, type = "response")
pred.glm <- rep(0, length(probs))
pred.glm[probs > 0.5] <- 1
table(pred.glm, mpg01.test)
```

```
        mpg01.test
pred.glm  0  1
       0 54  3
       1 10 51
```

```
mean(pred.glm != mpg01.test)
```

```
[1] 0.1101695
```

Using all 4 predictors (cylinders, weight, displacement, and horsepower) with a Logistic Regression Analysis the test error rate is 11.01695 *%.*

#g)

str(Auto)

```
'data.frame':   392 obs. of  10 variables:
 $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
 $ cylinders   : int  8 8 8 8 8 8 8 8 8 8 ...
 $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
 $ horsepower  : num  130 165 150 150 140 198 220 215 225 190 ...
 $ weight      : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
 $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
 $ year        : int  70 70 70 70 70 70 70 70 70 70 ...
 $ origin      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ name        : Factor w/ 304 levels "amc ambassador brougham",..: 49 36 231
14 161 141 54 223 241 2 ...
 $ mpg01       : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
data = scale(Auto[,-c(9,10)])
set.seed(1234)
train <- sample(1:dim(Auto)[1], 392*.7, rep=FALSE)
test <- -train
Training_dataSet = data[train,c("cylinders","horsepower","weight","acceleration")]
Testing_dataSet = data[test, c("cylinders", "horsepower","weight","acceleration")]
## KNN take the training response variable seperately
train.mpg01 = Auto$mpg01[train]
## we also need the have the testing_y seperately for assesing the model later on
test.mpg01= Auto$mpg01[test]
library(class)
set.seed(1234)
knn_pred_y = knn(Training_dataSet, Testing_dataSet, train.mpg01, k = 1)
table(knn_pred_y, test.mpg01)
```

```
          test.mpg01
knn_pred_y  0  1
         0 50  4
         1  6 58
```

```
mean(knn_pred_y != test.mpg01)
```

```
[1] 0.08474576
```

Test error rate is 8.474576 % for K=1.

```
knn_pred_y = NULL
error_rate = NULL
for(i in 1:dim(Testing_dataSet)[1]){
  set.seed(1234)
  knn_pred_y = knn(Training_dataSet,Testing_dataSet,train.mpg01,k=i)
  error_rate[i] = mean(test.mpg01 != knn_pred_y)
}
### find the minimum error rate
min_error_rate = min(error_rate)
print(min_error_rate)
```

```
[1] 0.06779661
```

K with lowest values

```
K = which(error_rate == min_error_rate)
print(K)
```

```
[1]  4
```

When we train a KNN model with k=4 then we get the lowest misclassification error rate of 6.779661%.

```
library(ggplot2)
qplot(1:dim(Testing_dataSet)[1], error_rate, xlab = "K", ylab = "Error Rate",
geom=c("point", "line"))
```