R Notebook

This question should be answered using the Weekly data set, which is part of the ISLR package. This data is similar in nature to the Smarket data from this chapter's lab, except that it contains 1,089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

```r
suppressMessages(library(ISLR)); data(Weekly)
suppressMessages(library('topicmodels'))
suppressMessages(library(MASS))
new_Weekly <- Weekly
summary(new_Weekly)
```

```
##       Year           Lag1               Lag2               Lag3
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260
##       Lag4               Lag5              Volume
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202
##  Median :  0.2380   Median :  0.2340   Median :1.00268
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821
##      Today          Direction
##  Min.   :-18.1950   Down:484
##  1st Qu.: -1.1540   Up  :605
##  Median :  0.2410
##  Mean   :  0.1499
##  3rd Qu.:  1.4050
##  Max.   : 12.0260
```

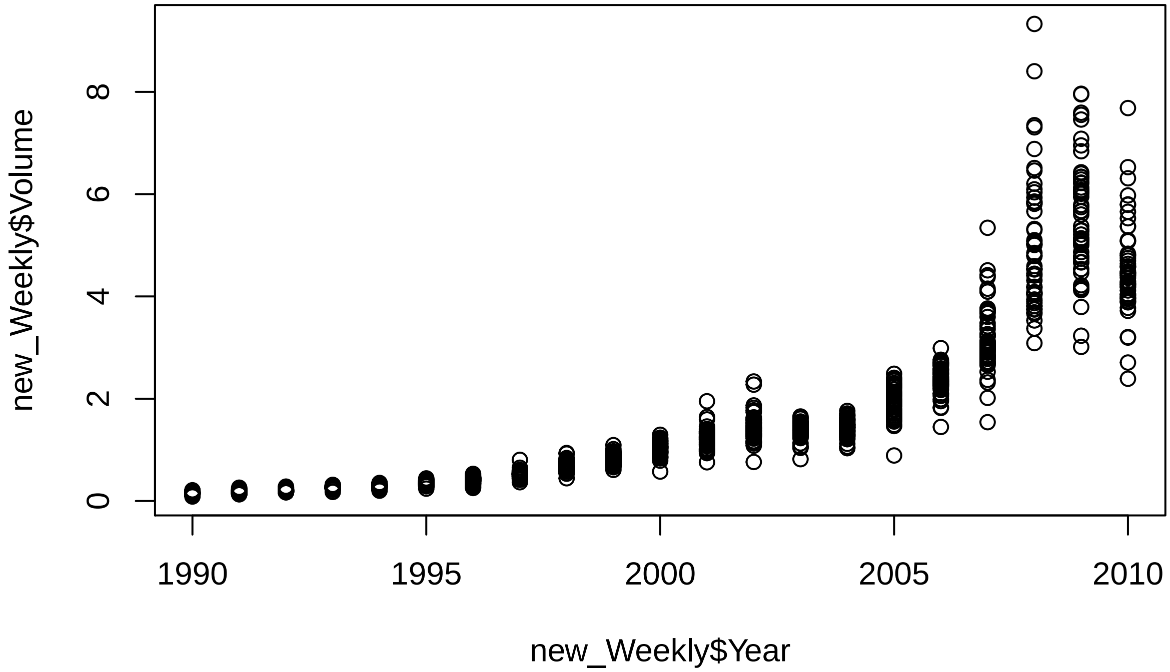
(a) Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?

```r
summary(new_Weekly)
```

```
##       Year           Lag1               Lag2               Lag3
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260
##       Lag4               Lag5              Volume
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202
##  Median :  0.2380   Median :  0.2340   Median :1.00268
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821
##      Today          Direction
```

```
##  Min.   :-18.1950   Down:484
##  1st Qu.: -1.1540   Up  :605
##  Median :  0.2410
##  Mean   :  0.1499
##  3rd Qu.:  1.4050
##  Max.   : 12.0260
```

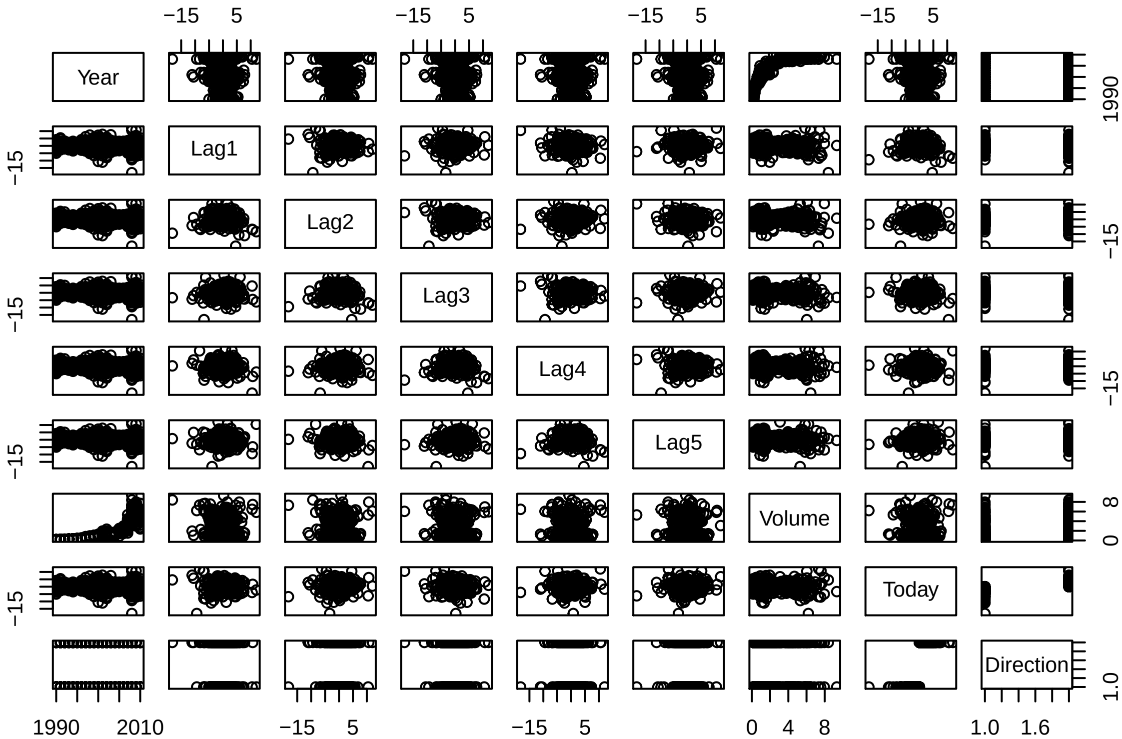
```r
c <- plot(x=new_Weekly$Year, y=new_Weekly$Volume)
```




```r
c
```

```
## NULL
```

```r
#new_Weekly$Direction <- unclass(Weekly$Direction)
pairs(new_Weekly)
```



Observations: * Volume has increased significantly over the years, it seems like the strongest relationship between variables. * Today and direction seem to follow a so * There is no stand out relationship between lags and anything at all.

(b) Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

```
convert_to_binomial <- function(x) {
    if (x == 'Up'){
      return(1)
    } else if (x == 'Down'){
      return(0)
    }
}
new_Weekly$Direction <- sapply(X=new_Weekly$Direction, FUN = convert_to_binomial)
model_1b <- glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,family=binomial(link='logit'),data=new_Weekly
summary(model_1b)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial(link = "logit"), data = new_Weekly)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
```

```
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

Looks like the model is poor fit. Lag 2 seems to be the only close to significant predictor but thats not even with a 0.05 alpha level.

(c) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

```r
#suppressMessages(install.packages("caret", dependencies = c("Depends", "Suggests")))
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```r
c_pre2 <- predict(model_1b, type="response")
class_prediction <-
  ifelse(c_pre2 > 0.50,
         1,
         0
  )
confusionMatrix(data=factor(class_prediction),reference=factor(new_Weekly$Direction))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  54  48
##          1 430 557
##
##                Accuracy : 0.5611
##                  95% CI : (0.531, 0.5908)
##     No Information Rate : 0.5556
##     P-Value [Acc > NIR] : 0.369
##
##                   Kappa : 0.035
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.11157
##             Specificity : 0.92066
##          Pos Pred Value : 0.52941
##          Neg Pred Value : 0.56434
```

```
##               Prevalence : 0.44444
##           Detection Rate : 0.04959
##     Detection Prevalence : 0.09366
##        Balanced Accuracy : 0.51612
##
##         'Positive' Class : 0
##
```

This model seems very inclined to predict an upward movement.

(d) Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag2 as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 and 2010).

```r
q1d_train_slice <- subset(new_Weekly, Year>1989 & Year<2009)
q1d_test_slice <- subset(new_Weekly, Year>2008)
model_1d <- glm(Direction~Lag2,family=binomial(link='logit'),data=q1d_train_slice)

c_pre3<-predict(model_1d, newdata=q1d_test_slice, type="response")
class_prediction_3 <-
  ifelse(c_pre3 > 0.50,
         1,
         0
  )

confusionMatrix(data=factor(class_prediction_3),reference=factor(q1d_test_slice$Direction))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0  9  5
##          1 34 56
##
##                Accuracy : 0.625
##                  95% CI : (0.5247, 0.718)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.2439
##
##                   Kappa : 0.1414
##
##  Mcnemar's Test P-Value : 7.34e-06
##
##             Sensitivity : 0.20930
##             Specificity : 0.91803
##          Pos Pred Value : 0.64286
##          Neg Pred Value : 0.62222
##              Prevalence : 0.41346
##          Detection Rate : 0.08654
##    Detection Prevalence : 0.13462
##       Balanced Accuracy : 0.56367
##
##        'Positive' Class : 0
##
```

(e) Repeat (d) using LDA.

```r
model_1e <- lda(Direction~Lag2,data=q1d_train_slice)

c_pre_q1e<-predict(model_1e, newdata=q1d_test_slice)
class_prediction_4 <-
  ifelse(c_pre_q1e[["x"]] > 0.50,
         1,
         0
  )

confusionMatrix(data=c_pre_q1e$class,reference=factor(q1d_test_slice$Direction))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0  9  5
##          1 34 56
##
##                Accuracy : 0.625
##                  95% CI : (0.5247, 0.718)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.2439
##
##                   Kappa : 0.1414
##
##  Mcnemar's Test P-Value : 7.34e-06
##
##             Sensitivity : 0.20930
##             Specificity : 0.91803
##          Pos Pred Value : 0.64286
##          Neg Pred Value : 0.62222
##              Prevalence : 0.41346
##          Detection Rate : 0.08654
##    Detection Prevalence : 0.13462
##       Balanced Accuracy : 0.56367
##
##        'Positive' Class : 0
##
```

(f) Repeat (d) using QDA.

```r
model_1f <- qda(Direction~Lag2,data=q1d_train_slice)

c_pre_q1f<-predict(model_1f, newdata=q1d_test_slice)


confusionMatrix(data=c_pre_q1f$class,reference=factor(q1d_test_slice$Direction))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0  0  0
##          1 43 61
##
```

```
##                Accuracy : 0.5865
##                  95% CI : (0.4858, 0.6823)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.5419
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 1.504e-10
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.5865
##              Prevalence : 0.4135
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

(g) Repeat (d) using KNN with K = 1.

```
library(class)
train.X <- matrix(q1d_train_slice$Lag2)
test.X <- matrix(q1d_test_slice$Lag2)
train.direction <- q1d_train_slice$Direction
test.direction <- q1d_test_slice$Direction

set.seed (1)
knn.pred=knn(train=train.X,test=test.X,cl=train.direction ,k=1)
confusionMatrix(data=knn.pred,reference=factor(q1d_test_slice$Direction))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 21 30
##          1 22 31
##
##                Accuracy : 0.5
##                  95% CI : (0.4003, 0.5997)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.9700
##
##                   Kappa : -0.0033
##
##  Mcnemar's Test P-Value : 0.3317
##
##             Sensitivity : 0.4884
##             Specificity : 0.5082
##          Pos Pred Value : 0.4118
##          Neg Pred Value : 0.5849
##              Prevalence : 0.4135
##          Detection Rate : 0.2019
```

```
##    Detection Prevalence : 0.4904
##       Balanced Accuracy : 0.4983
##
##        'Positive' Class : 0
##
```

(h) Which of these methods appears to provide the best results on this data? It seems like LDA and Logistic regression provide the exact same results with an accuracy of 62%. I would just one of these since they have the greatest accuracy of all the models.

(i) Experiment with different combinations of predictors, including possible transformations and interactions, for each of the methods. Report the variables, method, and associated confusion matrix that appears to provide the best results on the held out data. Note that you should also experiment with values for K in the KNN classifier.

Question #2

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. 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.

```r
auto <- data.frame(Auto)
median.mpg <- median(auto$mpg)
auto$mpg01 <-
  ifelse(auto$mpg > median.mpg,
         1,
         0
  )
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          mpg01
##  amc matador       :  5   Min.   :0.0
##  ford pinto        :  5   1st Qu.:0.0
##  toyota corolla    :  5   Median :0.5
##  amc gremlin       :  4   Mean   :0.5
##  amc hornet        :  4   3rd Qu.:1.0
```

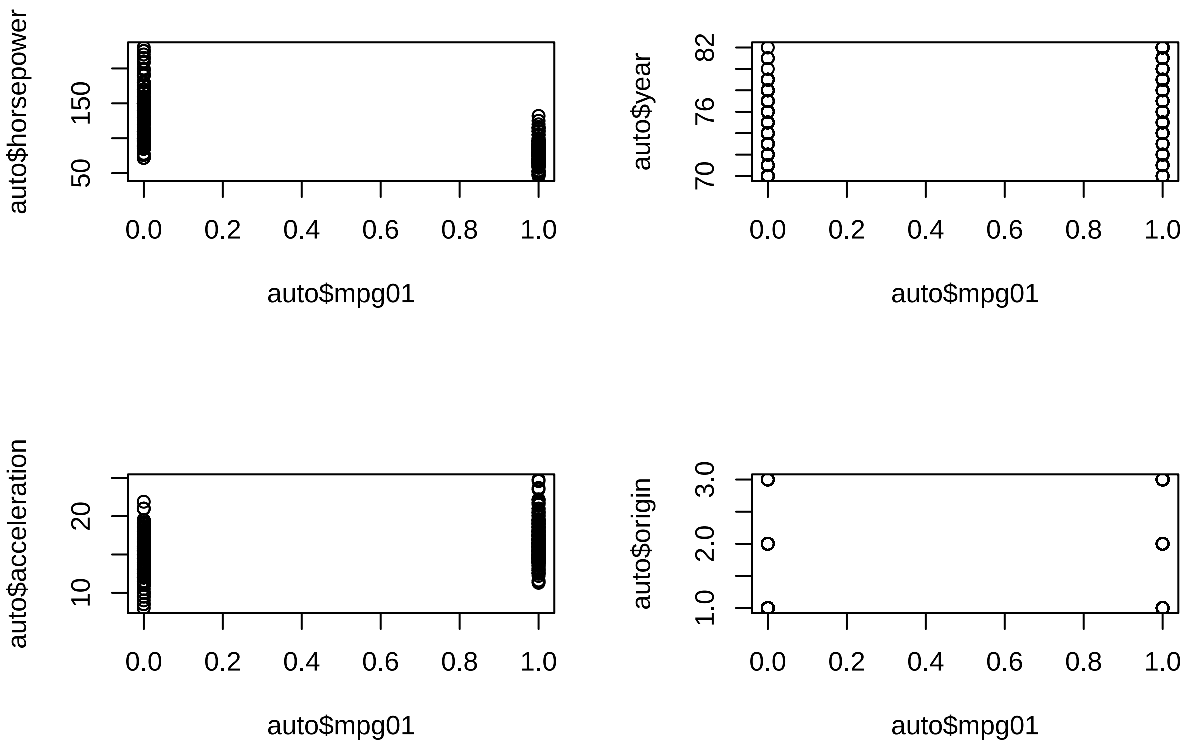
```
##   chevrolet chevette:  4   Max.   :1.0
##   (Other)           :365
```

(b) Explore the data graphically in order to investigate the association between mpg01 and the other features. Which of the other features seem most likely to be useful in predicting mpg01? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.

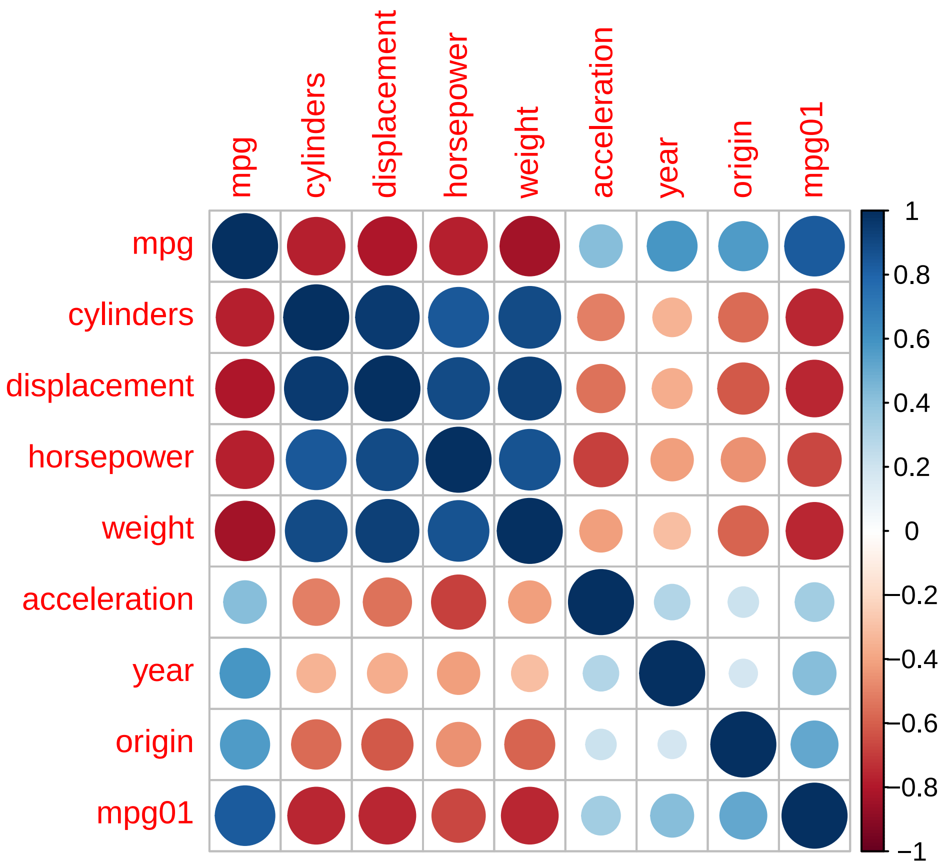
```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
par(mfrow=c(2,2))
plot(auto$mpg01, auto$horsepower)
plot(auto$mpg01, auto$year)
plot(auto$mpg01, auto$acceleration)
plot(auto$mpg01, auto$origin)
```




```
par(mfrow=c(1,1))
auto.numeric <- auto[,sapply(auto, is.numeric)]
M <- cor(x = as.matrix(auto.numeric))
corrplot(M, method = "circle")
```



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

```
smp_size <- floor(0.75 * nrow(auto))
set.seed(101)

train_ind <- sample(seq_len(nrow(auto)), size = smp_size)

train.auto <- auto[train_ind, ]
test.auto <- auto[-train_ind, ]
```

(d) Perform LDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

```
model_2d <- lda(mpg01 ~ cylinders + weight + displacement + horsepower,data=train.auto)

c_pre_2d<-predict(model_2d, newdata=test.auto)


confusionMatrix(data=c_pre_2d$class,reference=factor(test.auto$mpg01))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 46  4
##          1  5 43
##
##                Accuracy : 0.9082
```

```
##                  95% CI : (0.8328, 0.9571)
##     No Information Rate : 0.5204
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8162
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9020
##             Specificity : 0.9149
##          Pos Pred Value : 0.9200
##          Neg Pred Value : 0.8958
##              Prevalence : 0.5204
##          Detection Rate : 0.4694
##    Detection Prevalence : 0.5102
##       Balanced Accuracy : 0.9084
##
##        'Positive' Class : 0
##
```

```r
## Test Error Rate
print('This is the test error rate of the LDA Model: ')
```

```
## [1] "This is the test error rate of the LDA Model: "
```

```r
mean(c_pre_2d$class != test.auto$mpg01)
```

```
## [1] 0.09183673
```

(e) Perform QDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

```r
model_2e <- qda(mpg01~origin+year+acceleration,data=train.auto)

c_pre_2e<-predict(model_2e, newdata=test.auto)


confusionMatrix(data=c_pre_2e$class,reference=factor(test.auto$mpg01))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 43 13
##          1  8 34
##
##                Accuracy : 0.7857
##                  95% CI : (0.6913, 0.8622)
##     No Information Rate : 0.5204
##     P-Value [Acc > NIR] : 5.092e-08
##
##                   Kappa : 0.5689
##
##  Mcnemar's Test P-Value : 0.3827
##
##             Sensitivity : 0.8431
##             Specificity : 0.7234
```

```
##          Pos Pred Value : 0.7679
##          Neg Pred Value : 0.8095
##              Prevalence : 0.5204
##          Detection Rate : 0.4388
##    Detection Prevalence : 0.5714
##       Balanced Accuracy : 0.7833
##
##        'Positive' Class : 0
##
```

```
## Test Error Rate
print('This is the test error rate of the QDA Model: ')
```

```
## [1] "This is the test error rate of the QDA Model: "
```

```
mean(c_pre_2e$class != test.auto$mpg01)
```

```
## [1] 0.2142857
```

(f) Perform logistic regression on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

```
model_2f <- glm(mpg01~origin+year+acceleration,data=train.auto,family=binomial(link='logit'))

c_pre_2f<-predict(model_2f, newdata=test.auto, type="response")
class_prediction_2f <-
  ifelse(c_pre_2f > 0.50,
         1,
         0
  )

confusionMatrix(data=factor(class_prediction_2f),reference=factor(test.auto$mpg01))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 40 15
##          1 11 32
##
##                Accuracy : 0.7347
##                  95% CI : (0.6359, 0.8188)
##     No Information Rate : 0.5204
##     P-Value [Acc > NIR] : 1.163e-05
##
##                   Kappa : 0.4667
##
##  Mcnemar's Test P-Value : 0.5563
##
##             Sensitivity : 0.7843
##             Specificity : 0.6809
##          Pos Pred Value : 0.7273
##          Neg Pred Value : 0.7442
##              Prevalence : 0.5204
##          Detection Rate : 0.4082
##    Detection Prevalence : 0.5612
##       Balanced Accuracy : 0.7326
```

```
##
##        'Positive' Class : 0
##
```

```r
## Test Error Rate
print('This is the test error rate of the Logistic Model: ')
```

```
## [1] "This is the test error rate of the Logistic Model: "
```

```r
sum((factor(class_prediction_2f) == factor(test.auto$mpg01)), na.rm = TRUE) / length(c_pre_2d$class)
```

```
## [1] 0.7346939
```

(g) Perform KNN on the training data, with several values of K, in order to predict mpg01. Use only the variables that seemed most associated with mpg01 in (b). What test errors do you obtain? Which value of K seems to perform the best on this data set?

```r
library(class)
train.autoX <- (cbind(train.auto$origin, train.auto$year, train.auto$acceleration))
test.autoX <- (cbind(test.auto$origin, test.auto$year, test.auto$acceleration))
train.autoy <- train.auto$mpg01
test.autoy <- test.auto$mpg01

set.seed (1)
knn.pred=knn(train=train.autoX,test=test.autoX,cl=train.autoy ,k=1)


confusionMatrix(data=knn.pred,reference=factor(test.autoy))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 33 12
##          1 18 35
##
##                Accuracy : 0.6939
##                  95% CI : (0.5926, 0.783)
##     No Information Rate : 0.5204
##     P-Value [Acc > NIR] : 0.0003607
##
##                   Kappa : 0.3898
##
##  Mcnemar's Test P-Value : 0.3613104
##
##             Sensitivity : 0.6471
##             Specificity : 0.7447
##          Pos Pred Value : 0.7333
##          Neg Pred Value : 0.6604
##              Prevalence : 0.5204
##          Detection Rate : 0.3367
##    Detection Prevalence : 0.4592
##       Balanced Accuracy : 0.6959
##
##        'Positive' Class : 0
##
```

```r
## Test Error Rate k = 1
print('This is the test error rate of the KNN Model with k = 1: ')
```

```
## [1] "This is the test error rate of the KNN Model with k = 1: "
```

```r
sum((knn.pred == factor(test.autoy)), na.rm = TRUE) / length(test.autoy)
```

```
## [1] 0.6938776
```

```r
## Test Error Rate k = 2
knn.pred=knn(train=train.autoX,test=test.autoX,cl=train.autoy ,k=2)
print('This is the test error rate of the KNN Model with k = 2: ')
```

```
## [1] "This is the test error rate of the KNN Model with k = 2: "
```

```r
sum((knn.pred == factor(test.autoy)), na.rm = TRUE) / length(test.autoy)
```

```
## [1] 0.7346939
```

```r
## Test Error Rate k = 3
knn.pred=knn(train=train.autoX,test=test.autoX,cl=train.autoy ,k=3)
print('This is the test error rate of the KNN Model with k = 3: ')
```

```
## [1] "This is the test error rate of the KNN Model with k = 3: "
```

```r
sum((knn.pred == factor(test.autoy)), na.rm = TRUE) / length(test.autoy)
```

```
## [1] 0.7346939
```

```r
## Test Error Rate k = 4
knn.pred=knn(train=train.autoX,test=test.autoX,cl=train.autoy ,k=4)
print('This is the test error rate of the KNN Model with k = 4: ')
```

```
## [1] "This is the test error rate of the KNN Model with k = 4: "
```

```r
sum((knn.pred == factor(test.autoy)), na.rm = TRUE) / length(test.autoy)
```

```
## [1] 0.7040816
```

```r
print("Best performing K seems to be 2 and 3 at ~73% error rate.")
```

```
## [1] "Best performing K seems to be 2 and 3 at ~73% error rate."
```

```r
knn.pred=knn(train=train.autoX,test=test.autoX,cl=train.autoy ,k=2)
```

Question #3

Find a R package that can perform Naïve Bayesian analysis and use it to do Q1 (part d) and Q2 (part d).

```r
library(naivebayes)
```

```
## naivebayes 0.9.6 loaded
```

```r
# Naive Bayes to do Question 2 part d
model_3a <- naive_bayes(x = train.autoX, y = factor(train.autoy))

c_pre_3a<-predict(model_3a, newdata=test.autoX, type = "class")


confusionMatrix(data=c_pre_3a,reference=factor(test.autoy))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 43 16
##          1  8 31
##
##                Accuracy : 0.7551
##                  95% CI : (0.6579, 0.8364)
##     No Information Rate : 0.5204
##     P-Value [Acc > NIR] : 1.555e-06
##
##                   Kappa : 0.5061
##
##  Mcnemar's Test P-Value : 0.153
##
##             Sensitivity : 0.8431
##             Specificity : 0.6596
##          Pos Pred Value : 0.7288
##          Neg Pred Value : 0.7949
##              Prevalence : 0.5204
##          Detection Rate : 0.4388
##    Detection Prevalence : 0.6020
##       Balanced Accuracy : 0.7514
##
##        'Positive' Class : 0
##
```

```r
## Test Error Rate
print('This is the test error rate of the Naive Bayes Model: ')
```

```
## [1] "This is the test error rate of the Naive Bayes Model: "
```

```r
sum((c_pre_3a == factor(test.auto$mpg01)), na.rm = TRUE) / length(c_pre_3a)
```

```
## [1] 0.755102
```

```r
# Naive Bayes to do Question 1 D
train.X <- matrix(q1d_train_slice$Lag2)
test.X <- matrix(q1d_test_slice$Lag2)
train.direction <- q1d_train_slice$Direction
test.direction <- q1d_test_slice$Direction



model_3b <- naive_bayes(x = train.X, y = factor(train.direction))

c_pre_3b<-predict(model_3b, test.X, type = "class")


confusionMatrix(data=c_pre_3b,reference=factor(test.direction))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
```

```
##          0  0  0
##          1 43 61
##
##                Accuracy : 0.5865
##                  95% CI : (0.4858, 0.6823)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.5419
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 1.504e-10
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.5865
##              Prevalence : 0.4135
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

```r
## Test Error Rate
print('This is the test error rate of the Naive Bayes Model: ')
```

```
## [1] "This is the test error rate of the Naive Bayes Model: "
```

```r
sum((c_pre_3b == factor(test.direction)), na.rm = TRUE) / length(c_pre_3b)
```

```
## [1] 0.5865385
```

Question #4

Find a R package that can generate ROC curve. Use it to compare different models (LDA, QDA, logistic regression, KNN, naïve Bayesian) in Q1, Q2 and Q3.

```r
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
```

```
##     cov, smooth, var
```

```r
par(mfrow=c(2,3))
plot(roc(test.autoy, as.numeric(c_pre_3a)),main = "Naive Bayes Model Q2")
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
plot(roc(test.direction, as.numeric(c_pre_3b)),main = "Naive Bayes Model Q1")
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
plot(roc(test.autoy, as.numeric(knn.pred)),main = "KNN (k=2) Model Q2")
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
plot(roc(test.autoy, as.numeric(c_pre_2e$class)),main = "QDA Model Q2")
```

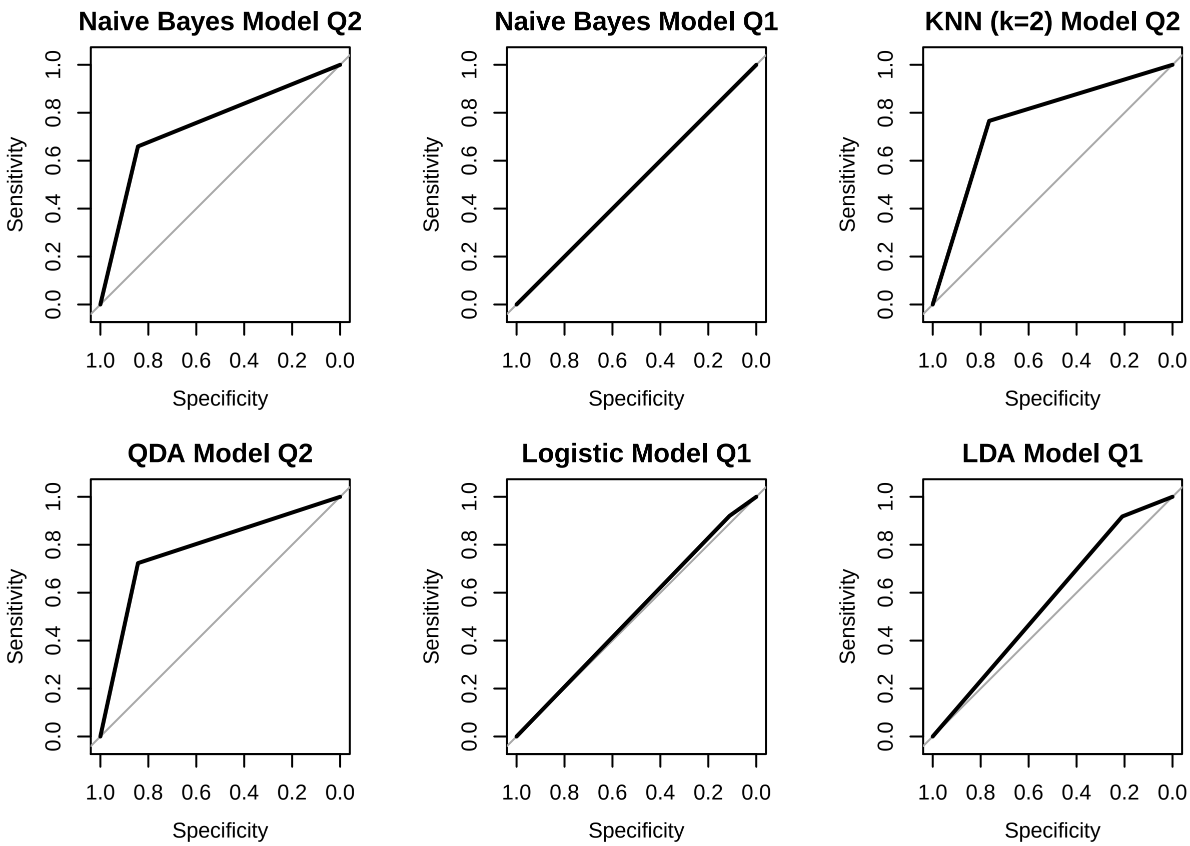
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
plot(roc(new_Weekly$Direction, as.numeric(class_prediction)),main = "Logistic Model Q1")
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
plot(roc(q1d_test_slice$Direction, as.numeric(c_pre_q1e$class)),main = "LDA Model Q1")
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```


Naive Bayes Model Q2
Naive Bayes Model Q1
KNN (k=2) Model Q2
QDA Model Q2
Logistic Model Q1
LDA Model Q1
Sensitivity
Specificity
0.0 0.2 0.4 0.6 0.8 1.0
1.0 0.8 0.6 0.4 0.2 0.0