Homework_3

Akeem Ajede

10/17/2019

Question 10

(a)

```
## Numerical Summary
library(ISLR)
dim(Weekly)
```

```
## [1] 1089    9
```

```
str(Weekly)
```

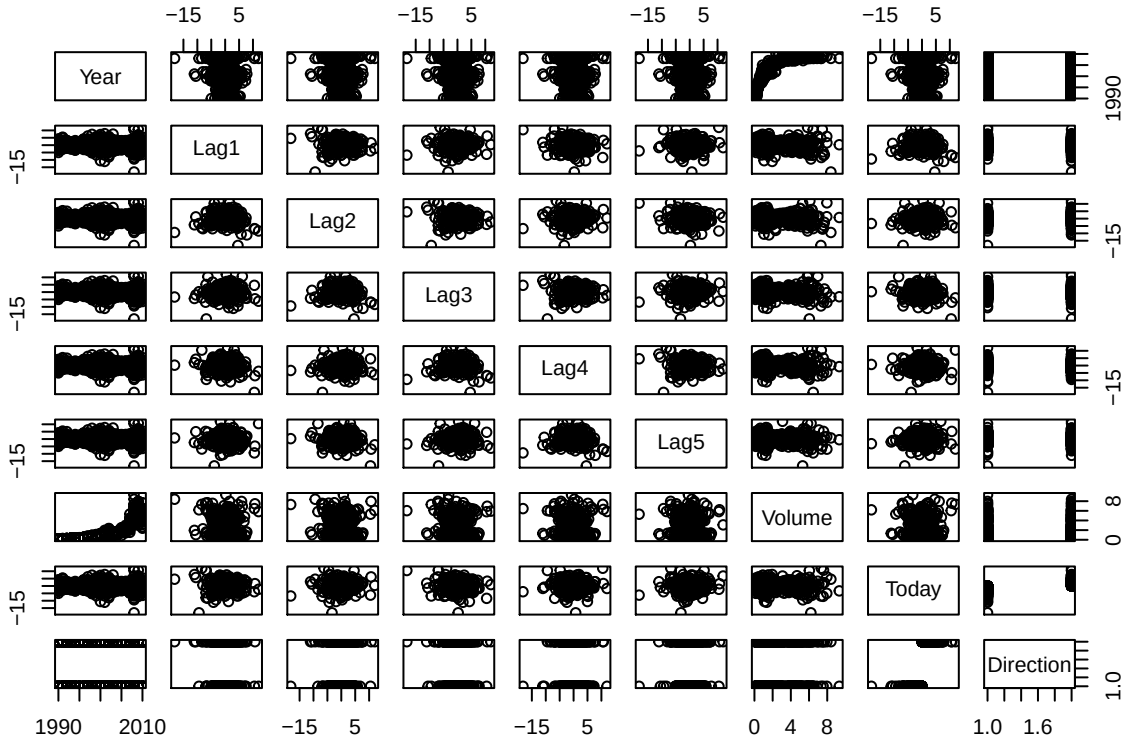
```
## 'data.frame':    1089 obs. of  9 variables:
##  $ Year     : num  1990 1990 1990 1990 1990 1990 1990 1990 1990 1990 ...
##  $ Lag1     : num  0.816 -0.27 -2.576 3.514 0.712 ...
##  $ Lag2     : num  1.572 0.816 -0.27 -2.576 3.514 ...
##  $ Lag3     : num  -3.936 1.572 0.816 -0.27 -2.576 ...
##  $ Lag4     : num  -0.229 -3.936 1.572 0.816 -0.27 ...
##  $ Lag5     : num  -3.484 -0.229 -3.936 1.572 0.816 ...
##  $ Volume   : num  0.155 0.149 0.16 0.162 0.154 ...
##  $ Today    : num  -0.27 -2.576 3.514 0.712 1.178 ...
##  $ Direction: Factor w/ 2 levels "Down","Up": 1 1 2 2 2 1 2 2 2 1 ...
```

```
summary(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
```

```
## Graphical Summary
pairs(Weekly)
```




```
cor(Weekly[,1:8]) #There is a strong positive relationship between "Year" and "Volume". lag2 and Lag5 s
```

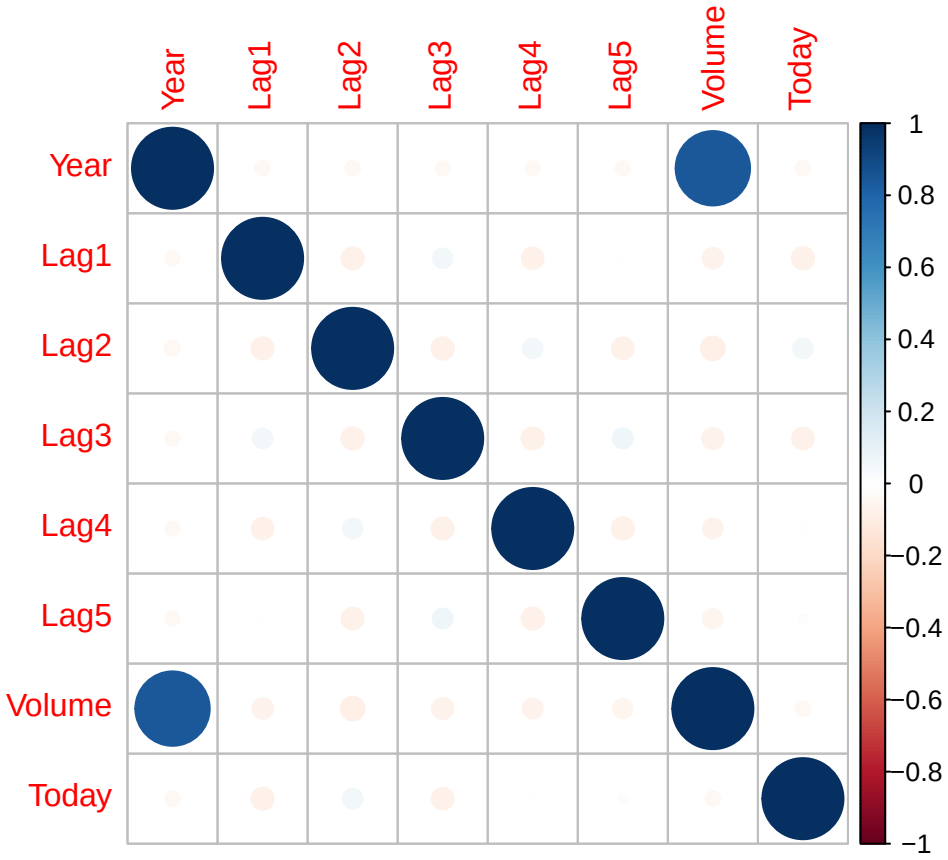
```
##              Year         Lag1        Lag2        Lag3         Lag4
## Year    1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923
## Lag1   -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876
## Lag2   -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535
## Lag3   -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865
## Lag4   -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000
## Lag5   -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027
## Volume  0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617
## Today  -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873
##                Lag5      Volume        Today
## Year   -0.030519101  0.84194162 -0.032459894
## Lag1   -0.008183096 -0.06495131 -0.075031842
## Lag2   -0.072499482 -0.08551314  0.059166717
## Lag3    0.060657175 -0.06928771 -0.071243639
## Lag4   -0.075675027 -0.06107462 -0.007825873
## Lag5    1.000000000 -0.05851741  0.011012698
```

```
## Volume -0.058517414  1.00000000 -0.033077783
## Today   0.011012698 -0.03307778  1.000000000
```

```
library(corrplot)
```

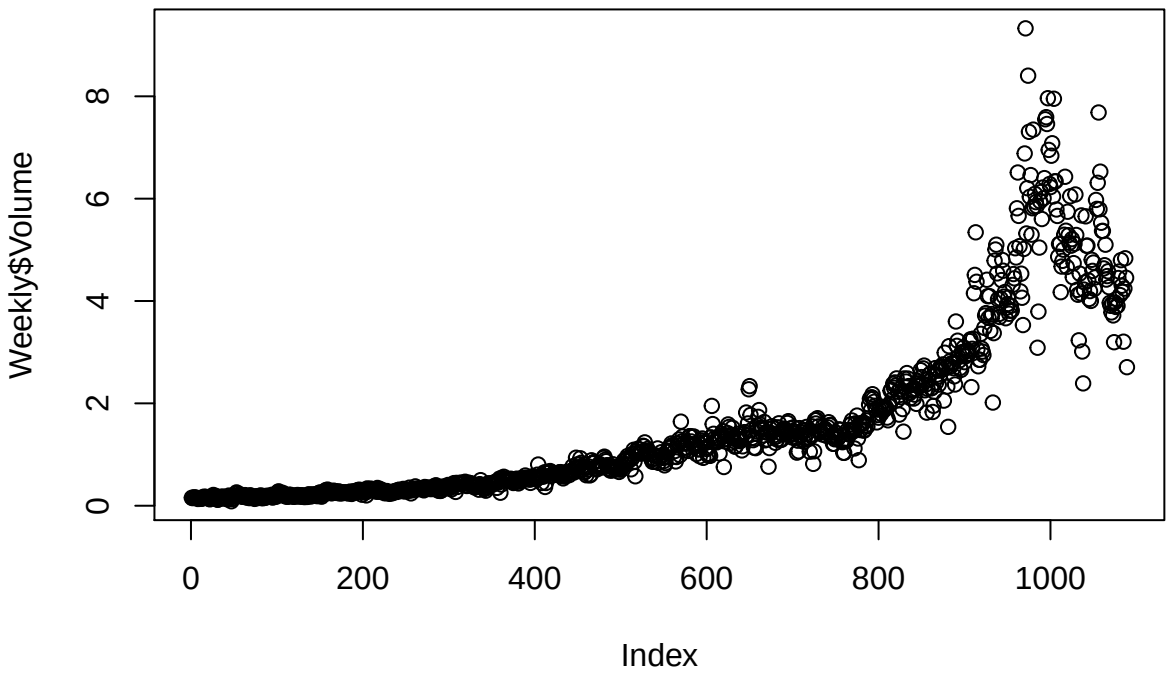
```
## corrplot 0.84 loaded
```

```
corrplot(cor(Weekly[,1:8]))
```

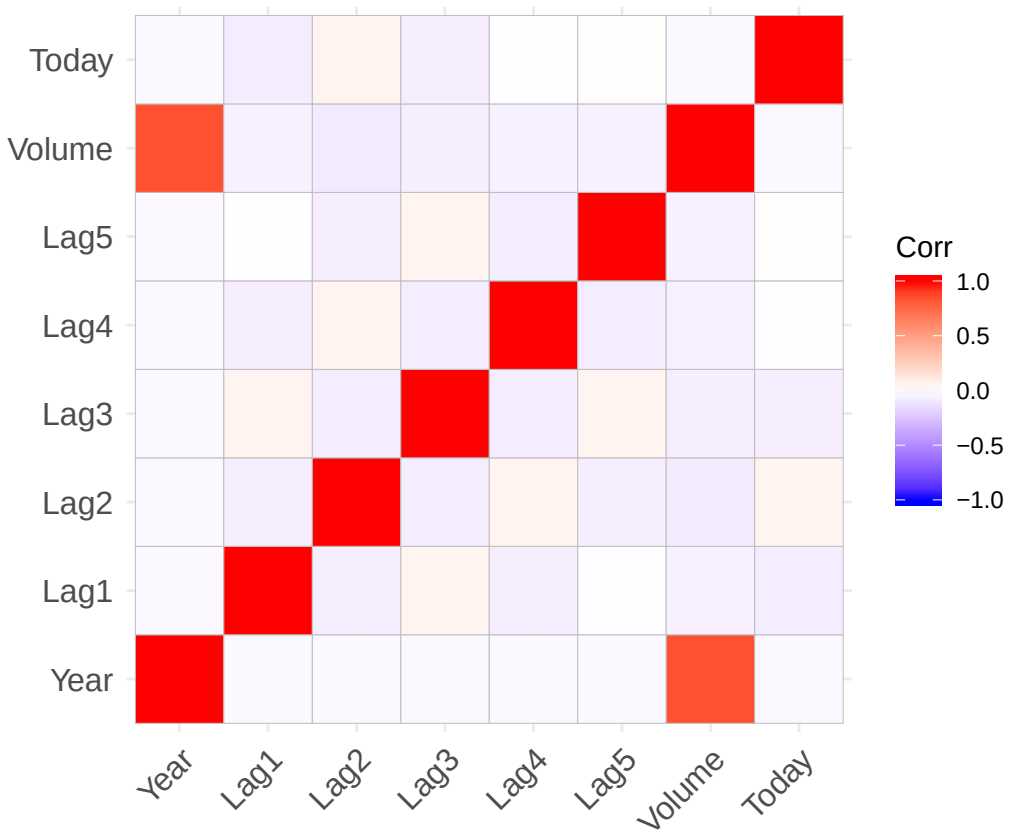



```
plot(Weekly$Volume) #Corroborates the previous deduction about the "Year" and "Volume" correlation.
library(ggcorrplot)
```

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




```
ggcorrplot(cor(Weekly[,1:8])) #Another approach to the dataset visualization
```



(b)

```
attach(Weekly)
glm.fit = glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume, data = Weekly,
              family = binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = 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
```

```
coef(glm.fit)
```

```
## (Intercept)        Lag1        Lag2        Lag3        Lag4        Lag5
##  0.26686414 -0.04126894  0.05844168 -0.01606114 -0.02779021 -0.01447206
##      Volume
## -0.02274153
```

Based on the p-value, only Lag2 is statistically signigicant.

(c)

```
glm.prob = predict(glm.fit, type = "response")
glm.prob[1:12] #Predicts P(Y=1|X), where 2 is the dummy variable for a rise in the stock market.
```

```
##         1         2         3         4         5         6         7
## 0.6086249 0.6010314 0.5875699 0.4816416 0.6169013 0.5684190 0.5786097
##         8         9        10        11        12
## 0.5151972 0.5715200 0.5554287 0.6092096 0.5370125
```

```
contrasts(Direction)
```

```
##      Up
## Down  0
## Up    1
```

```
glm.pred = rep("Down", 1089)
glm.pred[glm.prob>0.5]= "Up"
table(glm.pred, Direction)
```

```
##         Direction
## glm.pred Down  Up
##     Down   54  48
##     Up    430 557
```

```r
mean(glm.pred==Direction)
```

```
## [1] 0.5610652
```

The confusion matrix presents the wrongly classified data off the "left to right" diagonal matrix. In this scenario, the misclassified observations are 48 and 430, which sums up to 478 observations. However, the model accuracy (i.e., approx. 56.1%) does a slightly better job than a random guess (i.e., 50.0%).

(d)

```r
train = (Year<2009)
Weekly.9.10 = Weekly[!train,]
dim(Weekly.9.10)
```

```
## [1] 104   9
```

```r
Direction.9.10 = Direction[!train]

glm.fits = glm(Direction~Lag2, data = Weekly,
               family = binomial, subset = train)
glm.probs = predict(glm.fits, Weekly.9.10, type = "response")
glm.preds = rep("Down", 104)
glm.preds[glm.probs>.5]="Up"
table(glm.preds, Direction.9.10)
```

```
##          Direction.9.10
## glm.preds Down Up
##      Down    9  5
##      Up     34 56
```

```r
mean(glm.preds==Direction.9.10) #model accuracy
```

```
## [1] 0.625
```

```r
mean(glm.preds!= Direction.9.10) #Test error
```

```
## [1] 0.375
```

The overall model accuracy is 62.5%, and the test error is 37.5%. By utilizing the only significant predictor (i.e., Lag2) in the logistic regression model, model accuracy increased by approx. 6.5%.

The confusion matrix suggests that the accuracy of prediction of the upward and downward trend in the stock market are approx. 92% and 21%, respectively.

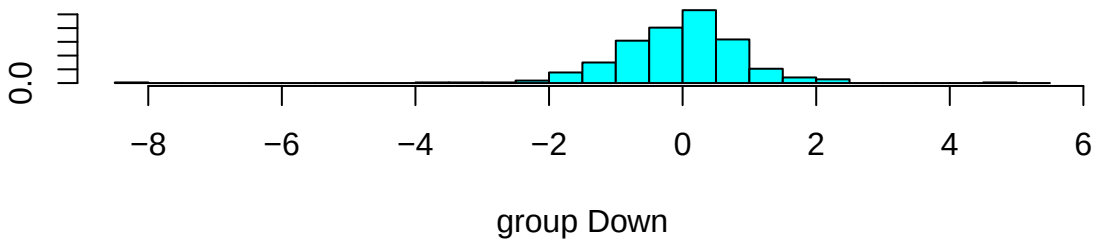
(e)

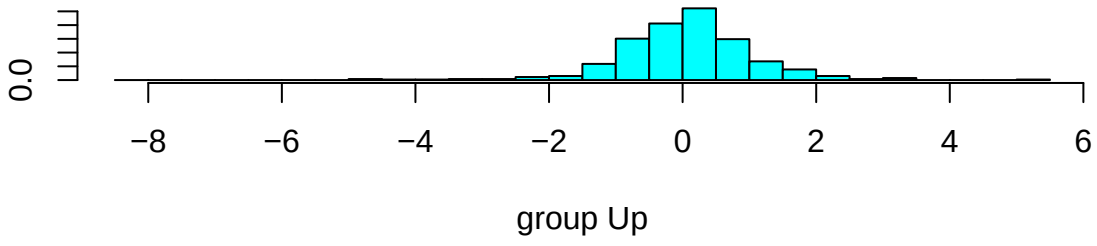
```r
library(MASS)

lda.fit = lda(Direction~Lag2, data = Weekly, subset=train)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag2, data = Weekly, subset = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##              Lag2
## Down -0.03568254
## Up    0.26036581
##
## Coefficients of linear discriminants:
##            LD1
## Lag2 0.4414162
```

```r
plot(lda.fit)
```





```
lda.pred = predict(lda.fit, Weekly.9.10)
names(lda.pred)
```

```
## [1] "class"     "posterior" "x"
```

```
lda.class = lda.pred$class
table(lda.class, Direction.9.10)
```

```
##          Direction.9.10
## lda.class Down Up
##      Down    9  5
##      Up     34 56
```

```
mean(lda.class==Direction.9.10)
```

```
## [1] 0.625
```

Model accuracy is 62.5%, which is identical with the result obtained in the Logistic Regression that considered only Lag2 as the predictor.

(f)

```
qda.fit = qda(Direction~Lag2, data = Weekly, subset = train)
qda.fit
```

```
## Call:
## qda(Direction ~ Lag2, data = Weekly, subset = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
```

```
qda.pred = predict(qda.fit, Weekly.9.10)
qda.class = qda.pred$class
table(qda.class,Direction.9.10)
```

```
##          Direction.9.10
## qda.class Down Up
##      Down    0  0
##      Up     43 61
```

```r
mean(qda.class==Direction.9.10)
```

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

The confusion matrix suggest that the model is 100% accurate in predicting market rise, but completely inaccurate (i.e., 0%) when predicting market fall.

Model accuracy is approx 58.7%.

(g)

```r
library(class)
train.X= as.matrix(Lag2[train])
test.X= as.matrix(Lag2[!train])
train.Direction=Direction[train]
set.seed(7)
knn.pred=knn(train.X,test.X,train.Direction,k=1)
table(knn.pred,Direction.9.10)
```

```
##         Direction.9.10
## knn.pred Down Up
##     Down   21 30
##     Up     22 31
```

```r
mean(knn.pred==Direction.9.10)
```

```
## [1] 0.5
```

The confusion matrix suggest that the model is approx 50.8% and 48.8% accurate in predicting the market rise and fall.

Model accuracy is approx. 50%.

(h)

With respect to overall model accuracy, Logistic regression and LDA provided the best results, followed by QDA and KNN-1.

h(i) - Freestyle

Logistic regression with Lag2 and Lag2~Lag1 synergy effect.

```r
fit.glm1 = glm(Direction ~ Lag2+Lag2:Lag1, data = Weekly, family = binomial, subset = train)
glm.prob1 = predict(fit.glm1, Weekly.9.10, type = "response")
glm.pred1 = rep("Down", length(glm.prob1))
glm.pred1[glm.prob1>.5] = "Up"
table(glm.pred1, Direction.9.10)
```

```
##          Direction.9.10
## glm.pred1 Down Up
##      Down    3  3
##      Up     40 58
```

```
mean(glm.pred1==Direction.9.10)
```

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

LDA with Lag2 and Lag2~Lag1 synergy effect.

```
lda.fit1 = lda(Direction ~ Lag2+Lag2:Lag1, data = Weekly, subset = train)
lda.pred1 = predict(lda.fit1, Weekly.9.10)
table(lda.pred1$class, Direction.9.10)
```

```
##       Direction.9.10
##        Down Up
##   Down    3  3
##   Up     40 58
```

```
mean(lda.pred1$class == Direction.9.10)
```

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

Model accuracy is equivalent to the logistic regression model accuracy (i.e., 58.65%).

QDA with Lag2 and square of (abs(Lag2))

```
qda.fit1 = qda(Direction ~ Lag2 + (abs(Lag2))^2, data = Weekly, subset = train)
qda.pred1 = predict(qda.fit1, Weekly.9.10)
table(qda.pred1$class, Direction.9.10)
```

```
##       Direction.9.10
##        Down Up
##   Down   11 12
##   Up     32 49
```

```
mean(qda.pred1$class==Direction.9.10)
```

```
## [1] 0.5769231
```

KNN (k = 5)

```
knn.pred1 = knn(train.X, test.X, train.Direction, k = 5)
table(knn.pred1, Direction.9.10)
```

```
##          Direction.9.10
## knn.pred1 Down Up
##      Down   15 20
##      Up     28 41
```

```r
mean(knn.pred1==Direction.9.10)
```

```
## [1] 0.5384615
```

Model accuracy is 53.85%.

KNN (k = 10)

```r
knn.pred2 = knn(train.X, test.X, train.Direction, k = 10)
table(knn.pred2, Direction.9.10)
```

```
##          Direction.9.10
## knn.pred2 Down Up
##      Down   18 20
##      Up     25 41
```

```r
mean(knn.pred2==Direction.9.10)
```

```
## [1] 0.5673077
```

Logistic regression and LDA outperformed the other analysis methods, based on model accuracy.

Question 11

(a) - Attach mpg01

```
attach(Auto)
```

```
## The following object is masked from package:ggplot2:
##
##     mpg
```

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

(b) - Explore the data

```
dim(Auto)
```

```
## [1] 392  10
```

```
str(Auto)
```

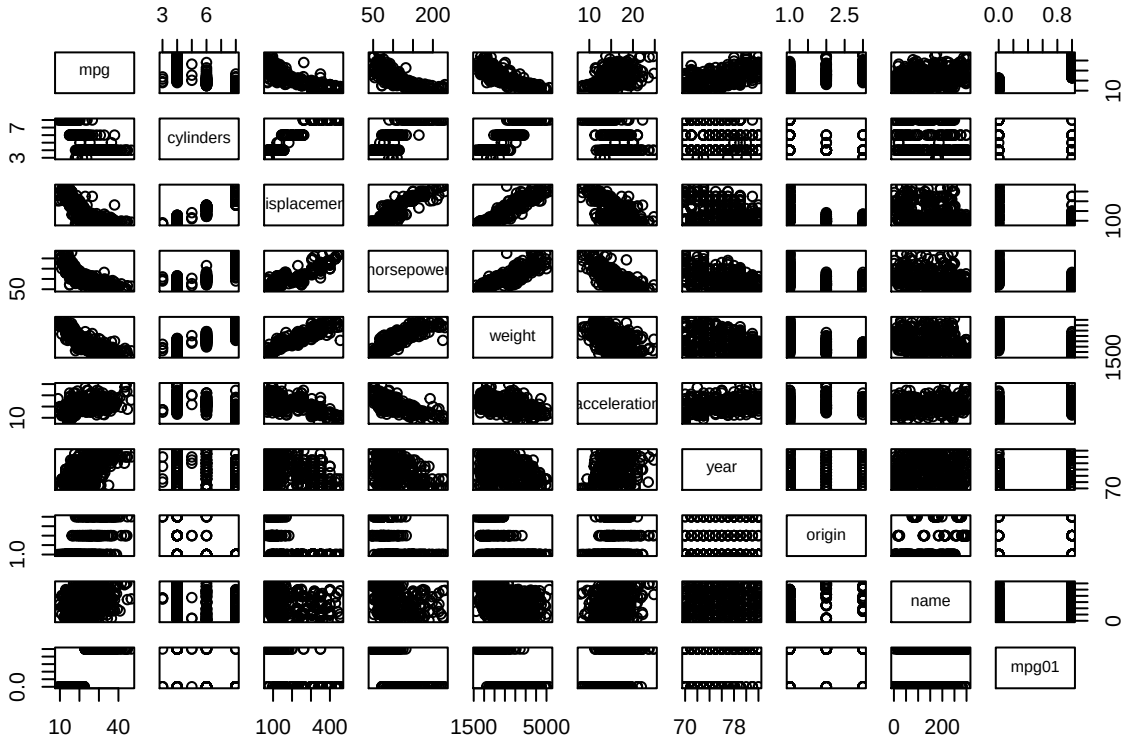
```
## 'data.frame':    392 obs. of  10 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : num  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : num  130 165 150 150 140 198 220 215 225 190 ...
##  $ weight      : num  3504 3693 3436 3433 3449 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ year        : num  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : num  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 1
##  $ mpg01       : num  0 0 0 0 0 0 0 0 0 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
```

```
pairs(Auto)
```

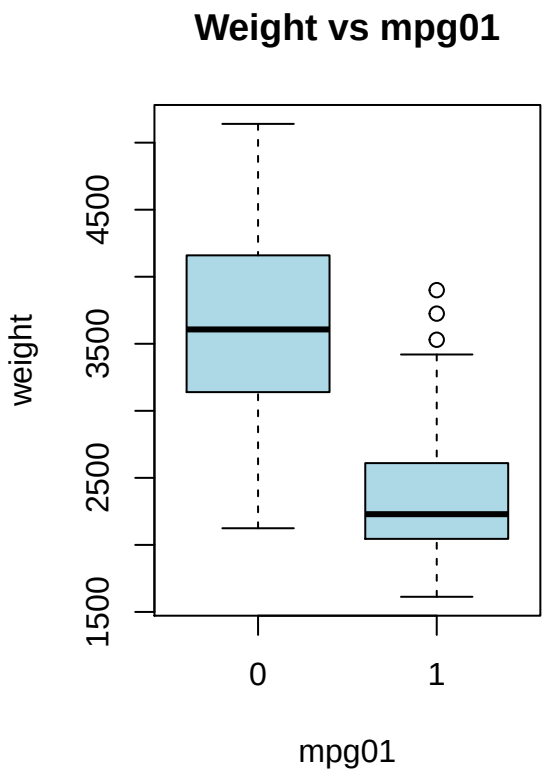


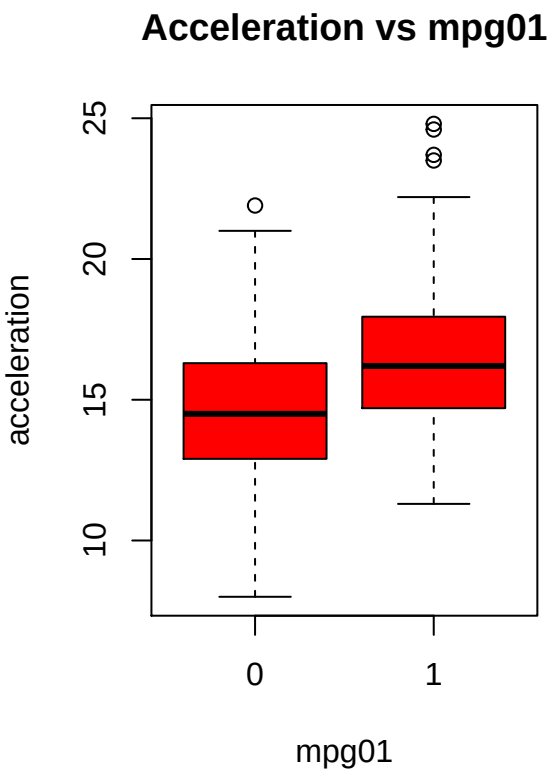

```
cor(Auto[,-9])
```

```
##                     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
```

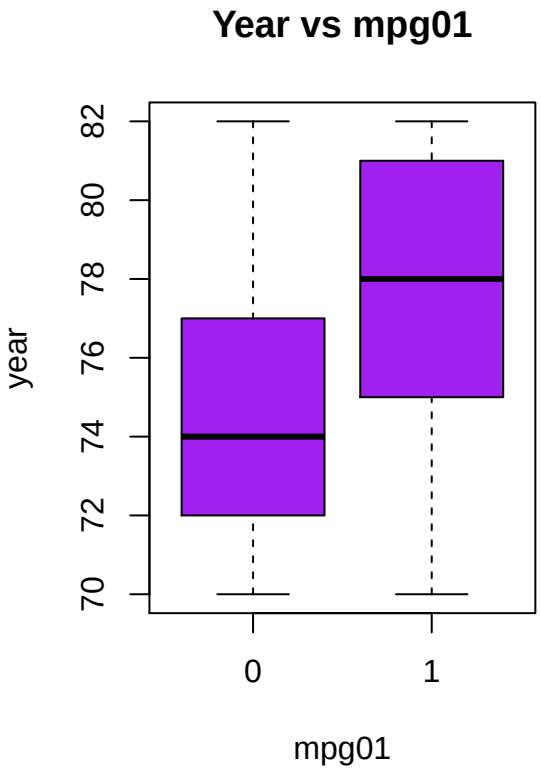
```
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
## mpg01         0.8369392 -0.7591939   -0.7534766 -0.6670526 -0.7577566
##              acceleration       year     origin      mpg01
## mpg             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
```

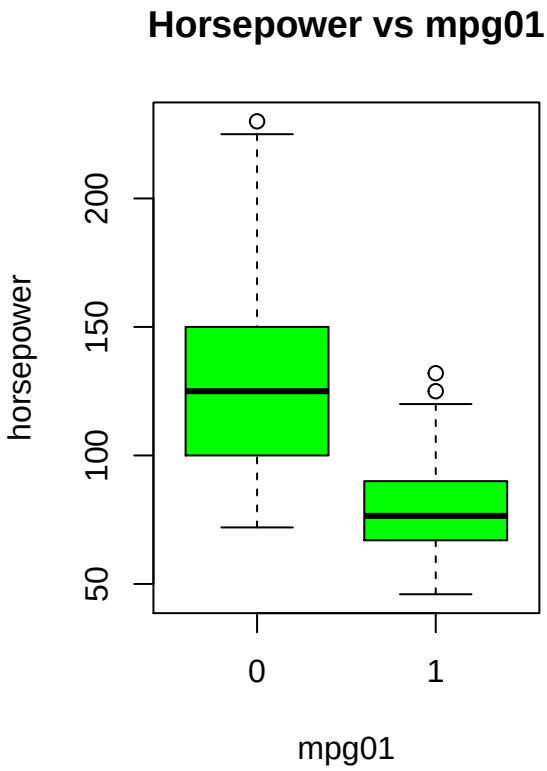
```
par(mfrow = c(1,2))
boxplot(weight~mpg01, data = Auto, main = "Weight vs mpg01", col = "light blue")
boxplot(acceleration~mpg01, data = Auto, main = "Acceleration vs mpg01", col = "red")
```



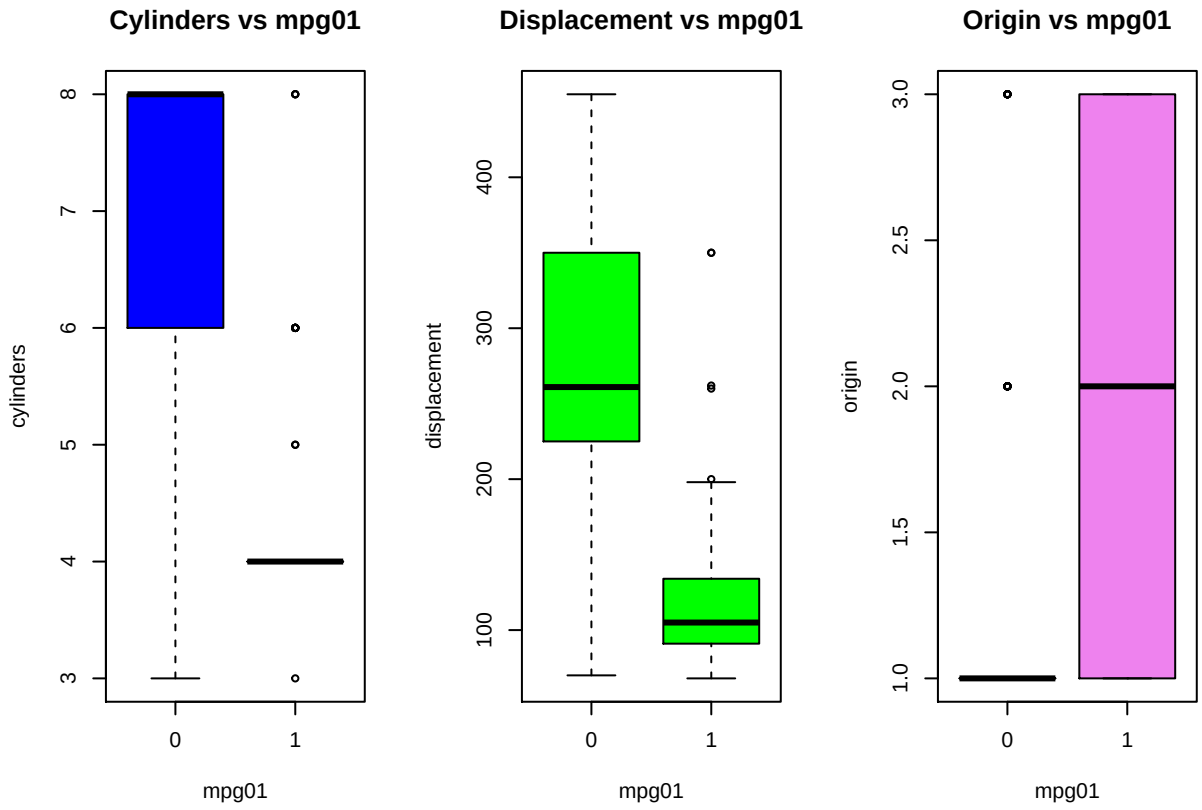



```
par(mfrow = c(1,2))
boxplot(year~mpg01, data = Auto, main = "Year vs mpg01", col = "purple")
boxplot(horsepower~mpg01, data = Auto, main = "Horsepower vs mpg01", col = "green")
```






```r
par(mfrow = c(1,3))
boxplot(cylinders~mpg01, data = Auto, main = "Cylinders vs mpg01", col = "Blue")
boxplot(displacement~mpg01, data = Auto, main = "Displacement vs mpg01", col = "green")
boxplot(origin~mpg01, data = Auto, main = "Origin vs mpg01", col = "Violet")
```



From the boxplots, we can conclude that some relationships exist between "mpg01" and "weight," "horsepower," "year," "acceleration," and "displacement."

(c) - split the data into training and testing sets

```
train = (year %% 2 == 0)
Auto.train = Auto[train, ]
Auto.test = Auto[!train, ]
mpg01.test <- mpg01[!train]
```

(d) - LDA

```
lda.fit2 = lda(mpg01 ~ weight+acceleration+horsepower+displacement+year,
               data = Auto, subset = train)
lda.fit2
```

```
## Call:
## lda(mpg01 ~ weight + acceleration + horsepower + displacement +
##     year, data = Auto, subset = train)
##
## Prior probabilities of groups:
##         0         1
```

```
## 0.4571429 0.5428571
##
## Group means:
##     weight acceleration horsepower displacement     year
## 0 3604.823     14.47500  133.14583     271.7396 74.10417
## 1 2314.763     16.62895   77.92105     111.6623 77.78947
##
## Coefficients of linear discriminants:
##                       LD1
## weight       -0.001583211
## acceleration  0.002468036
## horsepower    0.014846052
## displacement -0.008311844
## year          0.105990658
```

```
lda.pred2 = predict(lda.fit2,Auto.test)
table(lda.pred2$class,mpg01.test)
```

```
##    mpg01.test
##      0  1
##   0 84  6
##   1 16 76
```

```
mean(lda.pred2$class!=mpg01.test)
```

```
## [1] 0.1208791
```

Test error is approx.12.1%.

(e) - QDA

```
qda.fit2 = qda(mpg01 ~ weight+acceleration+horsepower+displacement+year,
               data = Auto, subset = train)
qda.fit2
```

```
## Call:
## qda(mpg01 ~ weight + acceleration + horsepower + displacement +
##     year, data = Auto, subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##     weight acceleration horsepower displacement     year
## 0 3604.823     14.47500  133.14583     271.7396 74.10417
## 1 2314.763     16.62895   77.92105     111.6623 77.78947
```

```
qda.pred2 = predict(qda.fit2,Auto.test)
table(qda.pred2$class,mpg01.test)
```

```
##    mpg01.test
##      0  1
##   0 89  8
##   1 11 74
```

```
mean(qda.pred2$class!=mpg01.test)
```

```
## [1] 0.1043956
```

Test error is approx.10.4%.

(f) - Logistic Regression

```
glm.fit2 = glm(mpg01 ~ weight+acceleration+horsepower+displacement+year,
               data = Auto, family = binomial, subset = train)
summary(glm.fit2)
```

```
##
## Call:
## glm(formula = mpg01 ~ weight + acceleration + horsepower + displacement +
##     year, family = binomial, data = Auto, subset = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4415  -0.0059   0.0142   0.1366   1.7122
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -11.044917   8.453730  -1.307   0.1914
## weight        -0.004446   0.001850  -2.404   0.0162 *
## acceleration  -0.180881   0.251599  -0.719   0.4722
## horsepower    -0.069563   0.037632  -1.849   0.0645 .
## displacement  -0.021693   0.011561  -1.876   0.0606 .
## year           0.478827   0.118756   4.032 5.53e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 289.577  on 209  degrees of freedom
## Residual deviance:  57.762  on 204  degrees of freedom
## AIC: 69.762
##
## Number of Fisher Scoring iterations: 8
```

```
glm.prob2 = predict(glm.fit2, Auto.test, type = "response")
glm.pred2 = rep(0, length(glm.prob2))
glm.pred2[glm.prob2 > 0.5] = 1
table(glm.pred2, mpg01.test)
```

```
##          mpg01.test
## glm.pred2  0  1
##         0 88  8
##         1 12 74
```

```
mean(glm.pred2 != mpg01.test)
```

```
## [1] 0.1098901
```

Test error is approx. 11.0%.

(g) - KNN (K = 1)

```
train.X1 = cbind(weight,acceleration,horsepower,displacement,year)[train,]
test.X1 = cbind(weight,acceleration,horsepower,displacement,year)[!train,]
train.mpg01 = mpg01[train]
set.seed(7)
knn.pred3 = knn(train.X1, test.X1, train.mpg01, k = 1)
table(knn.pred3, mpg01.test)
```

```
##          mpg01.test
## knn.pred3  0  1
##         0 83 11
##         1 17 71
```

```
mean(knn.pred3 != mpg01.test)
```

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

Test error is approx. 15.4%.

K = 10

```
train.X1 = cbind(weight,acceleration,horsepower,displacement,year)[train,]
test.X1 = cbind(weight,acceleration,horsepower,displacement,year)[!train,]
train.mpg01 = mpg01[train]
set.seed(7)
knn.pred4 = knn(train.X1, test.X1, train.mpg01, k = 10)
table(knn.pred4, mpg01.test)
```

```
##          mpg01.test
## knn.pred4  0  1
##         0 76  7
##         1 24 75
```

```r
mean(knn.pred4 != mpg01.test)
```

```
## [1] 0.1703297
```

Test error is approx. 15.9%.

K = 100

```r
train.X1 = cbind(weight,acceleration,horsepower,displacement,year)[train,]
test.X1 = cbind(weight,acceleration,horsepower,displacement,year)[!train,]
train.mpg01 = mpg01[train]
set.seed(7)
knn.pred5 = knn(train.X1, test.X1, train.mpg01, k = 100)
table(knn.pred5, mpg01.test)
```

```
##          mpg01.test
## knn.pred5  0  1
##         0 81  7
##         1 19 75
```

```r
mean(knn.pred5 != mpg01.test)
```

```
## [1] 0.1428571
```

Test error is approx. 14.3%. A K value of 100 seems to yield the lowest test error.