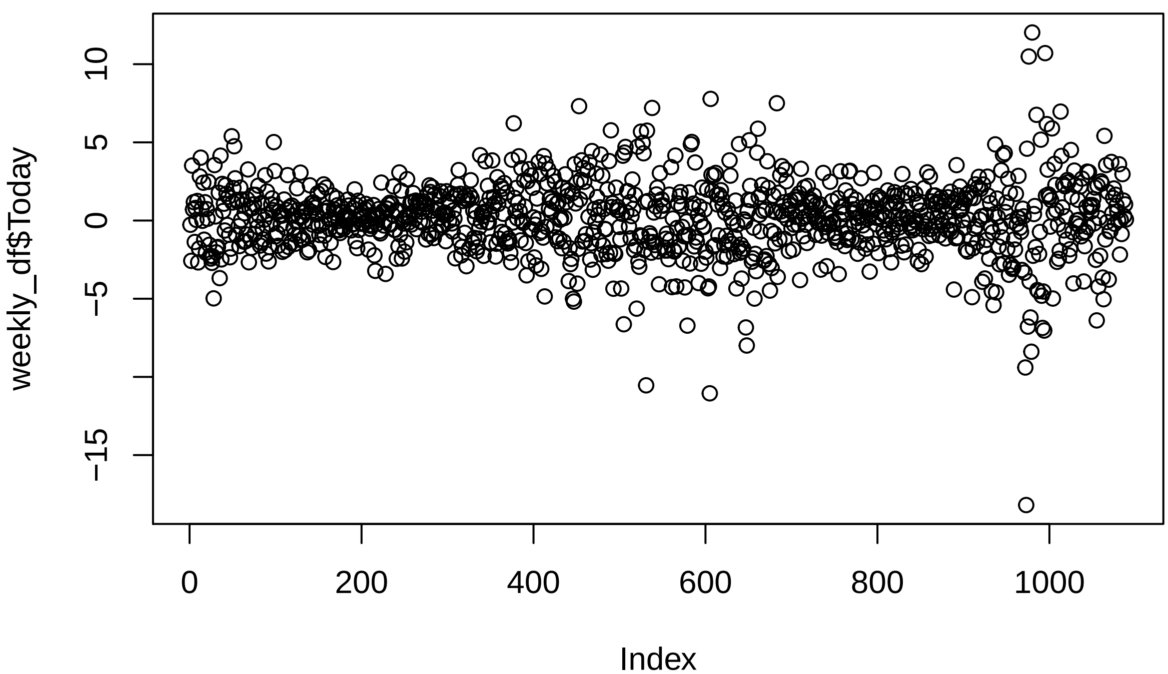
HomeWork_4

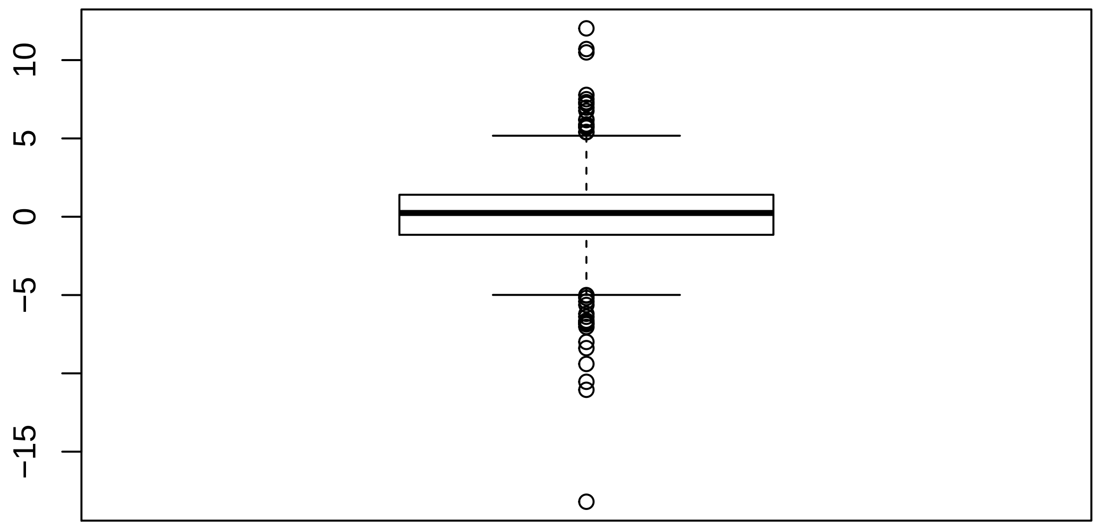
Viveksinh Solanki

4/7/2020

```r
### Problem 1 ###

## a)
weekly_df <- read.csv('Weekly.csv')
#View(weekly_df)

# plots
plot(weekly_df$Today)
```




```r
boxplot(weekly_df$Today)
```




```r
# From the plots, we can see that value for 'today' variable is lying between
# -5 and 5. From the scatter plot we can see that straight line can be fitted
# through whole dataset for 'today' as target variable

# summary stats
summary(weekly_df)
```

```
##       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
# From the summary stats, it is visible that except volume all other features
# have mean value ~0.15
# number of data points with 'Down' direction is 484
# number of data points with 'Up' direction is 605


## b) Logistic Regression
library(ISLR)

glm.fit=glm(weekly_df$Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,
            data=weekly_df,family=binomial)

summary(glm.fit)
```

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

```r
confint(glm.fit)
```

```
## Waiting for profiling to be done...
```

```
##                    2.5 %     97.5 %
## (Intercept)  0.098808746 0.43580101
## Lag1        -0.093477110 0.01029269
## Lag2         0.006197597 0.11169774
## Lag3        -0.068653910 0.03604309
## Lag4        -0.079952378 0.02401603
```

```
## Lag5        -0.066495108 0.03711989
## Volume      -0.095051949 0.04979338
```

```r
confint.default(glm.fit)
```

```
##                    2.5 %     97.5 %
## (Intercept)  0.098445204 0.43528308
## Lag1        -0.093032105 0.01049422
## Lag2         0.005787254 0.11109610
## Lag3        -0.068319640 0.03619735
## Lag4        -0.079657357 0.02407694
## Lag5        -0.066185275 0.03724115
## Volume      -0.095060526 0.04957746
```

```r
exp(coef(glm.fit))
```

```
## (Intercept)        Lag1        Lag2        Lag3        Lag4        Lag5
##   1.3058630   0.9595710   1.0601831   0.9840671   0.9725924   0.9856322
##      Volume
##   0.9775151
```

```r
# From above results, predictor Lag2 seems statistically significant.

## c)
glm.probs=predict(glm.fit, type="response")
glm.probs[1:5]
```

```
##         1         2         3         4         5
## 0.6086249 0.6010314 0.5875699 0.4816416 0.6169013
```

```r
dim(weekly_df)
```

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

```r
glm.pred=rep("Down",dim(weekly_df)[1])
glm.pred[glm.probs>0.5]="Up"
contrasts(weekly_df$Direction)
```

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

```r
# Confusion matrix
table(glm.pred,weekly_df$Direction)
```

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

```r
# Overall fraction of correct predictions
mean(glm.pred == weekly_df$Direction)
```

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

```r
#overall error rate
1-mean(glm.pred == weekly_df$Direction)
```

```
## [1] 0.4389348
```

```r
# Understanding confusion matrix and mistakes made by logistic regression
#error among direction 'Up'
48/(48+557)
```

```
## [1] 0.07933884
```

```r
#sensitivity (percentage of true 'Up' direction identified)
557/(48+557) ## (tp/tp+fn)
```

```
## [1] 0.9206612
```

```r
#specificity (percentage of true 'Down' direction that are correctly identified)
54/(54+430) ## (tn/tn+fp)
```

```
## [1] 0.1115702
```

```r
## d)

# split data into training and held out/testing sets
training=weekly_df[1:985,]
test=weekly_df[986:1089,]

glm.fit=glm(Direction~Lag2,data=training,family=binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag2, family = binomial, data = training)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.536  -1.264   1.021   1.091   1.368
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.20326    0.06428   3.162  0.00157 **
## Lag2         0.05810    0.02870   2.024  0.04298 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1350.5  on 983  degrees of freedom
## AIC: 1354.5
##
## Number of Fisher Scoring iterations: 4
```

```r
glm.probs=predict(glm.fit,newdata = test, type="response")
glm.pred=rep("Down",dim(test)[1])
glm.pred[glm.probs>0.5]="Up"
contrasts(test$Direction)
```

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

```r
# Confusion matrix
table(glm.pred,test$Direction)
```

```
##
## glm.pred Down Up
##     Down    9  5
##     Up     34 56
```

```r
# Overall fraction of correct predictions
logistic_acc <- mean(glm.pred == test$Direction)
logistic_acc
```

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

```r
#overall error rate
1-mean(glm.pred == test$Direction)
```

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

```r
## e) LDA
library(MASS)

lda.fit=lda(Direction~Lag2,data=training)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag2, data = training)
##
## 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
lda.pred=predict(lda.fit,test)
lda.class=lda.pred$class
contrasts(test$Direction)
```

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

```r
# Confusion matrix
table(lda.class ,test$Direction)
```

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

```r
# Overall fraction of correct predictions
lda_acc <- mean(lda.class == test$Direction)
lda_acc
```

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

```r
## f) QDA

qda.fit=qda(Direction~Lag2,data=training)
qda.fit
```

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

```r
qda.pred=predict(qda.fit,test)
qda.class=qda.pred$class
contrasts(test$Direction)
```

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

```r
# Confusion matrix
table(qda.class ,test$Direction)
```

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

```r
# Overall fraction of correct predictions
qda_acc <- mean(qda.class == test$Direction)
qda_acc
```

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

```r
## g) KNN with k=1
test.x=cbind(test$Lag2)
training.x=cbind(training$Lag2)
library(class)
knn.pred=knn(training.x,test.x,training$Direction,k=1)

contrasts(test$Direction)
```

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

```
# Confusion matrix
table(knn.pred ,test$Direction)
```

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

```
# Overall fraction of correct predictions
knn_acc <- mean(knn.pred == test$Direction)
knn_acc
```

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

```
## h) Compare above results - Logistic regression and LDA provide best results
# Predictions
comparison=cbind(test,glm.pred,lda.class,qda.class,knn.pred)
head(comparison)
```

```
##     Year   Lag1   Lag2   Lag3   Lag4   Lag5   Volume  Today Direction
## 986 2009  6.760 -1.698  0.926  0.418 -2.251 3.793110 -4.448      Down
## 987 2009 -4.448  6.760 -1.698  0.926  0.418 5.043904 -4.518      Down
## 988 2009 -4.518 -4.448  6.760 -1.698  0.926 5.948758 -2.137      Down
## 989 2009 -2.137 -4.518 -4.448  6.760 -1.698 6.129763 -0.730      Down
## 990 2009 -0.730 -2.137 -4.518 -4.448  6.760 5.602004  5.173        Up
## 991 2009  5.173 -0.730 -2.137 -4.518 -4.448 6.217632 -4.808      Down
##     glm.pred lda.class qda.class knn.pred
## 986       Up        Up        Up       Up
## 987       Up        Up        Up       Up
## 988     Down      Down        Up     Down
## 989     Down      Down        Up     Down
## 990       Up        Up        Up     Down
## 991       Up        Up        Up       Up
```

```
# Overall fraction of correct predictions
cbind(logistic_acc, lda_acc, qda_acc, knn_acc)
```

```
##      logistic_acc lda_acc   qda_acc knn_acc
## [1,]        0.625   0.625 0.5865385     0.5
```

```
## i) Experiments with different predictors

#-- A) predictors: Lag2 + Volume
col_names <- c('Lag2', 'Volume', 'Direction')
a_subset <- weekly_df[col_names]
training=a_subset[1:985,]
test=a_subset[986:1089,]

#-- I) Logistic Regression
glm.fit=glm(Direction~Lag2+Volume, data=training,family=binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag2 + Volume, family = binomial, data = training)
##
## Deviance Residuals:
```

```
##    Min      1Q  Median      3Q     Max
## -1.413  -1.262   1.020   1.087   1.485
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.27069    0.09006   3.006  0.00265 **
## Lag2         0.05350    0.02905   1.842  0.06554 .
## Volume      -0.05548    0.05180  -1.071  0.28417
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1349.4  on 982  degrees of freedom
## AIC: 1355.4
##
## Number of Fisher Scoring iterations: 4
```

```r
glm.probs=predict(glm.fit,newdata = test, type="response")
glm.pred=rep('Down',dim(test)[1])
glm.pred[glm.probs>0.5]='Up'

# Confusion matrix
table(glm.pred,test$Direction)
```

```
##
## glm.pred Down Up
##     Down   20 25
##     Up     23 36
```

```r
# Overall fraction of correct predictions
logistic_acc <- mean(glm.pred == test$Direction)
logistic_acc
```

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

```r
#-- II) LDA
lda.fit=lda(Direction~Lag2+Volume,
            data=training)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag2 + Volume, data = training)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2   Volume
## Down -0.03568254 1.266966
## Up    0.26036581 1.156529
##
## Coefficients of linear discriminants:
##               LD1
```

```
## Lag2    0.3590979
## Volume -0.3756392
```

```
lda.pred=predict(lda.fit,test)
lda.class=lda.pred$class
# Confusion matrix
table(lda.class ,test$Direction)
```

```
##
## lda.class Down Up
##      Down   20 25
##      Up     23 36
```

```
# Overall fraction of correct predictions
lda_acc <- mean(lda.class == test$Direction)
lda_acc
```

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

```
#-- III) QDA
qda.fit=qda(Direction~Lag2+Volume,
            data=training)
qda.fit
```

```
## Call:
## qda(Direction ~ Lag2 + Volume, data = training)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2   Volume
## Down -0.03568254 1.266966
## Up    0.26036581 1.156529
```

```
qda.pred=predict(qda.fit,test)
qda.class=qda.pred$class
# Confusion matrix
table(qda.class ,test$Direction)
```

```
##
## qda.class Down Up
##      Down   32 44
##      Up     11 17
```

```
# Overall fraction of correct predictions
qda_acc <- mean(qda.class == test$Direction)
qda_acc
```

```
## [1] 0.4711538
```

```
#-- IV) KNN for k=1,2,3,5,7

test.x=cbind(test$Lag2, test$Volume)
training.x=cbind(training$Lag2, training$Volume)
library(class)
```

```r
k_ls <- c(1,2,3,5,7)
k_acc <- c()
i=0

for(k in k_ls){
  knn.pred=knn(training.x,test.x,training$Direction,k=k)
  contrasts(test$Direction)
  # Confusion matrix
  table(knn.pred ,test$Direction)
  # Overall fraction of correct predictions
  knn_acc <- mean(knn.pred == test$Direction)
  k_acc <- c(k_acc,knn_acc)
  i=i+1
}

k_acc
```

```
## [1] 0.5576923 0.4711538 0.5480769 0.5288462 0.4711538
```

```r
#-- B) predictors: Lag1 + Lag3 + Lag4 + Lag5 + Volume
col_names <- c('Lag1', 'Lag3', 'Lag4', 'Lag5', 'Volume', 'Direction')
a_subset <- weekly_df[col_names]
training=a_subset[1:985,]
test=a_subset[986:1089,]

#-- I) Logistic Regression
glm.fit=glm(Direction~Lag1+Lag3+Lag4+Lag5+Volume, data=training,family=binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag3 + Lag4 + Lag5 + Volume,
##     family = binomial, data = training)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5522  -1.2490   0.9887   1.0868   1.4014
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.35621    0.09286   3.836 0.000125 ***
## Lag1        -0.06643    0.02925  -2.271 0.023120 *
## Lag3        -0.01972    0.02907  -0.678 0.497565
## Lag4        -0.02975    0.02908  -1.023 0.306292
## Lag5        -0.04051    0.02898  -1.398 0.162104
## Volume      -0.10376    0.05312  -1.953 0.050793 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1344.6  on 979  degrees of freedom
## AIC: 1356.6
```

```
##
## Number of Fisher Scoring iterations: 4
```

```
glm.probs=predict(glm.fit,newdata = test, type="response")
glm.pred=rep('Down',dim(test)[1])
glm.pred[glm.probs>0.5]='Up'

# Confusion matrix
table(glm.pred,test$Direction)
```

```
##
## glm.pred Down Up
##     Down   36 47
##     Up      7 14
```

```
# Overall fraction of correct predictions
logistic_acc <- mean(glm.pred == test$Direction)
logistic_acc
```

```
## [1] 0.4807692
```

```
#-- II) LDA
lda.fit=lda(Direction~Lag1+Lag3+Lag4+Lag5+Volume,
            data=training)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag1 + Lag3 + Lag4 + Lag5 + Volume, data = training)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##              Lag1       Lag3       Lag4       Lag5   Volume
## Down  0.289444444 0.17080045 0.15925624 0.21409297 1.266966
## Up   -0.009213235 0.08404044 0.09220956 0.04548897 1.156529
##
## Coefficients of linear discriminants:
##                LD1
## Lag1   -0.32218170
## Lag3   -0.09604607
## Lag4   -0.14492618
## Lag5   -0.19661300
## Volume -0.50985059
```

```
lda.pred=predict(lda.fit,test)
lda.class=lda.pred$class
# Confusion matrix
table(lda.class ,test$Direction)
```

```
##
## lda.class Down Up
##      Down   36 47
##      Up      7 14
```

```r
# Overall fraction of correct predictions
lda_acc <- mean(lda.class == test$Direction)
lda_acc
```

```
## [1] 0.4807692
```

```r
#-- III) QDA
qda.fit=qda(Direction~Lag1+Lag3+Lag4+Lag5+Volume,
            data=training)
qda.fit
```

```
## Call:
## qda(Direction ~ Lag1 + Lag3 + Lag4 + Lag5 + Volume, data = training)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##               Lag1       Lag3       Lag4       Lag5   Volume
## Down  0.289444444 0.17080045 0.15925624 0.21409297 1.266966
## Up   -0.009213235 0.08404044 0.09220956 0.04548897 1.156529
```

```r
qda.pred=predict(qda.fit,test)
qda.class=qda.pred$class
# Confusion matrix
table(qda.class ,test$Direction)
```

```
##
## qda.class Down Up
##      Down   36 53
##      Up      7  8
```

```r
# Overall fraction of correct predictions
qda_acc <- mean(qda.class == test$Direction)
qda_acc
```

```
## [1] 0.4230769
```

```r
#-- IV) KNN for k=1,2,3,5,7

test.x=cbind(test$Lag1, test$Lag3, test$Lag4, test$Lag5, test$Volume)
training.x=cbind(training$Lag1, training$Lag3, training$Lag4, training$Lag5,
                 training$Volume)
library(class)

k_ls <- c(1,2,3,5,7)
k_acc <- c()
i=0

for(k in k_ls){
  knn.pred=knn(training.x,test.x,training$Direction,k=k)
  contrasts(test$Direction)
  # Confusion matrix
  table(knn.pred ,test$Direction)
  # Overall fraction of correct predictions
```

```r
  knn_acc <- mean(knn.pred == test$Direction)
  k_acc <- c(k_acc,knn_acc)
  i=i+1
}

k_acc
```

```
## [1] 0.4807692 0.5096154 0.5192308 0.4326923 0.4711538
```

```r
#Answer:
#Best Variables: Lag1 + Lag3 + Lag4 + Lag5 + Volume
#Best Method: KNN with k=2
#Best confusion matrix:
#knn.pred Down Up
#Down    23 26
#Up      20 35

### Problem 2 ###

auto_df <- read.csv('Auto.csv')
#View(auto_df)

## a)
mpg01 <- rep(0, dim(auto_df)[1])

# Median of mpg
med_mpg <- median(auto_df$mpg)
med_mpg
```

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

```r
# mpg01 vector with 0's and 1's
mpg01[auto_df$mpg > med_mpg] = 1
mpg01
```

```
##   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 1 1 1 1 0 0 0 0 0 1 1 1 0 0 0
##  [36] 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0
##  [71] 0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0
## [106] 0 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 1
## [141] 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 0 1 0
## [176] 1 1 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0
## [211] 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 1 1 1
## [246] 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 1 0 0 0 0 1 1 0 0
## [281] 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
## [316] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [351] 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
## [386] 1 1 1 1 1 1 1
```

```r
auto_df <- cbind(auto_df, mpg01)
head(auto_df)
```

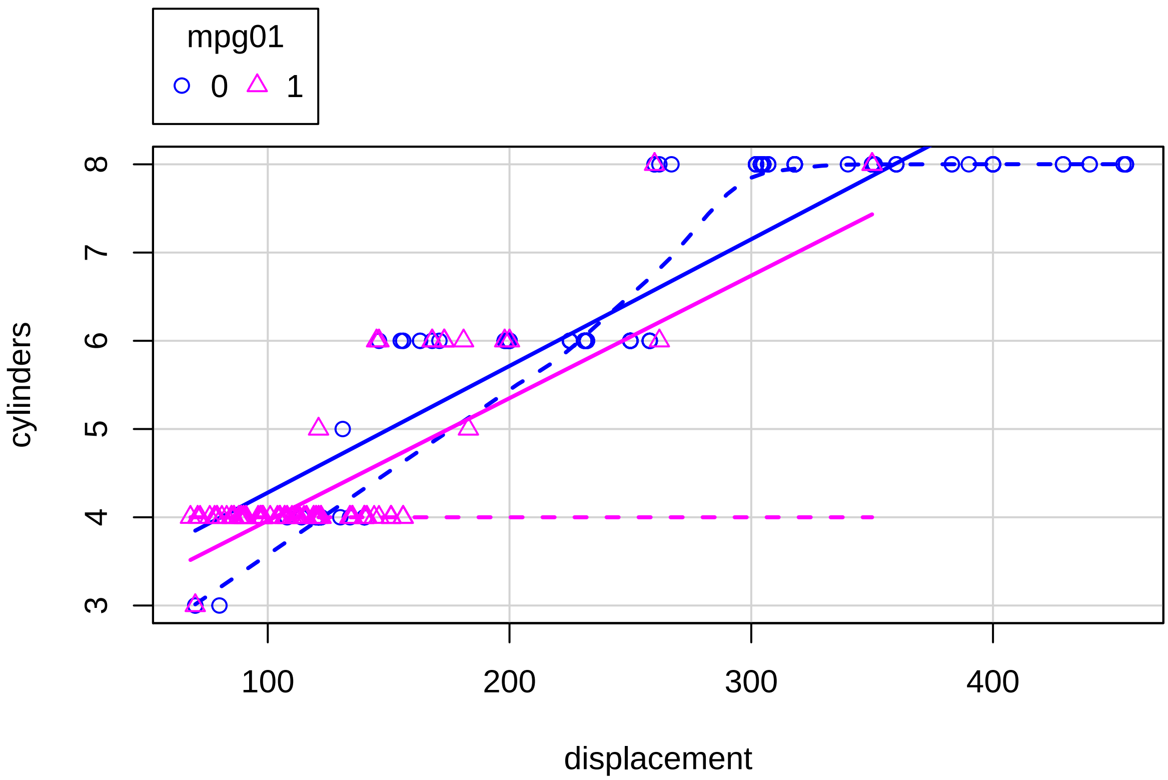
```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
```

```
## 5  17          8          302        140   3449          10.5    70       1
## 6  15          8          429        198   4341          10.0    70       1
##                         name mpg01
## 1 chevrolet chevelle malibu     0
## 2         buick skylark 320     0
## 3        plymouth satellite     0
## 4             amc rebel sst     0
## 5               ford torino     0
## 6          ford galaxie 500     0
```

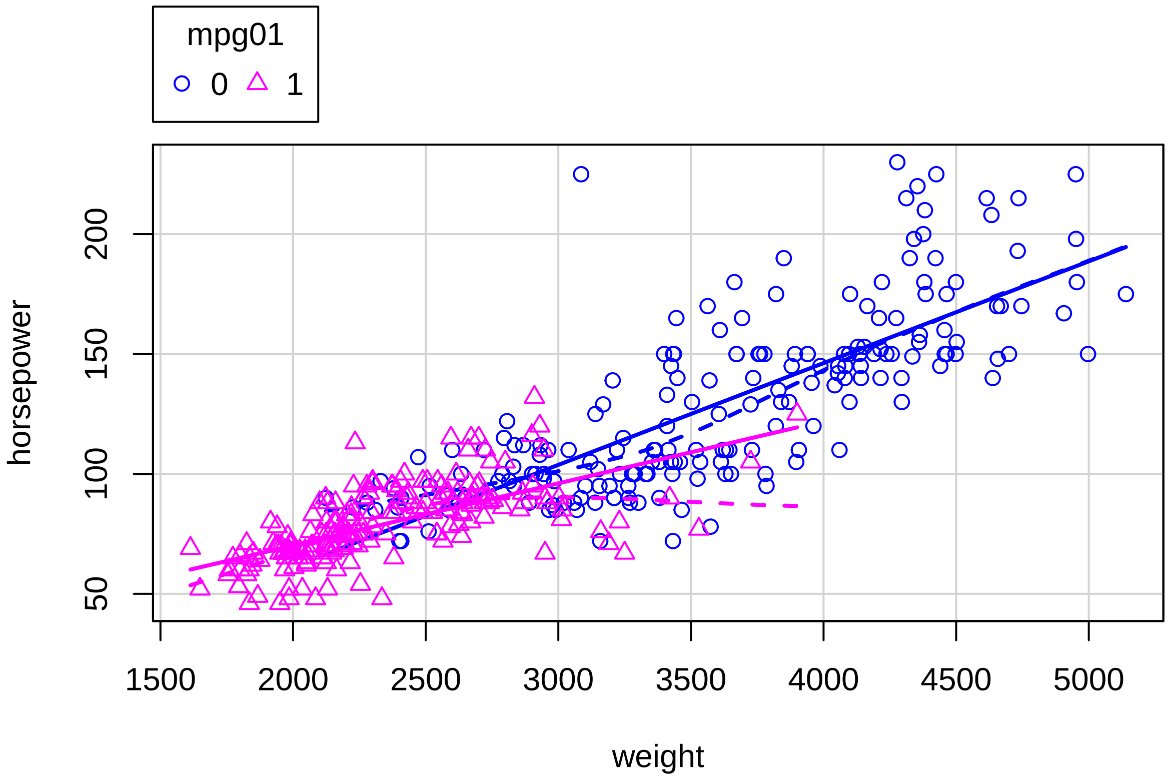
```r
## b) Explore relationship of mpg01 with other vars

library(car)
```

```
## Loading required package: carData
```

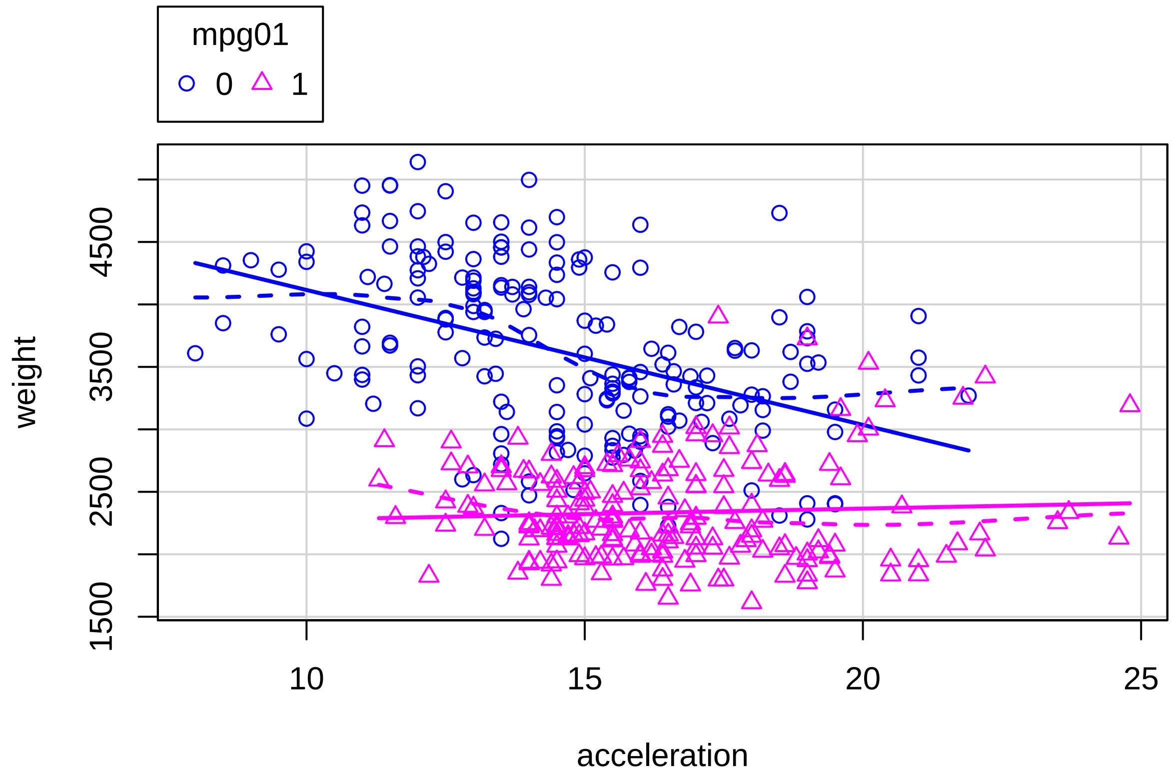
```r
## cylinders vs displacement w.r.t mpg01:
# We can see that mileage is higher for less number of cylinders and low displacement
# And mileage is lower for higher number of cylinders and higher displacement values
scatterplot(cylinders ~ displacement | mpg01, data=auto_df,
            xlab="displacement", ylab="cylinders")
```



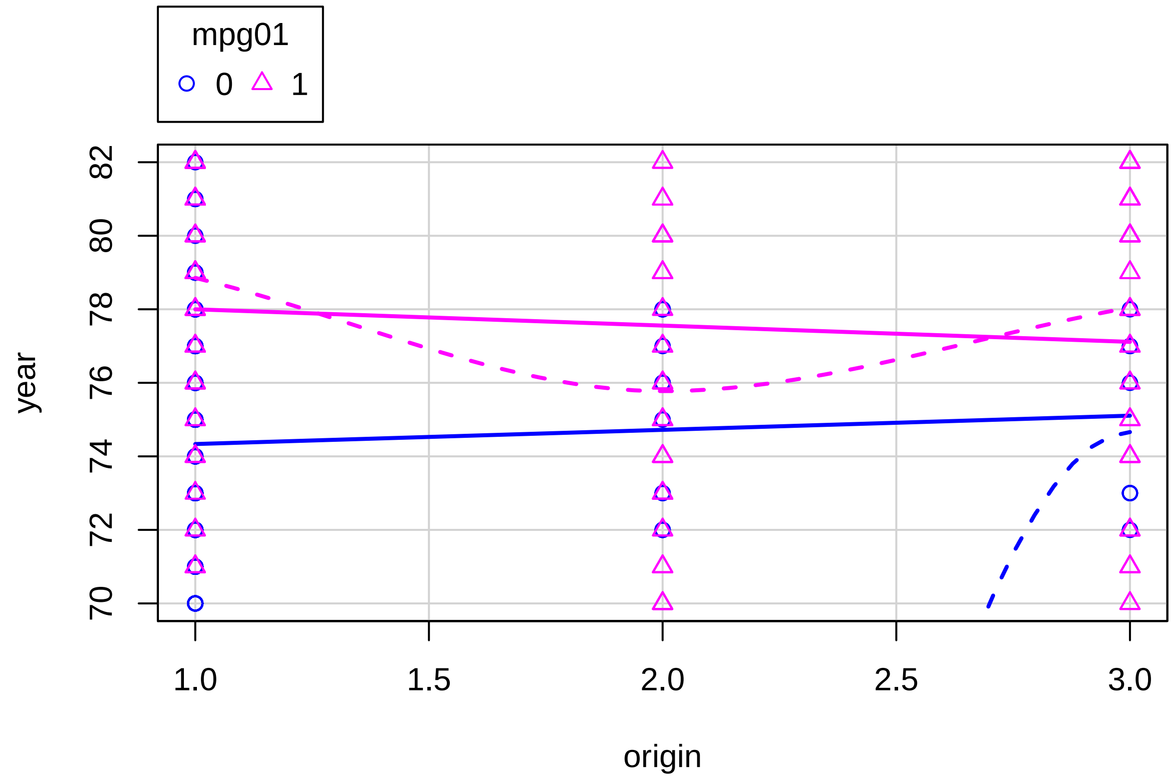

```r
## horsepower vs weight w.r.t mpg01:
# From plot: mileage is higher for lower weight values and lower horsepower values
scatterplot(horsepower ~ weight | mpg01, data=auto_df,
            xlab="weight", ylab="horsepower")
```



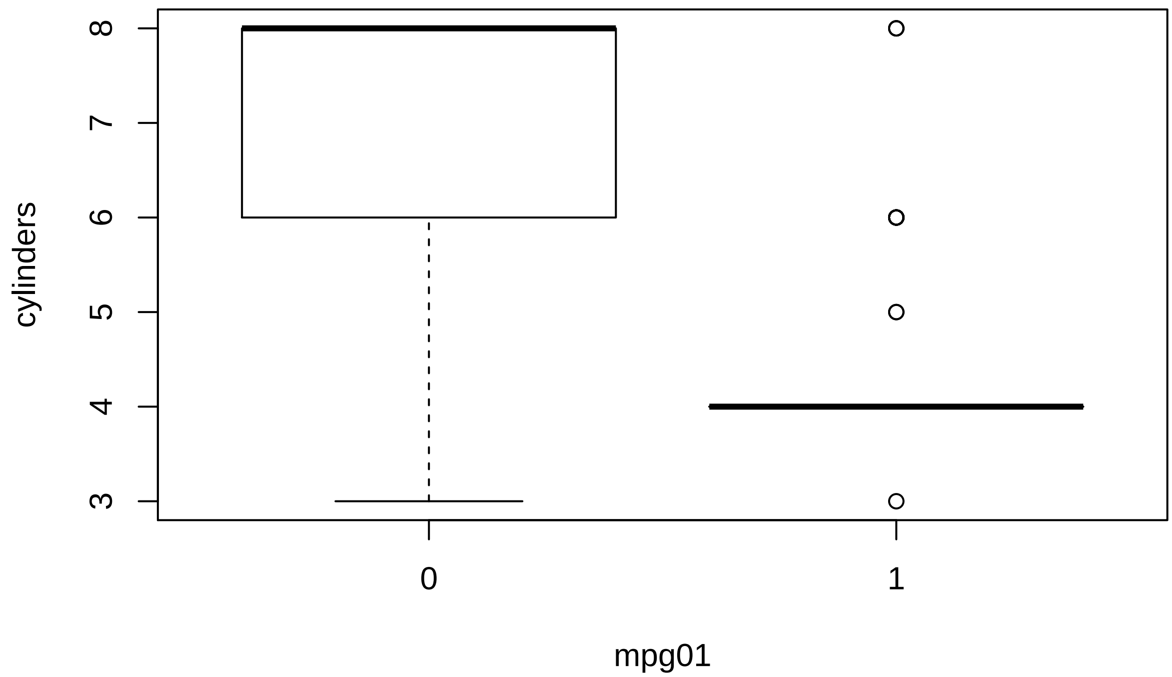

```r
## weight vs acceleration w.r.t. mpg01:
# From plot: mileage is slightly skewed
scatterplot(weight ~ acceleration | mpg01, data=auto_df,
            xlab="acceleration", ylab="weight")
```



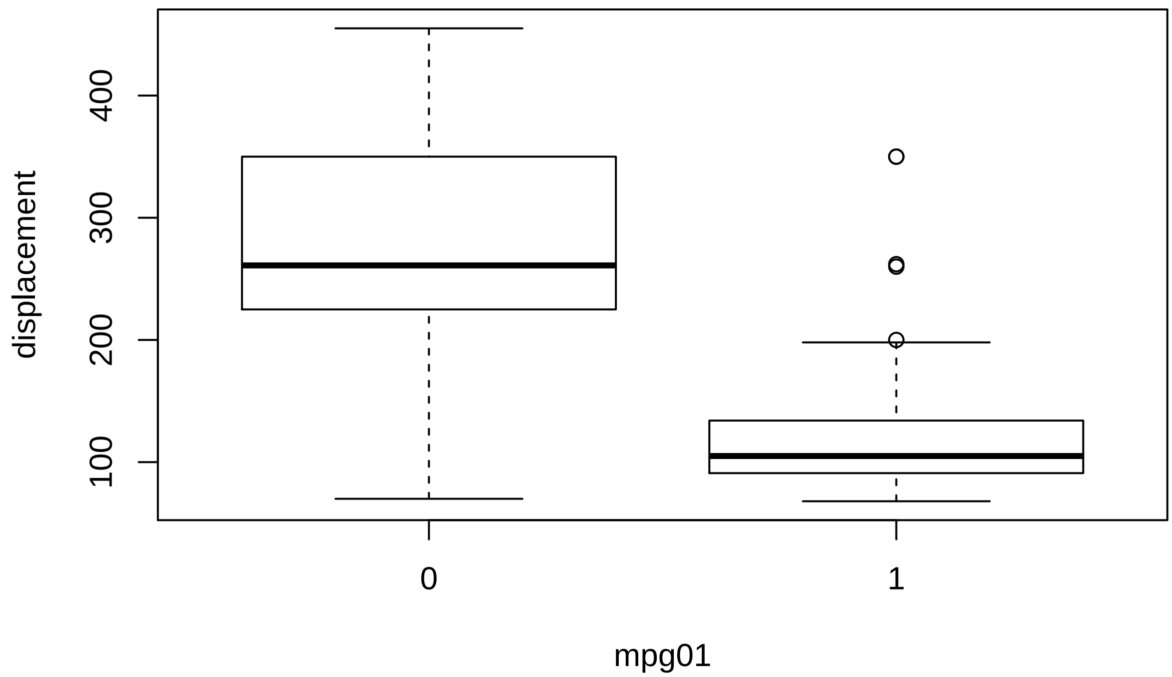

```
## year vs origin w.r.t mpg01:
# From plot: mileage doesn't seem to depend on year or origin, so we can ignore
# year and origin from predictors
scatterplot(year ~ origin | mpg01, data=auto_df, xlab="origin", ylab="year")
```



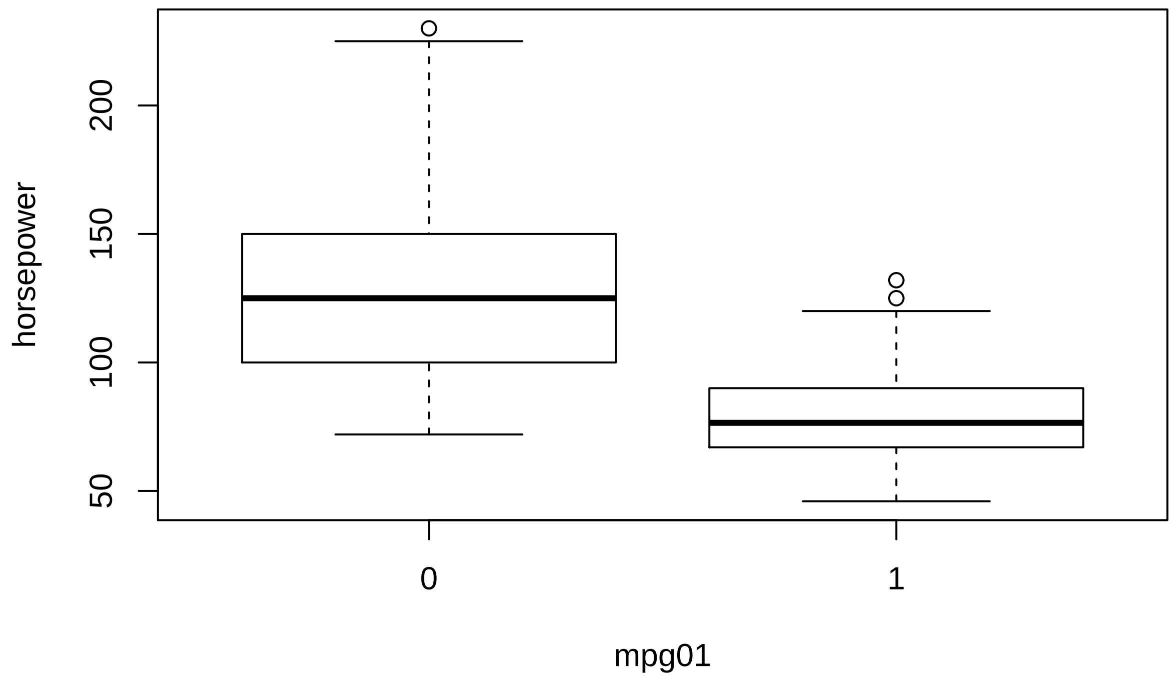

```
# From plot: Lower number cylinders give higher mileage
boxplot(cylinders ~ mpg01, data=auto_df, xlab="mpg01", ylab="cylinders")
```



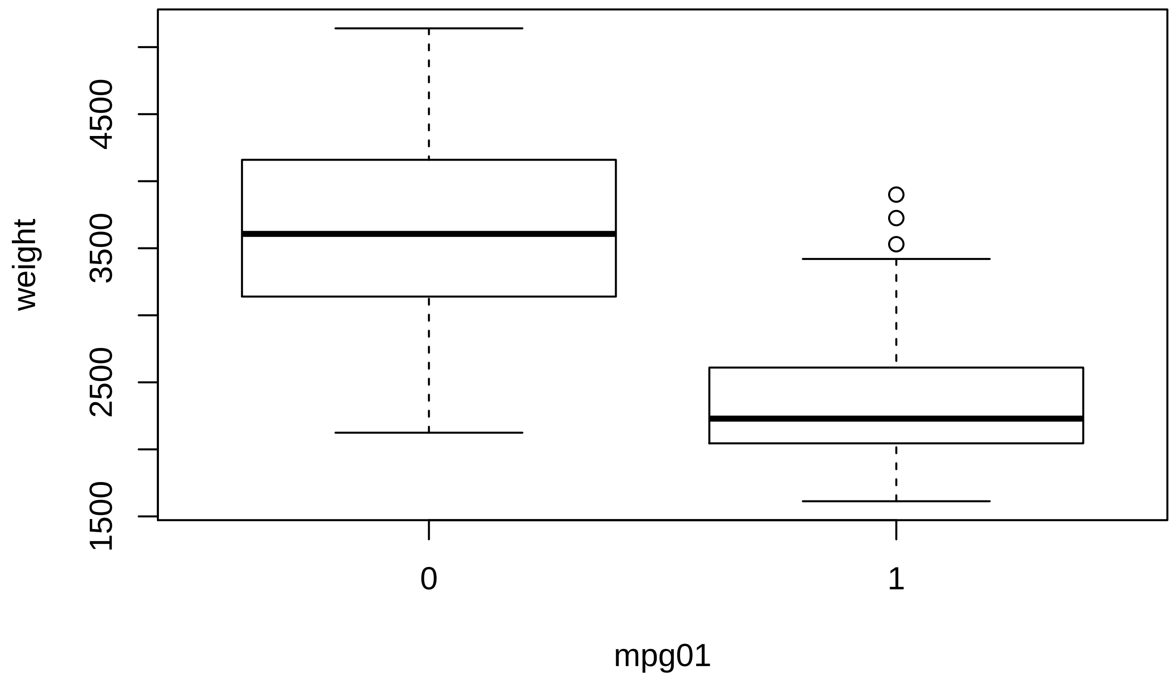

```r
# From plot: Higher displacement implies lower mileage
boxplot(displacement ~ mpg01, data=auto_df, xlab="mpg01", ylab="displacement")
```



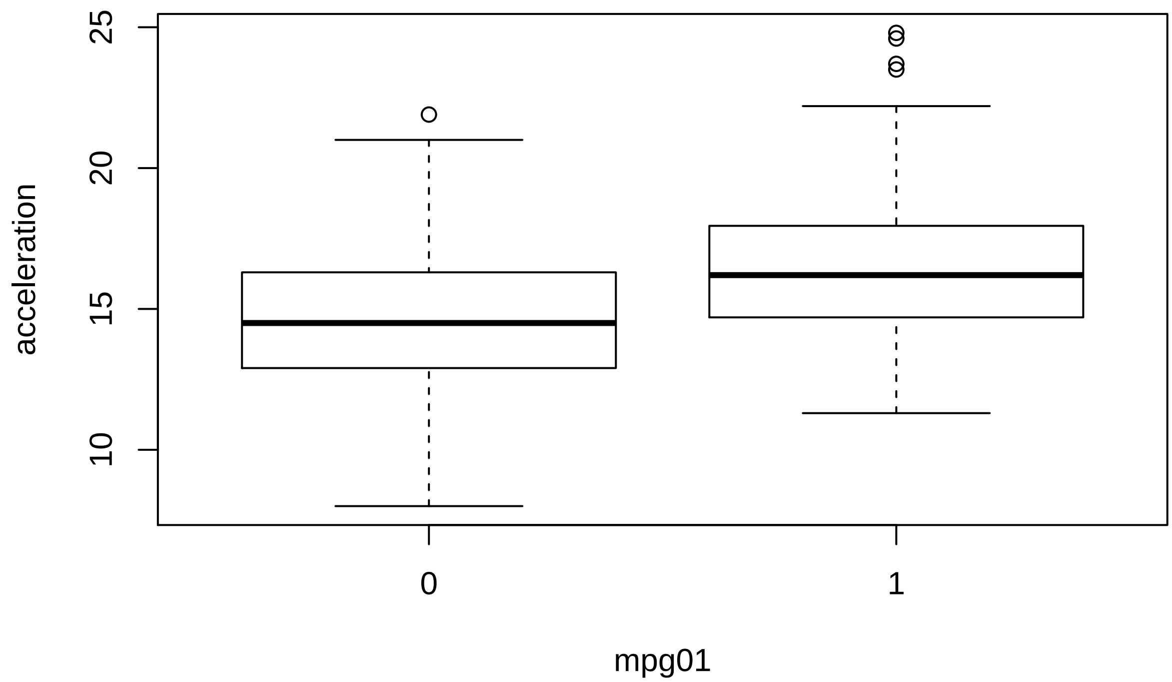

```r
# From plot: Mileage is higher for horsepower less than 100
boxplot(horsepower ~ mpg01, data=auto_df, xlab="mpg01", ylab="horsepower")
```



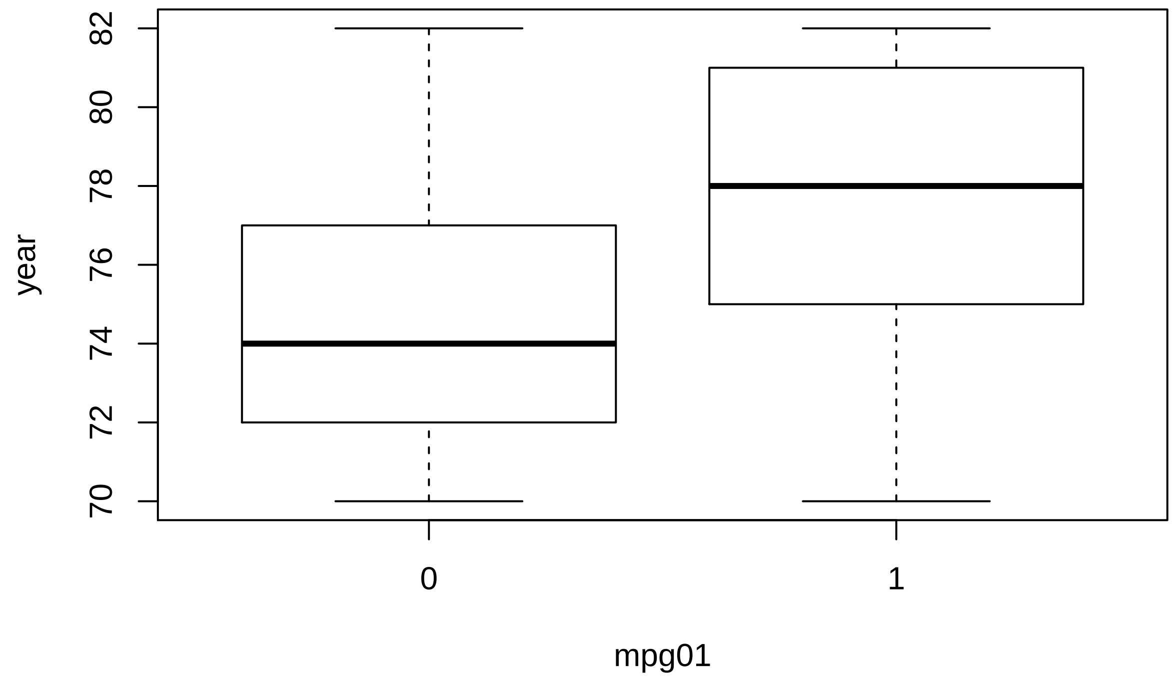

```r
# From plot: Mileage is higher for low weight vehicles
boxplot(weight ~ mpg01, data=auto_df, xlab="mpg01", ylab="weight")
```



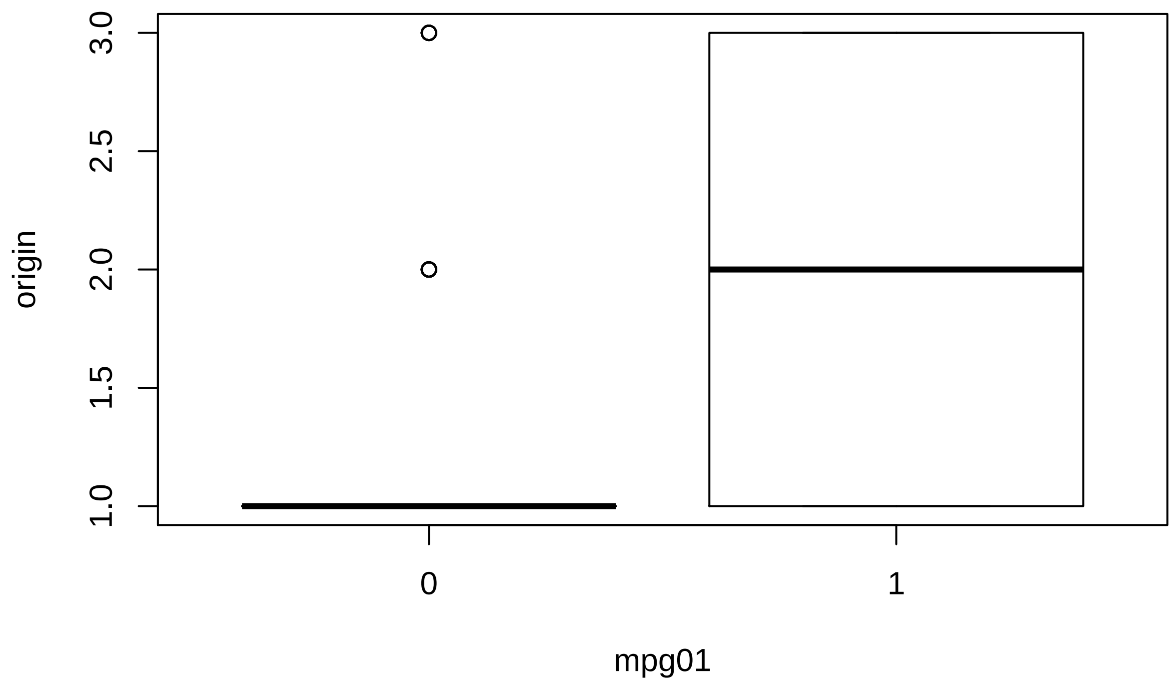

```
# From plot: Mileage doesn't seem to depend much on acceleration
boxplot(acceleration ~ mpg01, data=auto_df, xlab="mpg01", ylab="acceleration")
```




```r
# From plot, we can see that old cars had lower mileage than newer cars
boxplot(year ~ mpg01, data=auto_df, xlab="mpg01", ylab="year")
```




```r
# From plot: Origin doesn't seem to affect mileage much, we can ignore this variable
# from predictors
boxplot(origin ~ mpg01, data=auto_df, xlab="mpg01", ylab="origin")
```




```r
# From the above plots we can see that variables 'origin' and 'year'
# don't have relationship with target 'mpg01'. Hence, we can take rest of the
# variables as predictors for this dataset

## c) Train test split

# Processing and Scaling
col_names <- c('cylinders', 'displacement', 'horsepower', 'weight',
               'acceleration')
auto_df_subset <- auto_df[col_names]
#View(auto_df_subset)

# Normalized subset
auto_df_subset_scaled <- scale(auto_df_subset)
#View(auto_df_subset_scaled)

# add mpg01 col
auto_df_subset_scaled_final <- cbind(auto_df_subset_scaled, mpg01)
#View(auto_df_subset_scaled_final)

# Split
split_size = round(dim(auto_df)[1] * 0.8)
split_size
```

```
## [1] 314
```

```r
training=data.frame(auto_df_subset_scaled_final[1:split_size,])
test=data.frame(auto_df_subset_scaled_final[315:392,])

#View(training)
#View(test)
dim(training)
```

```
## [1] 314   6
```

```r
dim(test)
```

```
## [1] 78  6
```

```r
## d) LDA

lda.fit=lda(mpg01~cylinders+displacement+horsepower+weight+acceleration,
            data=training)
lda.fit
```

```
## Call:
## lda(mpg01 ~ cylinders + displacement + horsepower + weight +
##     acceleration, data = training)
##
## Prior probabilities of groups:
##         0         1
## 0.6082803 0.3917197
##
## Group means:
##    cylinders displacement horsepower     weight acceleration
## 0  0.7699698    0.7651627   0.688721  0.7687844    -0.365302
## 1 -0.7818856   -0.7861089  -0.671833 -0.8180198     0.332754
##
## Coefficients of linear discriminants:
##                      LD1
## cylinders    -0.71689917
## displacement -0.08463165
## horsepower    0.19728376
## weight       -0.93441398
## acceleration -0.04053379
```

```r
lda.pred=predict(lda.fit,test)
lda.class=lda.pred$class
# Confusion matrix
table(lda.class ,test$mpg01)
```

```
##
## lda.class  0  1
##         0  5 11
##         1  0 62
```

```r
# Overall fraction of correct predictions
lda_acc <- mean(lda.class == test$mpg01)
lda_acc
```

```
## [1] 0.8589744
```

```r
# Test error for LDA
1-lda_acc
```

```
## [1] 0.1410256
```

```r
## e) QDA

qda.fit=qda(mpg01~cylinders+displacement+horsepower+weight+acceleration,
            data=training)
qda.fit
```

```
## Call:
## qda(mpg01 ~ cylinders + displacement + horsepower + weight +
##     acceleration, data = training)
##
## Prior probabilities of groups:
##         0         1
## 0.6082803 0.3917197
##
## Group means:
##    cylinders displacement horsepower     weight acceleration
## 0  0.7699698     0.7651627   0.688721  0.7687844    -0.365302
## 1 -0.7818856    -0.7861089  -0.671833 -0.8180198     0.332754
```

```r
qda.pred=predict(qda.fit,test)
qda.class=qda.pred$class
# Confusion matrix
table(qda.class ,test$mpg01)
```

```
##
## qda.class  0  1
##         0  5 11
##         1  0 62
```

```r
# Overall fraction of correct predictions
qda_acc <- mean(qda.class == test$mpg01)
qda_acc
```

```
## [1] 0.8589744
```

```r
# Test error for QDA
1-qda_acc
```

```
## [1] 0.1410256
```

```r
## f) Logistic Regression

glm.fit=glm(mpg01~cylinders+displacement+horsepower+weight+acceleration,
            data=training,family=binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = mpg01 ~ cylinders + displacement + horsepower +
##     weight + acceleration, family = binomial, data = training)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -2.3245  -0.2535  -0.0134   0.2985   3.6850
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.6739     0.3362  -4.979 6.39e-07 ***
## cylinders     -0.2858     0.7886  -0.362   0.7171
## displacement  -1.1261     1.1715  -0.961   0.3365
## horsepower    -2.2804     0.9551  -2.388   0.0170 *
## weight        -1.4998     0.8714  -1.721   0.0852 .
## acceleration  -0.2551     0.3891  -0.656   0.5121
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 420.45  on 313  degrees of freedom
## Residual deviance: 151.72  on 308  degrees of freedom
## AIC: 163.72
##
## Number of Fisher Scoring iterations: 7
```

```r
glm.probs=predict(glm.fit,newdata = test, type="response")
glm.pred=rep(0,dim(test)[1])
glm.pred[glm.probs>0.5]=1

# Confusion matrix
table(glm.pred,test$mpg01)
```

```
##
## glm.pred  0  1
##        0  5 11
##        1  0 62
```

```r
# Overall fraction of correct predictions
logistic_acc <- mean(glm.pred == test$mpg01)
logistic_acc
```

```
## [1] 0.8589744
```

```r
# test error rate
1-logistic_acc
```

```
## [1] 0.1410256
```

```r
## g) KNN with different k-values: 1, 2, 3, 5, 7
test.x=cbind(test$cylinders, test$displacement, test$horsepower,
             test$weight, test$acceleration)
training.x=cbind(training$cylinders, training$displacement, training$horsepower,
                 training$weight, training$acceleration)
library(class)

# i) k=1
knn.pred=knn(training.x,test.x,training$mpg01,k=1)

# Confusion matrix
table(knn.pred ,test$mpg01)
```

```
##
## knn.pred  0  1
##        0  5 16
##        1  0 57
```

```r
# Overall fraction of correct predictions
knn_acc <- mean(knn.pred == test$mpg01)
knn_acc
```

```
## [1] 0.7948718
```

```r
# Test error rate
k_1_error <- 1-knn_acc
k_1_error
```

```
## [1] 0.2051282
```

```r
# ii) k=2

knn.pred=knn(training.x,test.x,training$mpg01,k=2)

# Confusion matrix
table(knn.pred ,test$mpg01)
```

```
##
## knn.pred  0  1
##        0  5 15
##        1  0 58
```

```r
# Overall fraction of correct predictions
knn_acc <- mean(knn.pred == test$mpg01)
knn_acc
```

```
## [1] 0.8076923
```

```r
# Test error rate
k_2_error <- 1-knn_acc
k_2_error
```

```
## [1] 0.1923077
```

```r
# iii) k=3

knn.pred=knn(training.x,test.x,training$mpg01,k=3)

# Confusion matrix
table(knn.pred ,test$mpg01)
```

```
##
## knn.pred  0  1
##        0  5 14
##        1  0 59
```

```r
# Overall fraction of correct predictions
knn_acc <- mean(knn.pred == test$mpg01)
knn_acc
```

```
## [1] 0.8205128
```

```r
# Test error rate
k_3_error <- 1-knn_acc
k_3_error
```

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

```r
# iv) k=5

knn.pred=knn(training.x,test.x,training$mpg01,k=5)

# Confusion matrix
table(knn.pred ,test$mpg01)
```

```
##
## knn.pred  0  1
##        0  5 13
##        1  0 60
```

```r
# Overall fraction of correct predictions
knn_acc <- mean(knn.pred == test$mpg01)
knn_acc
```

```
## [1] 0.8333333
```

```r
# Test error rate
k_5_error <- 1-knn_acc
k_5_error
```

```
## [1] 0.1666667
```

```r
# v) k=7

knn.pred=knn(training.x,test.x,training$mpg01,k=7)

# Confusion matrix
table(knn.pred ,test$mpg01)
```

```
##
## knn.pred  0  1
##        0  5 13
##        1  0 60
```

```r
# Overall fraction of correct predictions
knn_acc <- mean(knn.pred == test$mpg01)
knn_acc
```

```
## [1] 0.8333333
```

```r
# Test error rate
k_7_error <- 1-knn_acc
k_7_error
```

```
## [1] 0.1666667
```

```r
# *) comparing all k values
cbind(k_1_error, k_2_error, k_3_error, k_5_error, k_7_error)
```

```
##      k_1_error k_2_error k_3_error k_5_error k_7_error
## [1,] 0.2051282 0.1923077 0.1794872 0.1666667 0.1666667
```

```
# As we can see, k=2 gives the lowest error rate
```