HUDM6026 Homework_09

Chenguang Pan

Mar 31, 2023

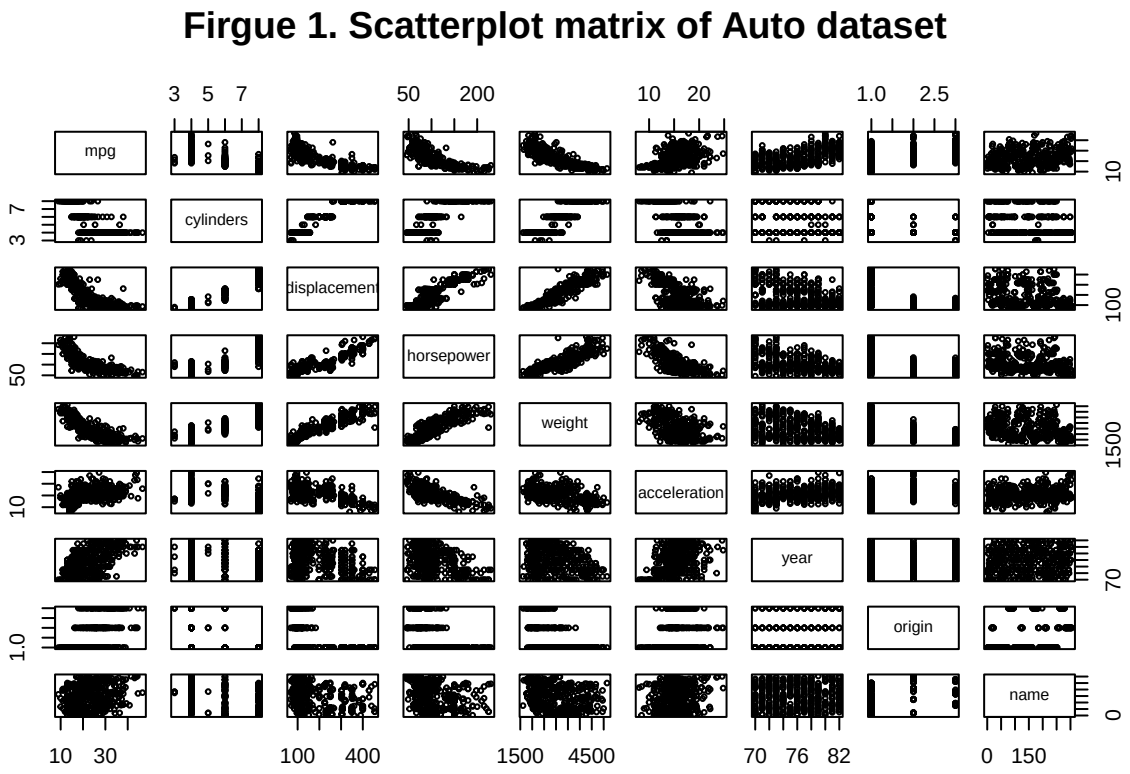
Q1: Chapter 3.9

Chapter 3, problem 9.Chapter 4, problems 1, 6, 13 This question involves the use of multiple linear regression on the Auto data set.

(a)Produce a scatterplot matrix which includes all of the variables in the data set.

MY SOLUTION:

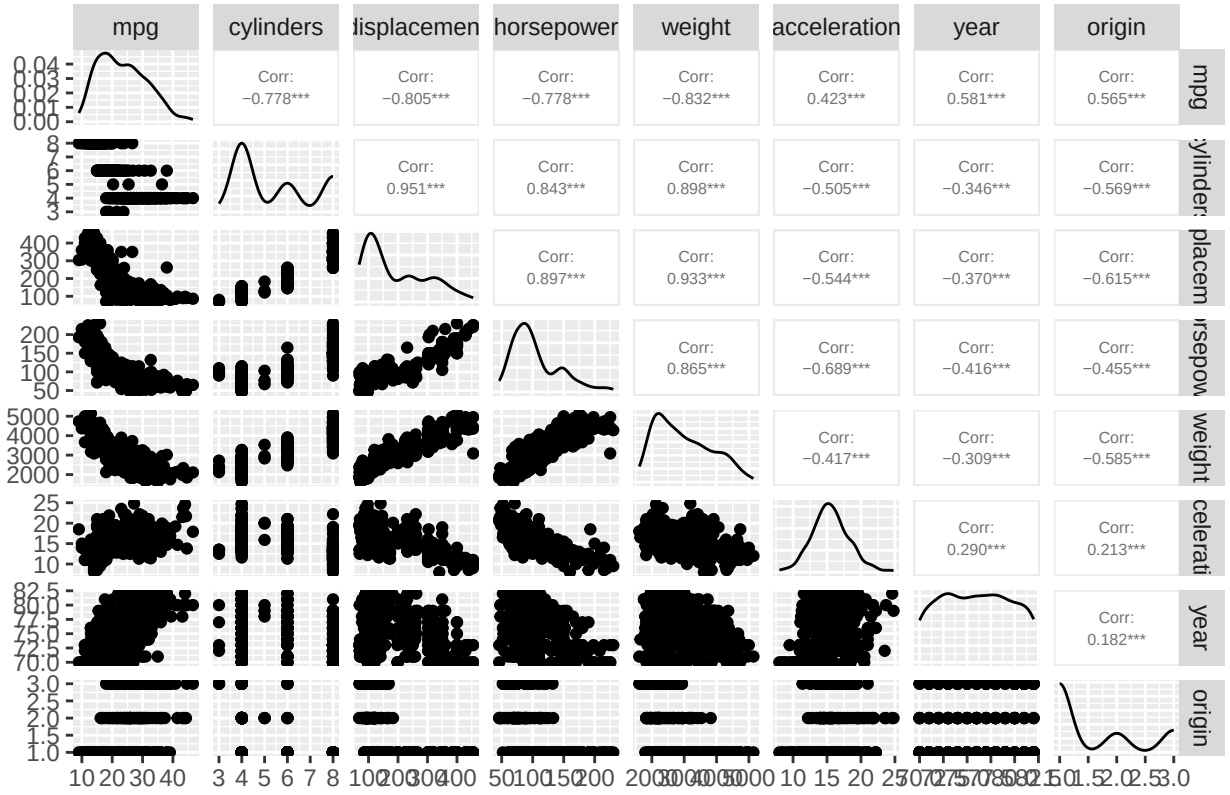
```
> # import the package and the data
> library(ISLR2)
> data(Auto)
> pairs(Auto,cex=0.5,
+       main = "Firgue 1. Scatterplot matrix of Auto dataset")
```



Or, one can use the package `GGally` to make the scatterplot in a more informative way. But before running this function, one should drop the categorical variable `name` in case of warning. More details can be found at https://r-charts.com/correlation/ggpairs/ .

```
> library(GGally)
> colnames(Auto)
[1] "mpg"          "cylinders"    "displacement" "horsepower"   "weight"
[6] "acceleration" "year"         "origin"       "name"
> ggpairs(Auto[,-9], # drop the name variable
+         # adjust the font size in the upper panel
+         upper = list(continuous=wrap("cor", size = 2.0)),
+         # add main title
+         title ="Firgue 2. Scatterplot matrix of Auto dataset using ggpairs()" )
```

Firgue 2. Scatterplot matrix of Auto dataset using ggpairs()



(b)Compute the matrix of correlations between the variables using the function cor(). You will need to exclude the name variable, which is qualitative

MY SOLUTION:
To make the corealtion matrix looking good, I abbreviated the variables' names to save space.

```
> # save the matrix to cor_mat
> cor_mat <- round(cor(Auto[,-9]),3)
> # abbreviate the variables' names
> names <- abbreviate(colnames(Auto[,-9]),4)
> rownames(cor_mat) <- names
```

```
> colnames(cor_mat) <- names
> cor_mat
        mpg   cyln   dspl   hrsp   wght   accl   year   orgn
mpg   1.000 -0.778 -0.805 -0.778 -0.832  0.423  0.581  0.565
cyln -0.778  1.000  0.951  0.843  0.898 -0.505 -0.346 -0.569
dspl -0.805  0.951  1.000  0.897  0.933 -0.544 -0.370 -0.615
hrsp -0.778  0.843  0.897  1.000  0.865 -0.689 -0.416 -0.455
wght -0.832  0.898  0.933  0.865  1.000 -0.417 -0.309 -0.585
accl  0.423 -0.505 -0.544 -0.689 -0.417  1.000  0.290  0.213
year  0.581 -0.346 -0.370 -0.416 -0.309  0.290  1.000  0.182
orgn  0.565 -0.569 -0.615 -0.455 -0.585  0.213  0.182  1.000
```

(c) Use the lm() function to perform a multiple linear regression with mpg as the response and all other variables except name as the predictors. Use the summary() function to print the results. Comment on the output. For instance:

MY SOLUTION:

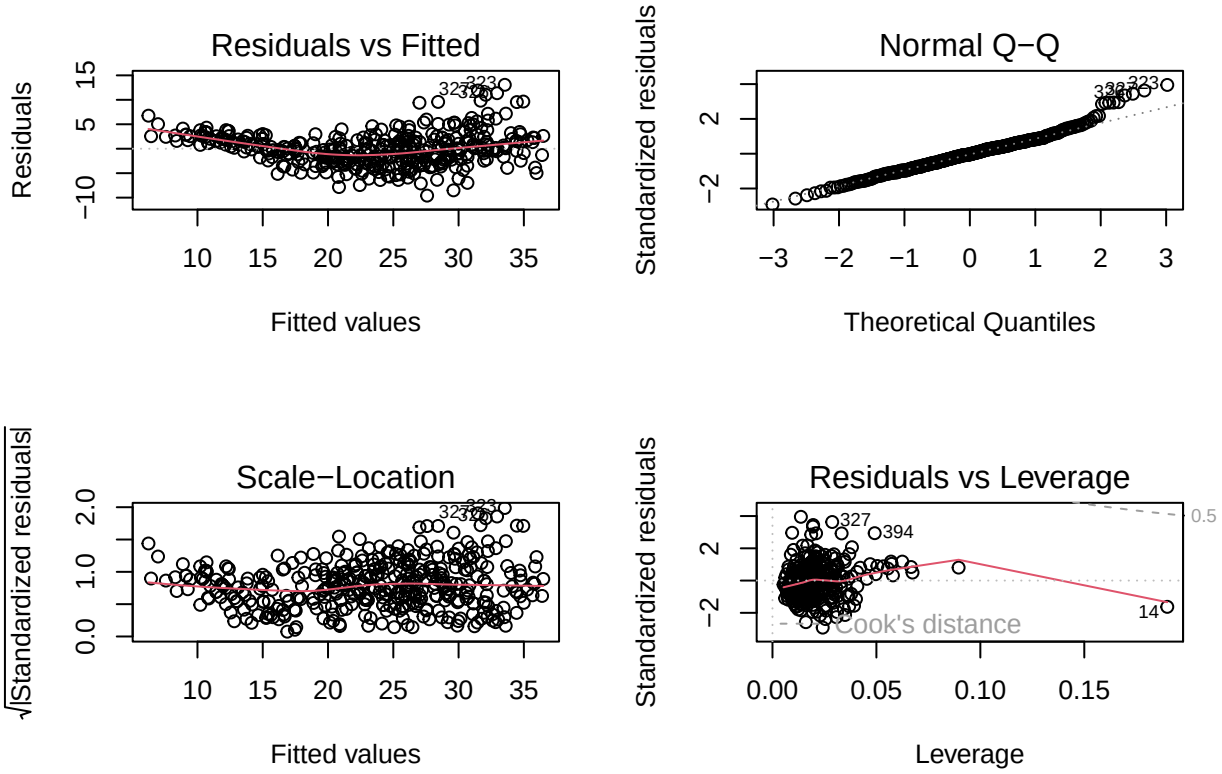
```
> # run the multiple linear regression model
> model_01 <- lm(mpg ~. -name, data = Auto )
> summary(model_01)

Call:
lm(formula = mpg ~ . - name, data = Auto)

Residuals:
    Min      1Q  Median      3Q     Max
-9.5903 -2.1565 -0.1169  1.8690 13.0604

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -17.218435   4.644294  -3.707  0.00024 ***
cylinders     -0.493376   0.323282  -1.526  0.12780
displacement   0.019896   0.007515   2.647  0.00844 **
horsepower    -0.016951   0.013787  -1.230  0.21963
weight        -0.006474   0.000652  -9.929  < 2e-16 ***
acceleration   0.080576   0.098845   0.815  0.41548
year           0.750773   0.050973  14.729  < 2e-16 ***
origin         1.426141   0.278136   5.127 4.67e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.328 on 384 degrees of freedom
Multiple R-squared:  0.8215,    Adjusted R-squared:  0.8182
F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

The multiple linear regression analysis results show that the overall model can explain 82.15% variance of the outcome variable `mpg`, and the mode predict the outcome well, $F(7, 384) = 252.4$, $p < .001$. The variable `displacement`, `weight`, `year`, and `origin` have a statistically significant relationship with the outcome. Specifically, one year increase (i.e., the) in `year` variable will be associated with .75 increase in the `mpg` after controlling for all other variables, $t = 14.729$, $p < .001$.

(d) Use the plot() function to produce diagnostic plots of the linear regression fit. Comment on any problems you see with the fit. Do the residual plots suggest any unusually large outliers? Does the leverage plot identify any observations with unusually high leverage?

MY SOLUTION:
Reference for this part https://www.andrew.cmu.edu/user/achoulde/94842/homework/regression_diagnostics.html.
http://www.h4labs.com/ml/islr/chapter03/03_09_melling.html

```
> par(mfrow=c(2,2))
> plot(model_01)
```



The *Residuals vs. Fitted plot* shows that the average value of the residuals at each value of fitted value (i.e., the red line) has a U-shape pattern, which might indicate that the data is not linear. In addition, the residuals presents non-equally distribution across the entire range of fitted values. It is a sign of non-constant variance.

The *normal Q-Q plot* shows that the residuals have a good normal distribution despite some data points at the tails.

The *scale-location plot* supports the findings from the *Residuals vs. Fitted plot* that there is non-equal variance. Also, it presents that there is no outlier since all values are within the range of [-2,2].

From the last plot, all data are well inside the Cook's distance lines (red-dotted line). Therefore, there is no influential case (high leverage data) in this dataset.

In conclusion, the diagnostic plots indicate the non-equal variance, no outliers, and not high leverage data.

(e)Use the and : symbols to fit linear regression models with interaction effects. Do any interactions appear to be statistically significant?

MY SOLUTION:

Tips: What is the difference between the symbol `*` and `:` in `lm()`?
The `*` symbol specifies that we want to include both the main effect and the interaction term. `lm(y ~ x * z)` equals `lm(y ~ x + z + x:z)`.
The `:` symbol specifies that we want to include only the interaction term. `lm(y ~ x : z)` equals `lm(y ~ x*z - x - z)`.

```
> colnames(Auto)
[1] "mpg"          "cylinders"    "displacement" "horsepower"   "weight"
[6] "acceleration" "year"         "origin"       "name"
> model_02 <- lm(mpg ~.-name + horsepower:weight, data = Auto)
> model_03 <- lm(mpg ~.-name + horsepower:origin, data = Auto)
> summary(model_02)

Call:
lm(formula = mpg ~ . - name + horsepower:weight, data = Auto)

Residuals:
   Min     1Q Median     3Q    Max
-8.589 -1.617 -0.184  1.541 12.001

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       2.876e+00  4.511e+00   0.638 0.524147
cylinders        -2.955e-02  2.881e-01  -0.103 0.918363
displacement      5.950e-03  6.750e-03   0.881 0.378610
horsepower       -2.313e-01  2.363e-02  -9.791  < 2e-16 ***
weight           -1.121e-02  7.285e-04 -15.393  < 2e-16 ***
acceleration     -9.019e-02  8.855e-02  -1.019 0.309081
year              7.695e-01  4.494e-02  17.124  < 2e-16 ***
origin            8.344e-01  2.513e-01   3.320 0.000986 ***
horsepower:weight 5.529e-05  5.227e-06  10.577  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.931 on 383 degrees of freedom
Multiple R-squared:  0.8618,    Adjusted R-squared:  0.859
F-statistic: 298.6 on 8 and 383 DF,  p-value: < 2.2e-16
> summary(model_03)

Call:
lm(formula = mpg ~ . - name + horsepower:origin, data = Auto)

Residuals:
   Min     1Q Median     3Q    Max
-9.277 -1.875 -0.225  1.570 12.080

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)      -2.196e+01  4.396e+00  -4.996 8.94e-07 ***
```

```
cylinders         -5.275e-01  3.028e-01  -1.742   0.0823 .
displacement      -1.486e-03  7.607e-03  -0.195   0.8452
horsepower         8.173e-02  1.856e-02   4.404 1.38e-05 ***
weight            -4.710e-03  6.555e-04  -7.186 3.52e-12 ***
acceleration      -1.124e-01  9.617e-02  -1.168   0.2434
year               7.327e-01  4.780e-02  15.328  < 2e-16 ***
origin             7.695e+00  8.858e-01   8.687  < 2e-16 ***
horsepower:origin -7.955e-02  1.074e-02  -7.405 8.44e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.116 on 383 degrees of freedom
Multiple R-squared:  0.8438,    Adjusted R-squared:  0.8406
F-statistic: 258.7 on 8 and 383 DF,  p-value: < 2.2e-16
```

The two models presents the statistically significant interactions between the `horsepower` and `weight`, also the `horsepower` and `origin`.

(f) Try a few different transformations of the variables

MY SOLUTION:

```
> model_04 <- lm(mpg ~.-name + I(horsepower^2)+I(horsepower^3),data = Auto)
> summary(model_04)

Call:
lm(formula = mpg ~ . - name + I(horsepower^2) + I(horsepower^3),
    data = Auto)

Residuals:
    Min      1Q  Median      3Q     Max
-8.5280 -1.6612 -0.0408  1.4789 11.9022

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)      1.326e+01  5.486e+00   2.417 0.016107 *
cylinders       -1.337e-01  3.242e-01  -0.412 0.680207
displacement    -3.516e-03  7.316e-03  -0.481 0.631083
horsepower      -6.284e-01  8.646e-02  -7.268 2.07e-12 ***
weight          -3.524e-03  6.697e-04  -5.262 2.39e-07 ***
acceleration    -2.994e-01  9.774e-02  -3.063 0.002343 **
year             7.429e-01  4.521e-02  16.432  < 2e-16 ***
origin           8.699e-01  2.528e-01   3.441 0.000643 ***
I(horsepower^2)  3.744e-03  7.134e-04   5.249 2.55e-07 ***
I(horsepower^3) -7.103e-06  1.831e-06  -3.881 0.000123 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.947 on 382 degrees of freedom
Multiple R-squared:  0.8607,    Adjusted R-squared:  0.8574
F-statistic: 262.3 on 9 and 382 DF,  p-value: < 2.2e-16
> model_05 <- lm(mpg ~.-name + log(horsepower),data = Auto)
```

```
> summary(model_05)

Call:
lm(formula = mpg ~ . - name + log(horsepower), data = Auto)

Residuals:
    Min      1Q  Median      3Q     Max
-8.5777 -1.6623 -0.1213  1.4913 12.0230

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)      8.674e+01  1.106e+01   7.839 4.54e-14 ***
cylinders       -5.530e-02  2.907e-01  -0.190 0.849230
displacement    -4.607e-03  7.108e-03  -0.648 0.517291
horsepower       1.764e-01  2.269e-02   7.775 7.05e-14 ***
weight          -3.366e-03  6.561e-04  -5.130 4.62e-07 ***
acceleration    -3.277e-01  9.670e-02  -3.388 0.000776 ***
year             7.421e-01  4.534e-02  16.368  < 2e-16 ***
origin           8.976e-01  2.528e-01   3.551 0.000432 ***
log(horsepower) -2.685e+01  2.652e+00 -10.127  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.959 on 383 degrees of freedom
Multiple R-squared:  0.8592,    Adjusted R-squared:  0.8562
F-statistic: 292.1 on 8 and 383 DF,  p-value: < 2.2e-16
```

Here, I tried to add the different transformations of `horsepower` variable. All models can explain the outcome well, as all p-values are less than .001. Besides, referring to the $adjusted R^2$, the `model_04` with the quadratic and cubic form with `horsepower` has the best performance, $adjusted R^2 = .8574$.

Q2: Chapter 4.1

Using a little bit of algebra, prove that (4.2) is equivalent to (4.3). In other words, the logistic function representation and logit represen- tation for the logistic regression model are equivalent

MY SOLUTION

For typing convenience, let $A = e^{\beta_0+\beta_1 X}$. Therefore, the formular (4.2) can be written as $p(X) = \frac{A}{1+A}$. This form-changing is similar to get the inverse function for $p(X)$. That is, we suppose that A is the function of $p(x)$. Thus, we can have

$$p(X) + p(X)A = A,$$
$$A[1 - p(X)] = p(X).$$

Then we have

$$A = \frac{p(X)}{1 - P(X)}.$$

After changing the A into the original form, finally we have

$$e^{\beta_0+\beta_1 X} = \frac{p(X)}{1 - p(X)}.$$

Q3: Chapter 4.6

Suppose we collect data for a group of students in a statistics class with variables X1 = hours studied, X2 = undergrad GPA, and Y = receive an A. We fit a logistic regression and produce estimated coefficient

(a)Estimate the probability that a student who studies for 40 h and has an undergrad GPA of 3.5 gets an A in the class.

MY SOLUTION

```
> prob_A  <- function(b0=-6, b1=0.05, b2=1,x1, x2){
+   # write the exponential part
+   exp_p <- exp(b0 + b1*x1 + b2*x2)
+   # write the logistic function
+   p_A <- exp_p/(1+exp_p)
+   return(p_A)
+ }
```

Plug the known values into the function above, we can have

```
> prob_A(x1=40,x2=3.5)
[1] 0.3775407
```

Therefore, the probability for that student to get an A is 0.378.

(b)Estimate the probability that a student who studies for 40 h and has an undergrad GPA of 3.5 gets an A in the class.

MY SOLUTION
To solve this question, we need to do some algebra. Rewrite the original function as following:

$$e^{\beta_0+\beta_1 x_1+\beta_2 x_2} = \frac{p(X)}{1-p(X)},$$

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = ln(\frac{p(X)}{1-p(X)}),$$

$$x_1 = \frac{ln(\frac{p(X)}{1-p(X)}) - \beta_0 - \beta_2 x_2}{\beta_1}.$$

Therefore, based on the last function, we plug all the known value,

```
> x1 <- (log(exp(1))-(-6)-1*(3.5))/0.05
> x1
[1] 70
```

The student need to spend 70 hours to have the 50% chance of getting an A. Note, if this is a real logistic model, the result should in the form of confidence interval rather than a single value.

Q4: Chapter 4.13

This question should be answered using the Weekly data set, which is part of the ISLR2 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.

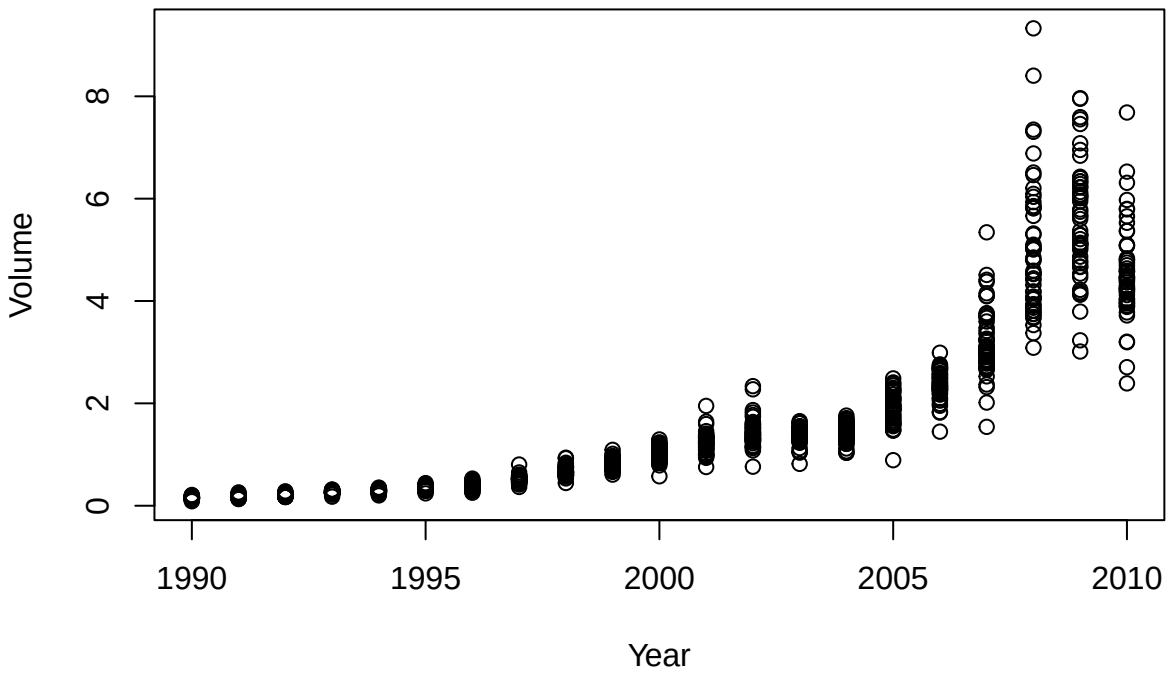
(a)Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?

MY SOLUTION

```
> library(ISLR2)
> attach(Weekly)
> dim(Weekly)
[1] 1089    9
> levels(Weekly$Direction)
[1] "Down" "Up"
> 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           Today         
 Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950  
 1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540  
 Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410  
 Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499  
 3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050  
 Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260  
 Direction
 Down:484
 Up  :605
```

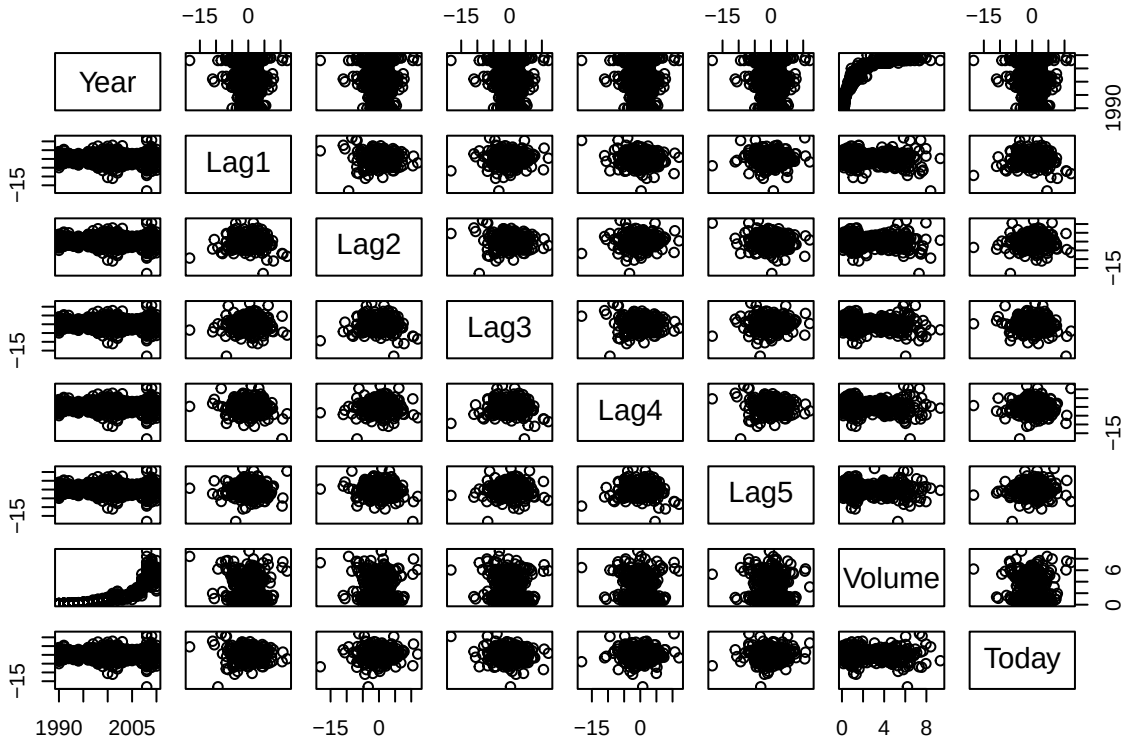
From the summary of each variable, there are 8 continuous variables and 1 categorical variable with "Down" and "Up" labels. Next, I checked the correlation matrix.

```
> attach(Weekly)
> round(cor(Weekly[,-9]),3)
         Year   Lag1   Lag2   Lag3   Lag4   Lag5 Volume  Today
Year    1.000 -0.032 -0.033 -0.030 -0.031 -0.031  0.842 -0.032
Lag1   -0.032  1.000 -0.075  0.059 -0.071 -0.008 -0.065 -0.075
Lag2   -0.033 -0.075  1.000 -0.076  0.058 -0.072 -0.086  0.059
Lag3   -0.030  0.059 -0.076  1.000 -0.075  0.061 -0.069 -0.071
Lag4   -0.031 -0.071  0.058 -0.075  1.000 -0.076 -0.061 -0.008
Lag5   -0.031 -0.008 -0.072  0.061 -0.076  1.000 -0.059  0.011
Volume  0.842 -0.065 -0.086 -0.069 -0.061 -0.059  1.000 -0.033
Today  -0.032 -0.075  0.059 -0.071 -0.008  0.011 -0.033  1.000
> plot(Year, Volume)
```



From the correlation matrix, one can find the only the variable `Year` and `Volume` showing the strong correlation. One possible explanation is that with the development of the global economy, the number of shares traded also increases.

```
> pairs(Weekly[,-9])
```



Here, I also checked the scatterplot matrix. It indicates that the strong correlation might only occurs on `Year` and `Volume`.

(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?

MY SOLUTION

```
> # run the logistic model using glm() function
> colnames(Weekly)
[1] "Year"      "Lag1"      "Lag2"      "Lag3"      "Lag4"      "Lag5"
[7] "Volume"    "Today"     "Direction"
> model_logi <- glm(Direction ~ Lag1+Lag2+Lag3+Lag4+Lag5+Volume,
+                   data = Weekly, family = binomial)
> summary(model_logi)

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
```

The logistic regression analysis shows that the variable `Lag2` is a statistically significant predictor $p = .030$. That is, one unit increase in the percentage returns of two days before(i.e., `Lag2`) will be associated with .058 increase in the log-odds of `Up` vs. `Down`, $z = 2.175$, $p = .030$, holding all other variables constant.

(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.

MY SOLUTION

First, use the `predict()` function to get the model-estimated values of the outcome. Note, the argument `type = "response"` option tells R to output probabilities of the form P(Y = 1|X), as opposed to other information such as the logit.

```
> # get the predicted value of the outcome
> model_logi_est <- predict(model_logi, type = "response")
> round(model_logi_est[1:20],3)
    1     2     3     4     5     6     7     8     9    10    11    12    13
0.609 0.601 0.588 0.482 0.617 0.568 0.579 0.515 0.572 0.555 0.609 0.537 0.513
   14    15    16    17    18    19    20
0.586 0.625 0.548 0.499 0.593 0.524 0.591
```

The estimated values looks good. Next to change the probabilities into two direction.

```
> dim(Weekly)
[1] 1089    9
> # create a vector of 1089 label "Down"
> pred_direct <- rep("Down", 1089)
> # change the Down to up if the probability is greater than .50
> pred_direct[model_logi_est > .50] <- "Up"
> # make a confusion matrix to compare the results
> table(pred_direct, Weekly$Direction)

pred_direct Down  Up
       Down   54  48
```

```
       Up    430 557
> # we can also to get the match rate by
> accuracy <- length(pred_direct[which(pred_direct == Weekly$Direction)])/ nrow(Weekly)
> accuracy
[1] 0.5610652
```

In this case, logistic regression correctly predicted the movement of the market 0.561 of the time.

(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).

MY SOLUTION

```
> attach(Weekly)
> # create a vector with T or F values at the length of dataset
> train <- (Year < 2009)
> # use this vector to subset the train data
> df_train <- Weekly[train,]
> df_test <- Weekly[!train,]
> # re-run the logistic regression model using only one predictor
> model_logi2 <- glm(Direction~ Lag2, data = df_train, family = binomial)
> # get the predicted value on the test dataset.
> model_logi_est2 <- predict(model_logi2, newdata=df_test, type = "response")
> # change the lable
> pred_direct2 <- ifelse(model_logi_est2 >0.5, "Up", "Down")
> # make a confusion matrix to compare the results
> table(pred_direct2, df_test$Direction)

pred_direct2 Down Up
        Down    9  5
        Up     34 56
> # we can also to get the match rate by
> accuracy <- length(pred_direct2[which(pred_direct2 == df_test$Direction)])/ nrow(df_test)
> accuracy
[1] 0.625
```

In this case, logistic regression correctly predicted the movement of the market 0.625 of the time.

(e)Repeat (d) using LDA

MY SOLUTION

Note LDA/QDA/Naive Bayes all belong to Bayesian Classifier Method with different distribution assumptions.

The LDA classifier results from assuming that the observation within each class come from a normal distribution with a class-specific mean and a common variance, and plugging estimates for these parameters into the Bayesian classifier.

```
> library(MASS)
> # run the LDA model
> lda_fit <- lda(Direction~ Lag2, data= df_train)
```

```
> # using the esitimated result to run on the test dataset
> # get the predicted value on the test dataset.
> lda_pred<- predict(lda_fit, newdata=df_test)
> # change the lable
> lda_class <- lda_pred$class
> table(lda_class, df_test$Direction)

lda_class Down Up
     Down    9  5
     Up     34 56
> mean(lda_class == df_test$Direction)
[1] 0.625
```

In this case, LDA correctly predicted the movement of the market 0.625 of the time.

(f)Repeat (d) using QDA

MY SOLUTION
running QDA is similar to LDA.

```
> # run the QDA model
> library(MASS)
> qda_fit <- qda(Direction~ Lag2, data= df_train)
> # using the esitimated result to run on the test dataset
> # get the predicted value on the test dataset.
> qda_pred<- predict(qda_fit, newdata=df_test)
> # change the lable
> qda_class <- qda_pred$class
> table(qda_class, df_test$Direction)

qda_class Down Up
     Down    0  0
     Up     43 61
> mean(qda_class == df_test$Direction)
[1] 0.5865385
```

In this case, QDA correctly predicted the movement of the market 0.587 of the time.

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

MY SOLUTION
Note, `knn()` function requires four inputs:
- A matrix containing the predictors associated with the training data;
- A matrix containing the predictors associated with the data for which we wish to make predictions;
- A vector containing the class labels for the training observations;
- A value for K, the number of nearest neighbors to be used by the classifier.

```
> library(class)
> # extract the required inputs
> train.X <- df_train$Lag2
> test.X <- df_test$Lag2
```

```
> train.Direction <- df_train$Direction
>
> #  run KNN
> knn.pred <- knn(as.matrix(train.X), as.matrix(test.X), train.Direction, k = 1)
> result <- round(mean(knn.pred == df_test$Direction),3)
> result
[1] 0.5
```

In this case, KNN correctly predicted the movement of the market 0.5 of the time.

(h)Repeat (d) using Naive Bayes

MY SOLUTION
Remember instead of assuming that these functions belong to a particular family of distributions, Naive Bayes make a simple assumption: Within the kth class, the p predictors are independent.

```
> library(e1071)
> nb_fit <- naiveBayes(Direction ~ Lag2,data=df_train)
> nb_class <- predict(nb_fit, newdata=df_test)
> table(nb_class, df_test$Direction)

nb_class Down Up
    Down    0  0
    Up     43 61
> result <- mean(nb_class == df_test$Direction)
> result
[1] 0.5865385
```

In this case, Naive Bayes Classifier correctly predicted the movement of the market 0.587 of the time.

(i)Which of these methods appears to provide the best results on this data

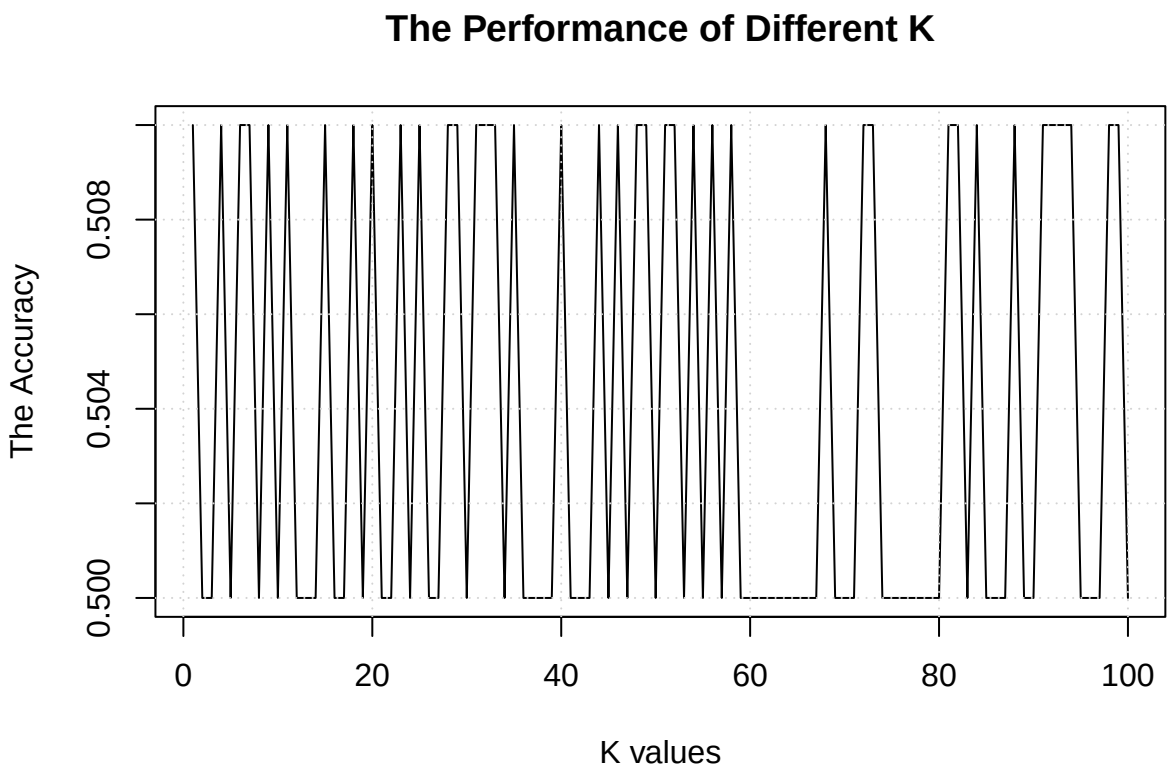
MY SOLUTION
The results show that the logistic regression and LDA has the best performance.

(j)Experiment with different combinations of predictors, including possible transformations and interactions, for each of the methods.

MY SOLUTION
I try the KNN with different K first.

```
> y_pre <- c()
> range_ <- c(1:100)
> for (x in range_){
+   knn.pred <- knn(as.matrix(train.X), as.matrix(test.X), train.Direction, k = 1)
+   result <- round(mean(knn.pred == df_test$Direction),3)
+   y_pre <- c(y_pre, result)
+ }
> plot(range_, y_pre,
+      type = "l",# to draw a line rather than a spot
+      main = "The Performance of Different K",
```

```
+      xlab = "K values",
+      ylab = "The Accuracy")
> grid()
```



As shown in the plot, with different K, the best accuracy that KNN can reach is identical. Therefore, I choose K = 1 for parsimony concern.

Next, I will try re-run several model by adding interaction of `Lag2` and `Volume` since the percentage return in the previous days may affect the investors confidence. Therefore, the trading volume `Volume` might depend on that. In addition, I arbitrarily choose to add the quadratic and cubic form of volume into the model. I will consider the predictive accuracy as the ultimate performance criterion.

```
> # re-run the logistic regression analysis
> attach(Weekly)
> # re-run the logistic regression model using only one predictor
> logi_inter <- glm(Direction~ Lag2 + Volume + Lag2:Volume+ I(Volume^2) + I(Volume^3), data = df_train,
> # get the summary of the estimation
> summary(logi_inter)

Call:
glm(formula = Direction ~ Lag2 + Volume + Lag2:Volume + I(Volume^2) +
    I(Volume^3), family = binomial, data = df_train)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.502  -1.260   1.020   1.088   1.538
```

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.321744   0.139137   2.312   0.0208 *
Lag2         0.048740   0.040485   1.204   0.2286
Volume      -0.163543   0.233817  -0.699   0.4843
I(Volume^2)  0.038516   0.088054   0.437   0.6618
I(Volume^3) -0.003066   0.008444  -0.363   0.7165
Lag2:Volume  0.002621   0.014104   0.186   0.8526
---
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: 1349.1  on 979  degrees of freedom
AIC: 1361.1

Number of Fisher Scoring iterations: 4
> # get the predicted value on the test dataset.
> logi_est <- predict(logi_inter, newdata=df_test, type = "response")
> # change the lable
> logi_direct <- ifelse(logi_est >0.5, "Up", "Down")
> # make a confusion matrix to compare the results
> table(logi_direct, df_test$Direction)

logi_direct Down Up
       Down   12 13
       Up     31 48
> # we can also to get the match rate by
> accuracy <- length(logi_direct[which(logi_direct == df_test$Direction)])/ nrow(df_test)
> accuracy
[1] 0.5769231
```

In this case, logistic regression correctly predicted the movement of the market 0.577 of the time.

Next, I re-run the LDA, QDA, and Naive Bayes model.

```
> # run the LDA model
> library(MASS)
> lda_fit <- lda(Direction~ Lag2 + Volume + Lag2:Volume+ I(Volume^2) + I(Volume^3), data= df_train)
> # using the esitimated result to run on the test dataset
> # get the predicted value on the test dataset.
> lda_pred<- predict(lda_fit, newdata=df_test)
> # change the lable
> lda_class <- lda_pred$class
> table(lda_class, df_test$Direction)

lda_class Down Up
     Down   13 13
     Up     30 48
> mean(lda_class == df_test$Direction)
[1] 0.5865385
```

In this case, LDA correctly predicted the movement of the market 0.587 of the time.

```
> # run the QDA model
> library(MASS)
> qda_fit <- qda(Direction~ Lag2 + Volume + Lag2:Volume+ I(Volume^2) + I(Volume^3), data= df_train)
> # using the esitimated result to run on the test dataset
> # get the predicted value on the test dataset.
> qda_pred<- predict(qda_fit, newdata=df_test)
> # change the lable
> qda_class <- qda_pred$class
> table(qda_class, df_test$Direction)

qda_class Down Up
     Down   31 45
     Up     12 16
> mean(qda_class == df_test$Direction)
[1] 0.4519231
```

In this case, QDA correctly predicted the movement of the market 0.452 of the time.

Note, the Bayes classier cannot handle the interaction term! I removed the interaction term in the model.

```
> library(e1071)
> nb_fit <- naiveBayes(Direction~ Lag2 + Volume + I(Volume^2) + I(Volume^3),data=df_train)
> nb_class <- predict(nb_fit, newdata=df_test)
> table(nb_class, df_test$Direction)

nb_class Down Up
    Down   43 57
    Up      0  4
> result <- mean(nb_class == df_test$Direction)
> result
[1] 0.4519231
```

In this case, Naive Bayes Classifier correctly predicted the movement of the market 0.452 of the time.

In conclusion, LDA has the best predictive accuracy 0.587 among these models. Comparing to the models in previous section, selecting the appropriate predictors in the right form is critical for model's performance. One should notice that it might not be a good choice to add as many as possible predictors to your model.

This dataset is obviously in a time-series form. Maybe some more advanced and flexible methods can be tested on this dataset, like long short-term memory networks (LSTM) model, although the model will become less interpretable. If I finished my class works, I will try run this analysis on Google Colab via Python.