Cpts 575 Data Science
Assignment 5
Jinyang Ruan
011696096
10/26/2021

1. Lode necessary libraries:

```{r}
library(kableExtra)
library(stringi)
library(textreadr)
library(tm)
library(SnowballC)
library(caret)
library(naivebayes)
```

Load data set and present all values in the appropriate types.

```{r}
cars = read.csv("D:/WSU Graduate/CPT_S 575 Data Science/cars.csv")
kable(head(cars), format = "latex", booktabs = T, caption="Dataset of cars.csv")%>%
    kable_styling(latex_options = c("hold_position"))
cars
```

knit_asis

data.frame 406 x 9

Description: df [406 x 9]

Car <chr>	MPG <dbl>	Cylinders <int>	Displacement <dbl>	Horsepower <int>	Weight <int>	Acceleration <dbl>
Chevrolet Chevelle Malibu	18.0	8	307.0	130	3504	12.0
Buick Skylark 320	15.0	8	350.0	165	3693	11.5
Plymouth Satellite	18.0	8	318.0	150	3436	11.0
AMC Rebel SST	16.0	8	304.0	150	3433	12.0
Ford Torino	17.0	8	302.0	140	3449	10.5
Ford Galaxie 500	15.0	8	429.0	198	4341	10.0
Chevrolet Impala	14.0	8	454.0	220	4354	9.0
Plymouth Fury iii	14.0	8	440.0	215	4312	8.5
Pontiac Catalina	14.0	8	455.0	225	4425	10.0
AMC Ambassador DPL	15.0	8	390.0	190	3850	8.5

1-10 of 406 rows | 1-7 of 9 columns   Previous 1 2 3 4 5 6 ... 41 Next

a. Perform a multiple linear regression with MPG as the response and all other variables except Car as the predictors.

```{r}
LR = lm(MPG ~ . - Car, data = cars)
summary (LR)
```

```
Call:
lm(formula = MPG ~ . - Car, data = cars)

Residuals:
     Min       1Q   Median       3Q      Max 
-28.3225  -2.0572   0.2173   2.2291  13.1024 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -2.266e+01  5.445e+00  -4.161 3.89e-05 ***
Cylinders    -3.549e-01  3.980e-01  -0.892  0.37318    
Displacement  2.192e-02  9.432e-03   2.324  0.02065 *  
Horsepower   -1.354e-02  1.353e-02  -1.001  0.31741    
Weight       -6.942e-03  7.833e-04  -8.862  < 2e-16 ***
Acceleration  1.395e-01  1.127e-01   1.238  0.21654    
Model         8.460e-01  6.309e-02  13.410  < 2e-16 ***
OriginJapan   1.040e+00  7.010e-01   1.484  0.13859    
OriginUS     -1.805e+00  6.937e-01  -2.602  0.00961 ** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.167 on 397 degrees of freedom
Multiple R-squared:  0.7588,	Adjusted R-squared:  0.754 
F-statistic: 156.1 on 8 and 397 DF,  p-value: < 2.2e-16
```

i) Which predictors appear to have a statistically significant relationship to the response, and how do you determine this?

From the results, we can know that the predictors {Displacement, Weight, Model, Origin} have a statistically significant relationship to the response MGP.

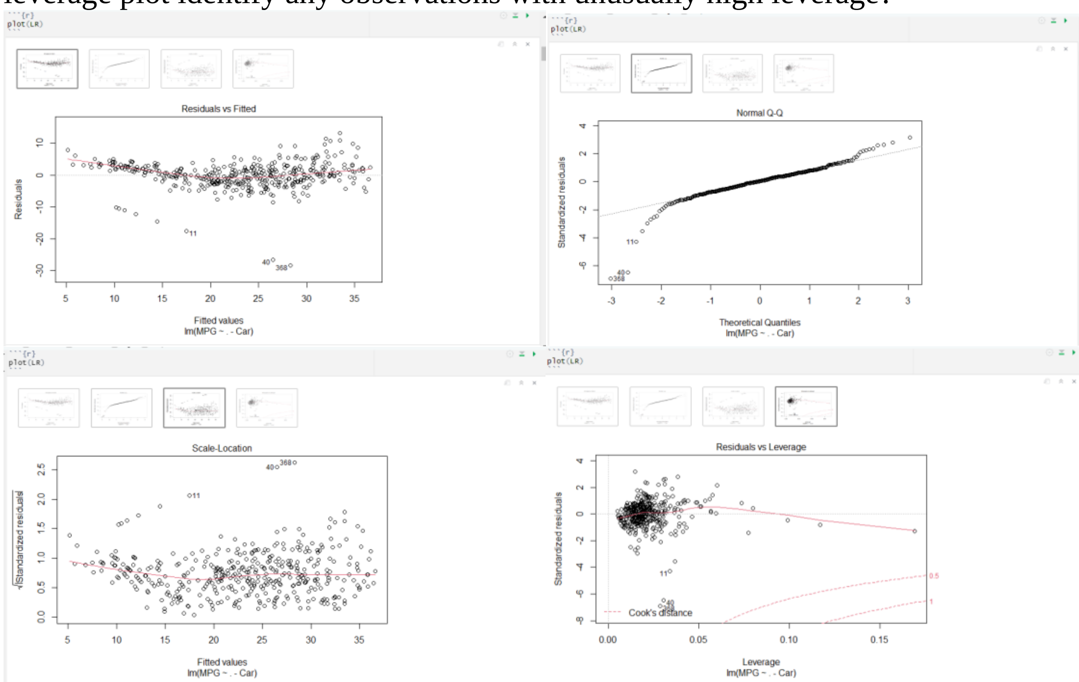
We can determine this though the p-Value which indicate how significant the relationship is (the number of stars).

ii) What does the coefficient for the Displacement variable suggest, in simple terms?

The coefficient of displacement is positive.

It suggests that how much the value of “MPG” will increase when the number of displacements increases by one while keeping all the other predictors constant.

b. 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?



Comments:

(1) Residuals vs Fitted: the points are almost around a horizontal line, little pattern in residuals.

(2) Normal Q-Q: the points in the Q–Q plot approximately lie on a line, so the distributions are linearly related.

(3) Scale - Location: That the red line is approximately horizontal. Then the average magnitude of the standardized residuals isn’t changing much as a function of the fitted values.

The spread around the red line doesn’t vary with the fitted values. Then the variability of magnitudes doesn’t vary much as a function of the fitted values.

(4) Residuals vs Leverage:

The rightmost point has a high leverage which means that it has a high influence, that is, it determines how much the predicted scores will change if the point is excluded.

c. Fit linear regression models with interaction effects. Do any interactions appear to be statistically significant?

```{r}
InterLR = lm(MPG ~ . - Car + Weight:Acceleration + Acceleration:Horsepower, data = cars)
summary(InterLR)
```

```
Call:
lm(formula = MPG ~ . - Car + Weight:Acceleration + Acceleration:Horsepower,
    data = cars)

Residuals:
     Min       1Q   Median       3Q      Max
-28.2430  -1.8230   0.3097   2.1362  12.4238

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)             -4.418e+01  7.603e+00  -5.811 1.28e-08 ***
Cylinders               -1.127e-01  4.078e-01  -0.276   0.7824
Displacement             6.796e-03  1.104e-02   0.616   0.5385
Horsepower              -1.584e-02  4.589e-02  -0.345   0.7302
Weight                   4.509e-04  2.474e-03   0.182   0.8555
Acceleration             1.310e+00  3.173e-01   4.129 4.45e-05 ***
Model                    8.752e-01  6.260e-02  13.981  < 2e-16 ***
OriginJapan              9.243e-01  7.015e-01   1.318   0.1884
OriginUS                -1.298e+00  7.158e-01  -1.814   0.0705 .
Weight:Acceleration     -4.118e-04  1.524e-04  -2.701   0.0072 **
Horsepower:Acceleration -4.568e-04  2.957e-03  -0.155   0.8773
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.094 on 395 degrees of freedom
Multiple R-squared:  0.7684,    Adjusted R-squared:  0.7625
F-statistic: 131.1 on 10 and 395 DF,  p-value: < 2.2e-16
```

Interaction such as weight: acceleration appears to be statistically significant.

2. Lode Boston data set:

```{r}
library(MASS)
Boston = MASS::Boston
kable(head(Boston), format = "latex", booktabs = T, caption="Dataset of Boston") %>%
    kable_styling(latex_options = c("hold_position"))
Boston
```

knit_asis

data.frame
506 x 14

Description: df [506 x 14]

	crim <dbl>	zn <dbl>	indus <dbl>	chas <int>	nox <dbl>	rm <dbl>	age <dbl>	dis <dbl>	rad <int>
1	0.00632	18.0	2.31	0	0.5380	6.575	65.2	4.0900	1
2	0.02731	0.0	7.07	0	0.4690	6.421	78.9	4.9671	2
3	0.02729	0.0	7.07	0	0.4690	7.185	61.1	4.9671	2
4	0.03237	0.0	2.18	0	0.4580	6.998	45.8	6.0622	3
5	0.06905	0.0	2.18	0	0.4580	7.147	54.2	6.0622	3
6	0.02985	0.0	2.18	0	0.4580	6.430	58.7	6.0622	3
7	0.08829	12.5	7.87	0	0.5240	6.012	66.6	5.5605	5
8	0.14455	12.5	7.87	0	0.5240	6.172	96.1	5.9505	5
9	0.21124	12.5	7.87	0	0.5240	5.631	100.0	6.0821	5
10	0.17004	12.5	7.87	0	0.5240	6.004	85.9	6.5921	5

1-10 of 506 rows | 1-10 of 14 columns

Previous 1 2 3 4 5 6 ... 51 Next

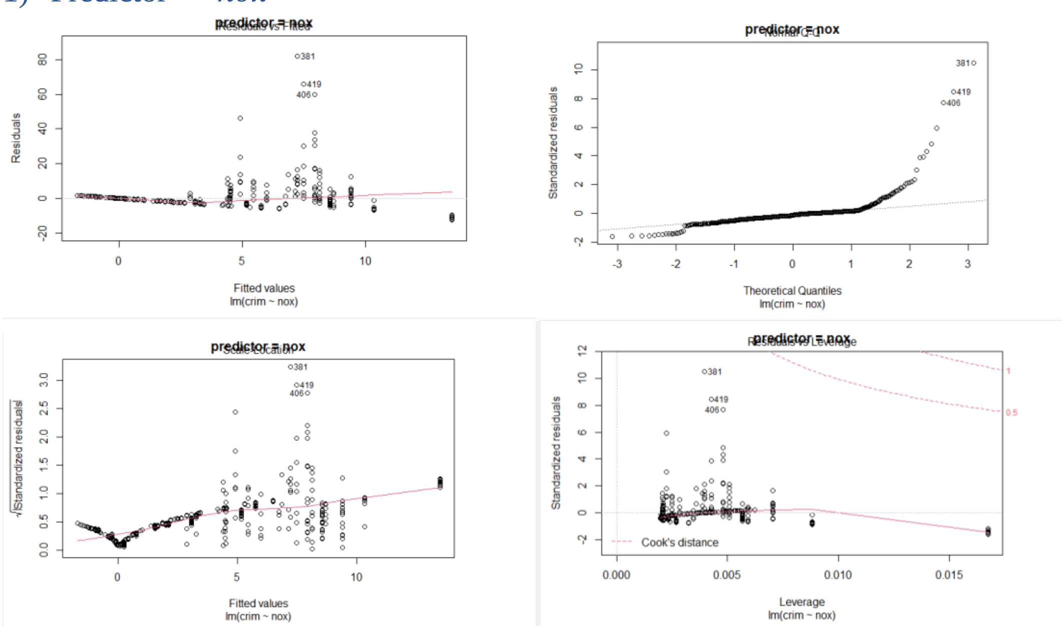
a. For each predictor, fit a simple linear regression model to predict the response. Include the code, but not the output for all models in your solution.

```{r}
LinearReg_zn      = lm(crim ~ zn,      data = Boston)
LinearReg_indus   = lm(crim ~ indus,   data = Boston)
LinearReg_chas    = lm(crim ~ chas,    data = Boston)
LinearReg_nox     = lm(crim ~ nox,     data = Boston)
LinearReg_rm      = lm(crim ~ rm,      data = Boston)
LinearReg_age     = lm(crim ~ age,     data = Boston)
LinearReg_dis     = lm(crim ~ dis,     data = Boston)
LinearReg_rad     = lm(crim ~ rad,     data = Boston)
LinearReg_tax     = lm(crim ~ tax,     data = Boston)
LinearReg_ptratio = lm(crim ~ ptratio, data = Boston)
LinearReg_black   = lm(crim ~ black,   data = Boston)
LinearReg_lstat   = lm(crim ~ lstat,   data = Boston)
LinearReg_medv    = lm(crim ~ medv,    data = Boston)
#summary (LinearReg_zn)
#summary (LinearReg_indus)
#summary (LinearReg_chas)
#summary (LinearReg_nox)
#summary (LinearReg_rm)
#summary (LinearReg_age)
#summary (LinearReg_dis)
#summary (LinearReg_rad)
#summary (LinearReg_tax)
#summary (LinearReg_ptratio)
#summary (LinearReg_black)
#summary (LinearReg_lstat)
#summary (LinearReg_medv)
```

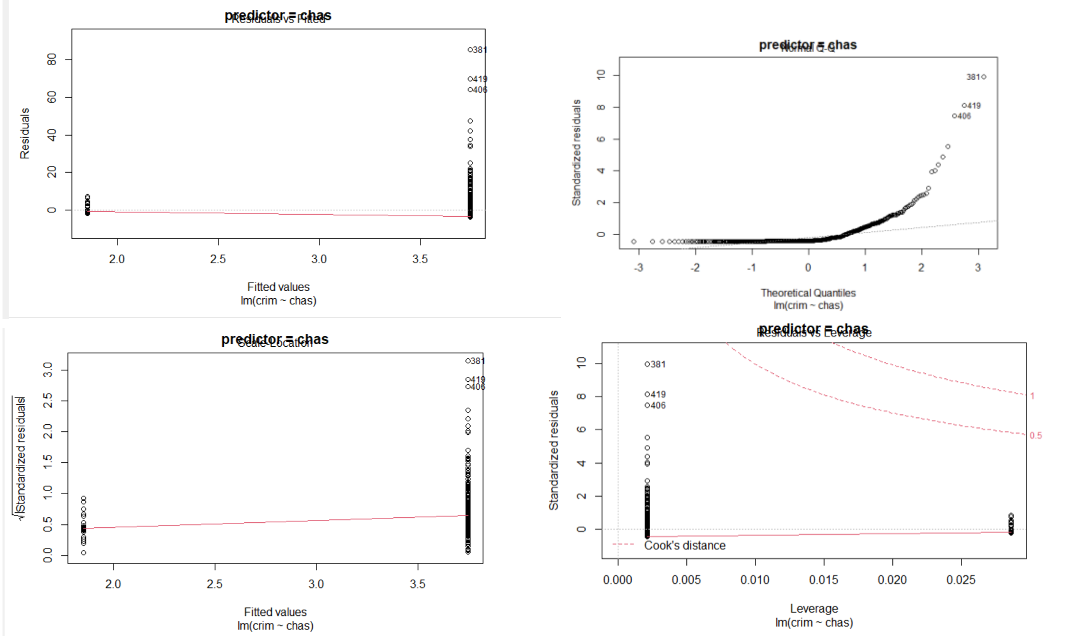
b. In which of the models is there a statistically significant association between the predictor and the response? Considering the meaning of each variable, discuss the relationship between $crim$ and nox, $chas$, rm, dis and $medv$ in particular. How do these relationships differ?

```{r}
plot(LinearReg_nox,  main = "predictor = nox")
plot(LinearReg_chas, main = "predictor = chas")
plot(LinearReg_rm,   main = "predictor = rm")
plot(LinearReg_dis,  main = "predictor = dis")
plot(LinearReg_medv, main = "predictor = medv")
```

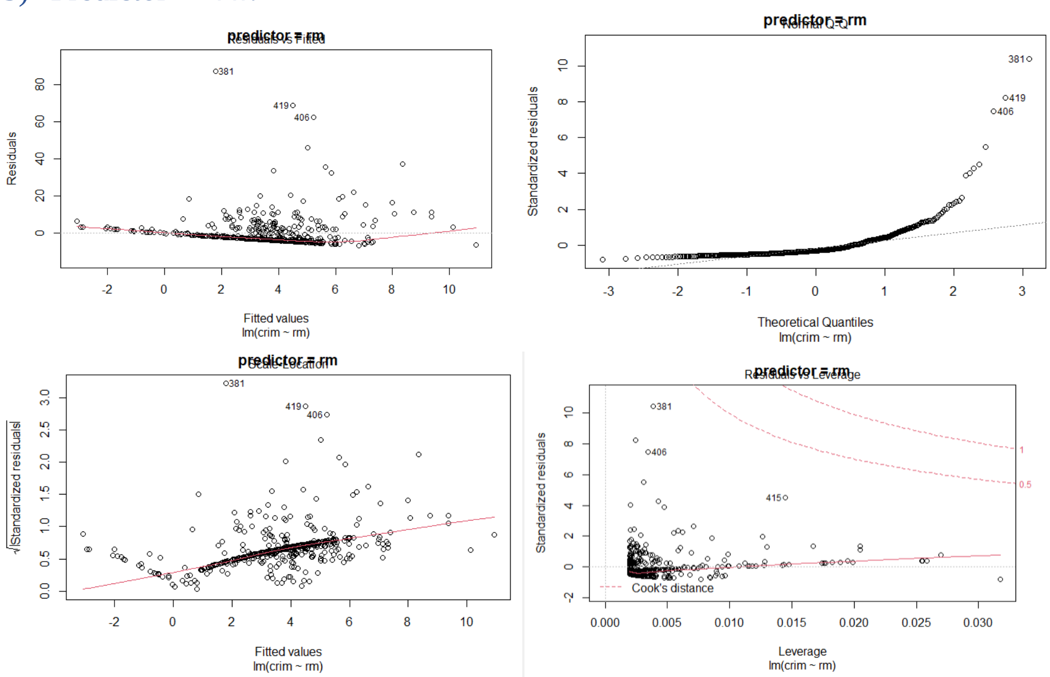
1) Predictor = "*nox*"



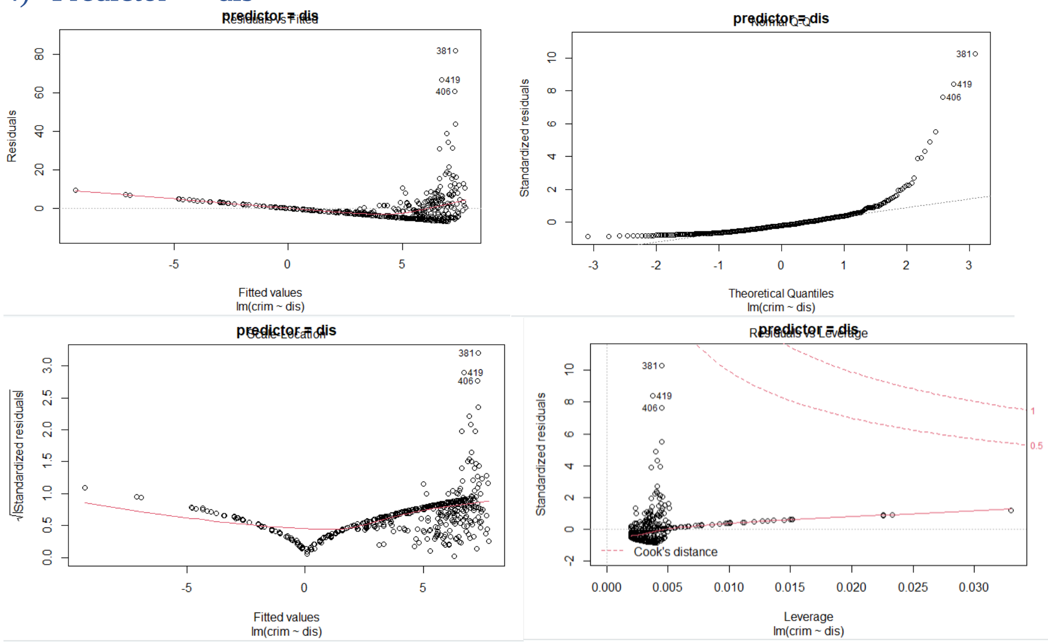
2) Predictor = "*chas*"



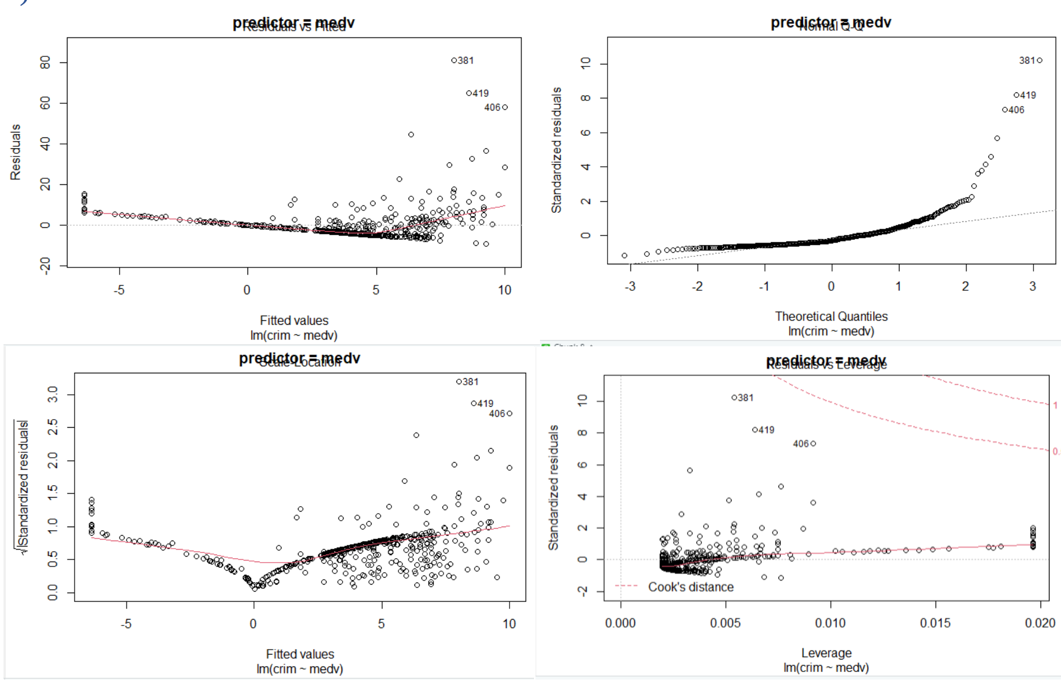
3) Predictor = "*rm*"



4) Predictor = “dis”



5) Predictor = “$medv$”




```{r}
Rsq_nox  = summary(LinearReg_nox)$r.squared
Rsq_chas = summary(LinearReg_chas)$r.squared
Rsq_rm   = summary(LinearReg_rm)$r.squared
Rsq_dis  = summary(LinearReg_dis)$r.squared
Rsq_medv = summary(LinearReg_medv)$r.squared
cat(Rsq_nox, Rsq_chas, Rsq_rm, Rsq_dis, Rsq_medv, sep=", ")
```

```
0.1772172, 0.003123869, 0.04806912, 0.1441494, 0.1507805
```

There is a statistically significant association between the predictor and the response for all variables except *chas*.

From the figures above, we can see that there is linear relationship between the nox and crim. And among all the four predictors, nox gets the highest R Squared value.

There is no association between $chas$ and $crim$. For the other four predictors nox, rm, dis, and $medv$, it is not a complete straight line in Residuals vs Fitted, so there is little pattern in residuals.

c. Fit a multiple regression model to predict the response using all the predictors. Describe your results. For which predictors can we reject the null hypothesis H0: βj = 0?

```
```{r}
LinearReg = lm(crim ~ . - crim, data = Boston)
summary(LinearReg)
```
```

```
Call:
lm(formula = crim ~ . - crim, data = Boston)

Residuals:
   Min     1Q Median     3Q    Max 
-9.924 -2.120 -0.353  1.019 75.051 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  17.033228   7.234903   2.354 0.018949 *  
zn            0.044855   0.018734   2.394 0.017025 *  
indus        -0.063855   0.083407  -0.766 0.444294    
chas         -0.749134   1.180147  -0.635 0.525867    
nox         -10.313535   5.275536  -1.955 0.051152 .  
rm            0.430131   0.612830   0.702 0.483089    
age           0.001452   0.017925   0.081 0.935488    
dis          -0.987176   0.281817  -3.503 0.000502 ***
rad           0.588209   0.088049   6.680 6.46e-11 ***
tax          -0.003780   0.005156  -0.733 0.463793    
ptratio      -0.271081   0.186450  -1.454 0.146611    
black        -0.007538   0.003673  -2.052 0.040702 *  
lstat         0.126211   0.075725   1.667 0.096208 .  
medv         -0.198887   0.060516  -3.287 0.001087 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.439 on 492 degrees of freedom
Multiple R-squared:  0.454,	Adjusted R-squared:  0.4396 
F-statistic: 31.47 on 13 and 492 DF,  p-value: < 2.2e-16
```

Comments:

i) There are five predictors including $\{zn, dis, rad, black, mdev\}$ have significant association with crim.

ii) R-squared value is higher for multiple regression when being compared to the simple regressions.

iii) For the predictors $\{zn, dis, rad, black, mdev\}$, p-values are all less than 0.05, we can reject these predictors.

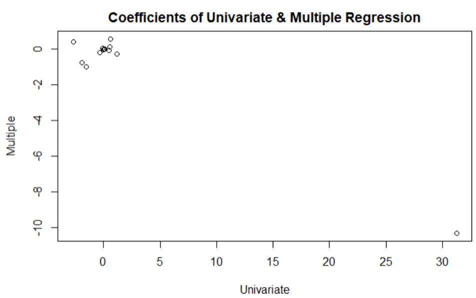
d. How do your results from (a) compare to your results from (c)?

````
```{r}
CoefSimple = vector("numeric", 0)
CoefSimple = c(LinearReg_zn$coefficient[2],    LinearReg_indus$coefficient[2],
               LinearReg_chas$coefficient[2],  LinearReg_nox$coefficient[2],
               LinearReg_rm$coefficient[2],    LinearReg_age$coefficient[2],
               LinearReg_dis$coefficient[2],   LinearReg_rad$coefficient[2],
               LinearReg_tax$coefficient[2],   LinearReg_ptratio$coefficient[2],
               LinearReg_black$coefficient[2], LinearReg_lstat$coefficient[2],
               LinearReg_medv$coefficient[2])
print (CoefSimple)

CoefMultiple = vector("numeric", 0)
CoefMultiple = c(LinearReg$coefficients[-1])
print (CoefMultiple)

plot(CoefSimple, CoefMultiple,
     main = "Coefficients of Univariate & Multiple Regression",
     xlab = "Uni-variate", ylab = "Multiple")
```
````

```
         zn       indus        chas         nox          rm         age         dis 
rad         tax 
-0.07393498  0.50977633 -1.89277655 31.24853120 -2.68405122  0.10778623 -1.55090168 
0.61791093  0.02974225 
    ptratio       black       lstat        medv 
 1.15198279 -0.03627964  0.54880478 -0.36315992 
          zn        indus         chas          nox           rm          age 
dis 
  0.044855215 -0.063854824 -0.749133611 -10.313534912  0.430130506  0.001451643 
-0.987175726 
          rad          tax      ptratio        black        lstat         medv 
  0.588208591 -0.003780016 -0.271080558 -0.007537505  0.126211376 -0.198886821 
```



(1) The regression coefficients are different in univariate and multiple regression. In univariate regression, we only consider the average effect of an increase in the specific predictor, while ignoring other predictors. In multiple regression, we consider the average effect of an increase in the predictor, while holding other predictors fixed.
(2) From the plot we can know the coefficient for most predictors are around 0 in both univariate and multiple regression.

e. Is there evidence of non-linear association between any of the predictors and the response?

````
```{r}
LinearReg_zn = lm(crim ~ poly(zn, 3), data = Boston)
LinearReg_indus = lm(crim ~ poly(indus, 3), data = Boston)
LinearReg_nox = lm(crim ~ poly(nox, 3), data = Boston)
LinearReg_rm = lm(crim ~ poly(rm, 3), data = Boston)
LinearReg_age = lm(crim ~ poly(age, 3), data = Boston)
LinearReg_dis = lm(crim ~ poly(dis, 3), data = Boston)
LinearReg_rad = lm(crim ~ poly(rad, 3), data = Boston)
LinearReg_tax = lm(crim ~ poly(tax, 3), data = Boston)
LinearReg_ptratio = lm(crim ~ poly(ptratio, 3), data = Boston)
LinearReg_black = lm(crim ~ poly(black, 3), data = Boston)
LinearReg_lstat = lm(crim ~ poly(lstat, 3), data = Boston)
LinearReg_medv = lm(crim ~ poly(medv, 3),  data = Boston)
#summary(LinearReg_zn)
#summary(LinearReg_indus)
#summary(LinearReg_nox)
#summary(LinearReg_rm)
#summary(LinearReg_age)
#summary(LinearReg_dis)
#summary(LinearReg_rad)
#summary(LinearReg_tax)
#summary(LinearReg_ptratio)
#summary(LinearReg_black)
#summary(LinearReg_lstat)
#summary(LinearReg_medv)
```
````

One output instance:

```
Call:
lm(formula = crim ~ poly(zn, 3), data = Boston)

Residuals:
   Min     1Q Median     3Q    Max
-4.821 -4.614 -1.294  0.473 84.130

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    3.6135     0.3722   9.709  < 2e-16 ***
poly(zn, 3)1 -38.7498     8.3722  -4.628  4.7e-06 ***
poly(zn, 3)2  23.9398     8.3722   2.859  0.00442 **
poly(zn, 3)3 -10.0719     8.3722  -1.203  0.22954
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 8.372 on 502 degrees of freedom
Multiple R-squared:  0.05824,	Adjusted R-squared:  0.05261
F-statistic: 10.35 on 3 and 502 DF,  p-value: 1.281e-06
```

Looking at the p-value, we can get the following observations:
(1) For predictors $\{zn, rm, rad, tax, lstat\}$, the cubic coefficient is not statistically significant
(2) For predictors $\{indus, nox, age, dis, ptratio, medv\}$, the adequacy of the cubic fit
(3) For predictor $\{black\}$, the quadratic and cubic coefficients are not statistically significant, there is no non-linear effect.

3. Suppose we collect data for a group of students in a statistics class with variables:
   X1 = hours studied,
   X2 = undergrad GPA,
   X3 = PSQI score (a sleep quality index), and
   Y = receive an A.
   We fit a logistic regression and produce estimated coefficient, $\beta0 = -7$, $\beta1 = 0.1$, $\beta2 = 1$, $\beta3 = -.04$.
   a. Estimate the probability that a student who studies for 30 h, has a PSQI score of 11 and has an undergrad GPA of 3.0 gets an A in the class. Show your work.
   $\hat{y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 = -7 + 0.1X_1 + X_2 - 0.04X_3$
   $\because X_1 = 30,\ X_2 = 3.0, X_3 = 11$
   $\therefore \hat{y} = -7 + 3 + 3 - 0.44 = -1.44$
   $$=> P(X) = \frac{e^{\hat{y}}}{1 + e^{\hat{y}}} = \frac{e^{-1.44}}{1 + e^{-1.44}} \approx 19.15\%$$
   b. How many hours would the student in part (a) need to study to have a 60 % chance of getting an A in the class? Show your work.
   Assume that the student needs to study h hours.
   Then we can get:
   $\hat{y} = -7 + 0.1h + 3 - 0.44 = 0.1h - 4.44$
   $=> P(X) = \frac{e^{(0.1h-4.44)}}{1+e^{(0.1h-4.44)}} = 60\%$
   $=> e^{0.1h-4.44} = 1.5$
   $=> h \approx (0.41 + 4.44) * 10 = 48.5$
   c. How many hours would a student with a 3.0 GPA and a PSQI score of 5 need to study to have a 50 % chance of getting an A in the class? Show your work.
   Assume that the student needs to study h hours.
   Then we can get:
   $\hat{y} = -7 + 0.1h + 3 - 0.2 = 0.1h - 4.2$

$$=> P(X) = \frac{e^{(0.1h-4.42)}}{1+e^{(0.1h-4.42)}} = 50\%$$
$$=> e^{0.1h-4.42} = 1$$
$$=> h = 4.42 * 10 = 44.2$$

4. For this question, you will use a naïve Bayes model to classify consumer complaints by the category of financial product or service the complaints are related to.

Data set prepare:

```{r}
complaints = read.csv("D:/WSU Graduate/CPT_S 575 Data Science/consumer_complaints.csv")
complaints$Product = as.factor(complaints$Product)
str (complaints)
```

```
'data.frame':   323229 obs. of  2 variables:
 $ Product           : Factor w/ 9 levels "Bank account or service",..: 8 2 2 2 7 8 8 2 2 2 ...
 $ Consumer_complaint: chr  "I contacted Ally on Friday XX/XX/XXXX after falling behind on payments due to being
out of work for a short per"| __truncated__ "" "" "" ...
```

a. Tokenization

```{r}
TokenDTM = VCorpus(VectorSource(complaints$Consumer_complaint)) %>%
           tm_map(removeNumbers) %>%
           tm_map(content_transformer(tolower)) %>%
           tm_map(removeWords, stopwords("english"))  %>%
           tm_map(stemDocument) %>%
           tm_map(stripWhitespace) %>%
           DocumentTermMatrix(control=list(wordLengths=c(3,30)))  %>%
           removeSparseTerms(0.99)
inspect(TokenDTM)
```

```
<<DocumentTermMatrix (documents: 323229, terms: 477)>>
Non-/sparse entries: 4882185/149298048
Sparsity           : 97%
Maximal term length: 12
Weighting          : term frequency (tf)
Sample             :
        Terms
Docs     {$.} account bank call check loan payment receiv told xxxx
  133593    5       4    1   17     0    1       3      1   10  178
  14944    11       5    6   12     2   28      22      3    2  138
  152802   81       4    4    0     0    0      83      2    2  265
  156820   17       1    0   55    10    1      12     14   55  248
  168573   69       2    2    5     4    0       6      5    1  295
  284797    0       4    4    1     0    0       3     12    0  887
  287577    0       0    0    0     0  853      46      0    2   70
  292477   13      11    0   13     0   59      23     18    8  247
  64849    58      88   23   21    30    0      10     25    6  121
  73151    30       1   14   14     5    0       0      0    6  197
```

Show a non-zero entries of a random row:

```{r}
TokenDf = as.data.frame(as.matrix(TokenDTM), stringsAsFactors=False)
# show a non-zero entries of a random row
Random = sample(1:nrow(TokenDf), 1, replace=FALSE)
RandomRow = TokenDf[Random,]
RandomRow[which(RandomRow != 0)]
```

Description: df [1 x 0]

116019

1 row

b. Classification

Reduce feature sets and remove correlated features:

```{r}
CorSet = cor(TokenDf) %>%
         findCorrelation(cutoff=0.5)
TokenDTM = TokenDTM[, -c(CorSet)]
```

Split data into a training set and a test set and Build Naïve Bayes classifier.

```{r}
convert_counts = function(x) {
  x <- ifelse(x > 0, "Yes", "No")
}

GetConfMatrix = function (Proportion, LowFreq) {
    #Split the original dataset and check the proportion of each categories
    Index = createDataPartition(complaints$Product, p = Proportion, list = FALSE, times = 1)

    # Proportion Of Consumer complaints categories in Train Data
    TrainComplaints = complaints[Index,]
    TrainLabels    = TrainComplaints$Product
    prop.table(table(TrainComplaints$Product))
    #Proportion of Article categories in Test Data
    TestComplaints = complaints[-Index,]
    TestLabels    = TestComplaints$Product
    prop.table(table(TestComplaints$Product))

    #Split the DTM
    TrainDtm = TokenDTM[Index,]
    TestDtm = TokenDTM[-Index,]
    Freq = findFreqTerms(TrainDtm, LowFreq)

    # Build classifier
    TrainDtm = TrainDtm[, Freq]
    TrainDtm = apply(TrainDtm, MARGIN = 2, convert_counts)
    Classifier = naive_bayes(TrainDtm, TrainLabels, laplace = 1)
    summary (Classifier)

    # Predict
    TestDtm = TestDtm[, Freq]
    TestDtm = apply(TestDtm, MARGIN = 2, convert_counts)
    Pred = predict(Classifier, TestDtm)

    # Return the confusion matrix
    ConfMatrix = confusionMatrix(Pred, TestLabels)
    return (ConfMatrix)
}
```

```{r}
ConfMatrix = GetConfMatrix (0.8, 10)
ConfMatrix
```

```
==================================================== Naive Bayes ====================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 258587
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0172
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

---------------------------------------------------------------------------------------------------------------------
```

```
Confusion Matrix and Statistics

                            Reference
Prediction                   Bank account or service Checking or savings account Consumer Loan
  Bank account or service                      14619                       13871          4630
  Checking or savings account                   1269                        3650           167
  Consumer Loan                                   37                          31           155
  Money transfers                                  3                           1             0
  Other financial service                          0                           0             0
  Payday loan                                      0                           1             5
  Student loan                                   285                         403           372
  Vehicle loan or lease                          468                        1067           753
  Virtual currency                               560                        1401           238
                            Reference
Prediction                   Money transfers Other financial service Payday loan Student loan
  Bank account or service                811                     153         849         7234
  Checking or savings account            130                      15          44          371
  Consumer Loan                            1                       1          26          135
  Money transfers                          2                       1           0            0
  Other financial service                  0                       0           0            0
  Payday loan                              0                       0           6            3
  Student loan                            25                      12         118         3287
  Vehicle loan or lease                   45                      18          52          692
  Virtual currency                        56                      11          13          940
                            Reference
Prediction                   Vehicle loan or lease Virtual currency
  Bank account or service                     2842                0
  Checking or savings account                  289                2
  Consumer Loan                                164                0
  Money transfers                                0                0
  Other financial service                        0                0
  Payday loan                                    0                0
  Student loan                                 366                0
  Vehicle loan or lease                       1512                0
  Virtual currency                             429                1

Overall Statistics

               Accuracy : 0.3594
                 95% CI : (0.3557, 0.3631)
    No Information Rate : 0.316
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.161

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bank account or service Class: Checking or savings account Class: Consumer Loan
Sensitivity                                  0.8479                            0.17870             0.024525
Specificity                                  0.3589                            0.94828             0.993227
Pos Pred Value                               0.3248                            0.61479             0.281818
Neg Pred Value                               0.8664                            0.71425             0.903810
Prevalence                                   0.2667                            0.31597             0.097769
Detection Rate                               0.2262                            0.05646             0.002398
Detection Prevalence                         0.6963                            0.09184             0.008508
Balanced Accuracy                            0.6034                            0.56349             0.508876
                     Class: Money transfers Class: Other financial service Class: Payday loan
Sensitivity                       1.869e-03                       0.000000          5.415e-03
Specificity                       9.999e-01                       1.000000          9.999e-01
Pos Pred Value                    2.857e-01                            NaN          4.000e-01
Neg Pred Value                    9.835e-01                       0.996736          9.829e-01
Prevalence                        1.655e-02                       0.003264          1.714e-02
Detection Rate                    3.094e-05                       0.000000          9.282e-05
Detection Prevalence              1.083e-04                       0.000000          2.320e-04
Balanced Accuracy                 5.009e-01                       0.500000          5.026e-01
                     Class: Student loan Class: Vehicle loan or lease Class: Virtual currency
Sensitivity                      0.25960                      0.26990               3.333e-01
Specificity                      0.96958                      0.94758               9.436e-01
Pos Pred Value                   0.67523                      0.32820               2.740e-04
Neg Pred Value                   0.84316                      0.93187               1.000e+00
Prevalence                       0.19588                      0.08666               4.641e-05
Detection Rate                   0.05085                      0.02339               1.547e-05
Detection Prevalence             0.07531                      0.07127               5.645e-02
Balanced Accuracy                0.61459                      0.60874               6.384e-01
```

```{r}
Accuracy    = ConfMatrix$overall['Accuracy']*100
sprintf("Proportion = 0.8, Accuracy = %.2f", Accuracy)
```

```
[1] "Proportion = 0.8, Accuracy = 35.94"
```

Try to get best splitting proportion by accuracy:

```{r}
PropSet  = c(0.6, 0.7, 0.9)
BsetProp = 0.8
for(prop in PropSet){
    CurConfMatrix = GetConfMatrix (prop, 10)
    CurAccuracy    = CurConfMatrix$overall['Accuracy']*100
    print (sprintf("Proportion = %.2f, Accuracy = %.2f", prop, CurAccuracy))
    if (CurAccuracy > Accuracy) {
        Accuracy    = CurAccuracy
        ConfMatrix = CurConfMatrix
        BsetProp    = prop
    }
}

sprintf("We get best accuracy = %.2f with Proportion = %.2f", Accuracy, BsetProp)
```

```
================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 161616
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.316
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0172
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

-----------------------------------------------------------------------------------------------------------------
[1] "Proportion = 0.50, Accuracy = 36.05"
```

```
================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 193940
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0171
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

-----------------------------------------------------------------------------------------------------------------
[1] "Proportion = 0.60, Accuracy = 36.01"
```

```
================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 226265
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0172
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

-----------------------------------------------------------------------------------------------------------------
[1] "Proportion = 0.70, Accuracy = 35.93"
```

```
================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 290910
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0171
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

---------------------------------------------------------------------------------------------------------------------
[1] "Proportion = 0.90, Accuracy = 36.28"
[1] "We get best accuracy = 36.28 with Proportion = 0.90"
```

In this case we get best accuracy when proportion is 0.9.

Then try to get best low frequency by accuracy.

```{r}
FreqSet  = c(1, 5, 20, 40, 60, 80)
BestFreq = 10
for(Freq in FreqSet){
    CurConfMatrix = GetConfMatrix (BsetProp, Freq)
    CurAccuracy   = CurConfMatrix$overall['Accuracy']*100
    print (sprintf("LowFreq = %d, Accuracy = %.2f", Freq, CurAccuracy))
    if (CurAccuracy > Accuracy) {
        Accuracy   = CurAccuracy
        ConfMatrix = CurConfMatrix
        BestFreq   = Freq
    }
}
sprintf("We get best accuracy = %.2f with lowFreq = %.2f and spliting proportion = %.2f",
         Accuracy, BestFreq, BsetProp)
```

```
================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 290910
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0171
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

---------------------------------------------------------------------------------------------------------------------
[1] "LowFreq = 1, Accuracy = 35.88"
```

```
================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 290910
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0171
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

---------------------------------------------------------------------------------------------------------------------
[1] "LowFreq = 5, Accuracy = 35.67"
```

```
================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 290910
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0171
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

-----------------------------------------------------------------------------------------------------------------
[1] "LowFreq = 20, Accuracy = 36.15"
================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 290910
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0171
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

-----------------------------------------------------------------------------------------------------------------
[1] "LowFreq = 40, Accuracy = 36.01"
================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 290910
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0171
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

-----------------------------------------------------------------------------------------------------------------
[1] "LowFreq = 60, Accuracy = 36.01"

================================================== Naive Bayes ==================================================

- Call: naive_bayes.default(x = TrainDtm, y = TrainLabels, laplace = 1)
- Laplace: 1
- Classes: 9
- Samples: 290910
- Features: 261
- Conditional distributions:
    - Bernoulli: 261
- Prior probabilities:
    - Bank account or service: 0.2667
    - Checking or savings account: 0.3159
    - Consumer Loan: 0.0978
    - Money transfers: 0.0166
    - Other financial service: 0.0033
    - Payday loan: 0.0171
    - Student loan: 0.1959
    - Vehicle loan or lease: 0.0867
    - Virtual currency: 1e-04

-----------------------------------------------------------------------------------------------------------------
[1] "LowFreq = 80, Accuracy = 35.88"
[1] "We get best accuracy = 36.28 with lowFreq = 10.00 and spliting proportion = 0.90"
```

Finally, we get the best accuracy 36.28 with low frequency = 10 and splitting proportion = 0.90

Show the confusion matrix:

```{r}
print (ConfMatrix)
```

```
Confusion Matrix and Statistics

                             Reference
Prediction                    Bank account or service Checking or savings account Consumer Loan
  Bank account or service                        7306                        7011          2295
  Checking or savings account                     613                        1863            88
  Consumer Loan                                    18                          14            79
  Money transfers                                   1                           1             0
  Other financial service                           0                           0             0
  Payday loan                                       0                           0             1
  Student loan                                    130                         174           187
  Vehicle loan or lease                           237                         498           374
  Virtual currency                                315                         651           136
                             Reference
Prediction                    Money transfers Other financial service Payday loan Student loan
  Bank account or service                 395                      85         409         3582
  Checking or savings account              68                       4          24          176
  Consumer Loan                             1                       1          15           45
  Money transfers                           4                       0           0            0
  Other financial service                   0                       0           0            0
  Payday loan                               0                       0           2            1
  Student loan                             13                       6          53         1681
  Vehicle loan or lease                    18                       7          35          346
  Virtual currency                         36                       2          16          500
                             Reference
Prediction                    Vehicle loan or lease Virtual currency
  Bank account or service                      1381                1
  Checking or savings account                   142                0
  Consumer Loan                                 100                0
  Money transfers                                 0                0
  Other financial service                         0                0
  Payday loan                                     0                0
  Student loan                                  194                0
  Vehicle loan or lease                         789                0
  Virtual currency                              195                0

Overall Statistics

               Accuracy : 0.3628
                 95% CI : (0.3575, 0.368)
    No Information Rate : 0.316
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.1656

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bank account or service Class: Checking or savings account Class: Consumer Loan
Sensitivity                                  0.8476                            0.18243             0.025000
Specificity                                  0.3604                            0.94956             0.993347
Pos Pred Value                               0.3252                            0.62559             0.289377
Neg Pred Value                               0.8667                            0.71545             0.903857
Prevalence                                   0.2667                            0.31598             0.097775
Detection Rate                               0.2261                            0.05764             0.002444
Detection Prevalence                         0.6951                            0.09214             0.008447
Balanced Accuracy                            0.6040                            0.56600             0.509173
                     Class: Money transfers Class: Other financial service Class: Payday loan
Sensitivity                       0.0074766                       0.000000          3.610e-03
Specificity                       0.9999371                       1.000000          9.999e-01
Pos Pred Value                    0.6666667                            NaN          5.000e-01
Neg Pred Value                    0.9835670                       0.996751          9.829e-01
Prevalence                        0.0165537                       0.003249          1.714e-02
Detection Rate                    0.0001238                       0.000000          6.188e-05
Detection Prevalence              0.0001856                       0.000000          1.238e-04
Balanced Accuracy                 0.5037069                       0.500000          5.018e-01
                     Class: Student loan Class: Vehicle loan or lease Class: Virtual currency
Sensitivity                      0.26552                      0.28169               0.000e+00
Specificity                      0.97087                      0.94868               9.427e-01
Pos Pred Value                   0.68950                      0.34245               0.000e+00
Neg Pred Value                   0.84438                      0.93297               1.000e+00
Prevalence                       0.19589                      0.08667               3.094e-05
Detection Rate                   0.05201                      0.02441               0.000e+00
Detection Prevalence             0.07544                      0.07129               5.727e-02
Balanced Accuracy                0.61820                      0.61518               4.714e-01
```