Assignment 5

WenLi

10/21/2020

1. This question involves the use of multiple linear regression on the Auto data set from the course webpage (https://scads.eecs.wsu.edu/index.php/datasets/). Ensure that you remove missing values from the dataframe, and that values are represented in the appropriate types.

```
library(kableExtra)
Original = read.csv("Auto.csv", na.strings="?")
Auto     = na.omit(Original)
kable(head(Auto), format = "latex", booktabs = T, caption="Dataset of Auto.csv")%>%
    kable_styling(latex_options = c("hold_position"))
```

Table 1: Dataset of Auto.csv

mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	name
18	8	307	130	3504	12.0	70	1	chevrolet chevelle malibu
15	8	350	165	3693	11.5	70	1	buick skylark 320
18	8	318	150	3436	11.0	70	1	plymouth satellite
16	8	304	150	3433	12.0	70	1	amc rebel sst
17	8	302	140	3449	10.5	70	1	ford torino
15	8	429	198	4341	10.0	70	1	ford galaxie 500

a. Perform a multiple linear regression with mpg as the response and all other variables except name as the predictors. Show a printout of the result (including coefficient, error and t values for each predictor). Comment on the output:

```
LinearReg = lm(mpg ~ cylinders + displacement + horsepower + weight + year + origin, data = Auto)
summary (LinearReg)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     year + origin, data = Auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7604 -2.1791 -0.1535  1.8524 13.1209
```

```
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.556e+01  4.175e+00  -3.728 0.000222 ***
## cylinders    -5.067e-01  3.227e-01  -1.570 0.117236
## displacement  1.927e-02  7.472e-03   2.579 0.010287 *
## horsepower   -2.389e-02  1.084e-02  -2.205 0.028031 *
## weight       -6.218e-03  5.714e-04 -10.883  < 2e-16 ***
## year          7.475e-01  5.079e-02  14.717  < 2e-16 ***
## origin        1.428e+00  2.780e-01   5.138 4.43e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.326 on 385 degrees of freedom
## Multiple R-squared:  0.8212, Adjusted R-squared:  0.8184
## F-statistic: 294.6 on 6 and 385 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?

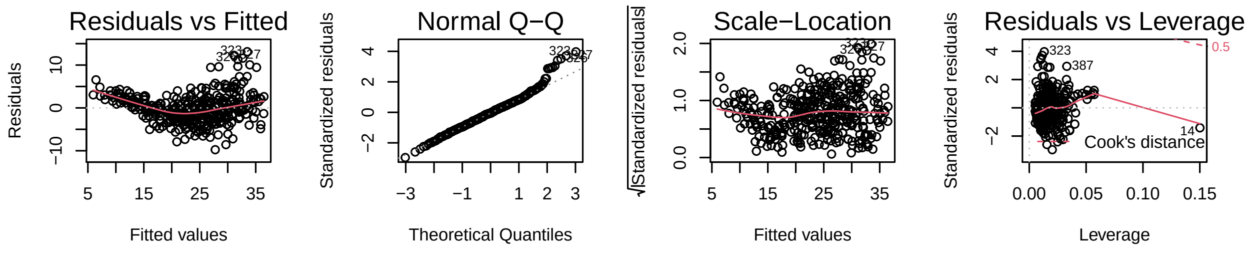
From the results, we can know that the predictors {displacement, horsepower, weight, year, origin} have a statistically significant relationship to the response mpg. we can determine this though the p-Value which indicate how significant the relationship is (the number of stars).

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

The coefficient of displacement is positive. It suggest that how much the value of "mpg" will increase when the number of displacement 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?

```r
par(mfrow=c(1,4))
plot(LinearReg)
```



(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: Point 14 has a high leverage which means that it has a high influence ie 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?

```
InterLinearReg = lm(mpg ~ . - name + weight:acceleration + origin:year + acceleration:horsepower, data=A
summary(InterLinearReg)
```

```
##
## Call:
## lm(formula = mpg ~ . - name + weight:acceleration + origin:year +
##     acceleration:horsepower, data = Auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6898 -1.8389 -0.1018  1.6369 11.6747
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -2.066e+01  9.548e+00  -2.163  0.03113 *
## cylinders              -3.635e-02  3.130e-01  -0.116  0.90759
## displacement           -7.709e-03  8.083e-03  -0.954  0.34084
## horsepower              4.534e-02  3.618e-02   1.253  0.21082
## weight                  1.195e-03  1.919e-03   0.623  0.53394
## acceleration            1.483e+00  2.423e-01   6.118 2.35e-09 ***
## year                    5.105e-01  1.065e-01   4.792 2.37e-06 ***
## origin                 -1.121e+01  4.441e+00  -2.524  0.01200 *
## weight:acceleration    -3.484e-04  1.199e-04  -2.905  0.00389 **
## year:origin             1.568e-01  5.707e-02   2.748  0.00628 **
## horsepower:acceleration -6.281e-03  2.533e-03  -2.480  0.01356 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.089 on 381 degrees of freedom
## Multiple R-squared:  0.8473, Adjusted R-squared:  0.8433
## F-statistic: 211.5 on 10 and 381 DF,  p-value: < 2.2e-16
```

Interactions such as weight:acceleration, year:origin, horsepower:acceleration appear to be statistically significant.

2. This problem involves the Boston data set, which we saw in class. We will now try to predict per capita crime rate using the other variables in this data set. In other words, per capita crime rate is the response, and the other variables are the predictors.

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

Table 2: Dataset of Boston

crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0.00632	18	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
0.02985	0	2.18	0	0.458	6.430	58.7	6.0622	3	222	18.7	394.12	5.21	28.7

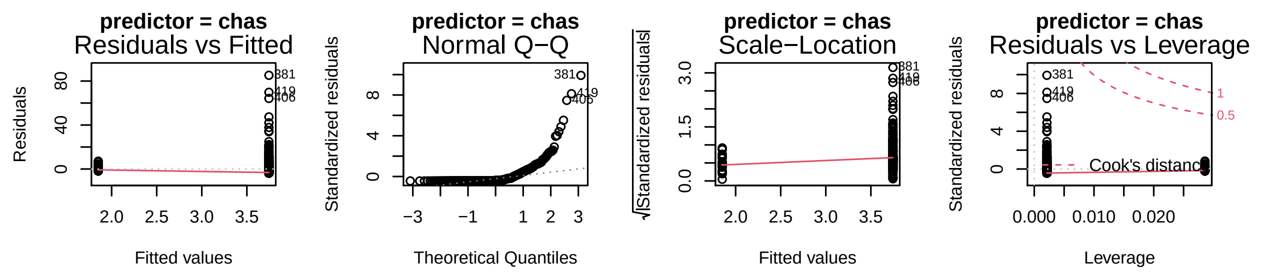
a. For each predictor, fit a simple linear regression model to predict the response. Include the code, but not the output for all models in your solution.

```
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)
```

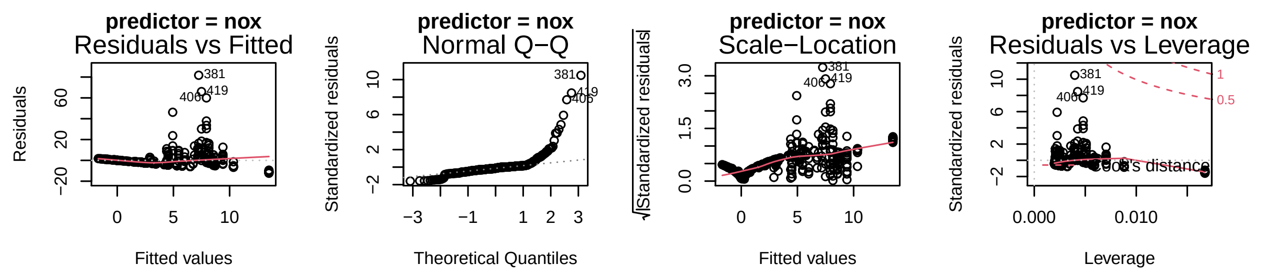
b. In which of the models is there a statistically significant association between the predictor and the response? Considering the meaning of each variable, discuss the relationship between crim and nox, chas, medv and dis in particular. How do these relationships differ?

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

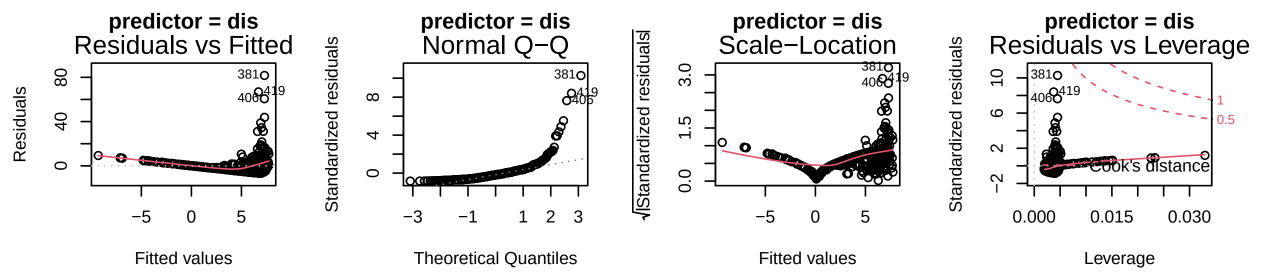
```
par(mfrow = c(1,4))
plot(LinearReg_chas, main = "predictor = chas")
```



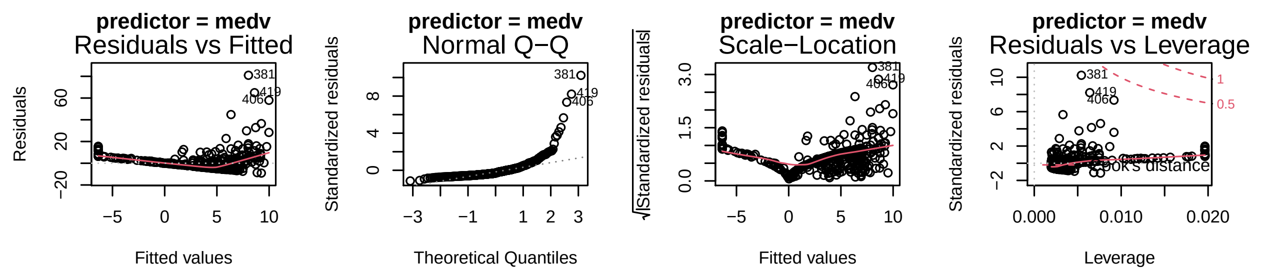
```r
plot(LinearReg_nox,  main = "predictor = nox")
```



```r
plot(LinearReg_dis,  main = "predictor = dis")
```



```r
plot(LinearReg_medv, main = "predictor = medv")
```



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

```
## 0.003123869, 0.1772172, 0.1441494, 0.1507805
```

From the figures, we can see that there is linear relationship between the nox and crim. And amond all the four predictors, nox gets the highest R Squared value. There is no association between chas and crim. For the other two predictors dis, 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.

```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
```

(1) There are four predictors including {zn, dis, rad, black, mdev} have significant association with crim.
(2) R-squared value is higher for multiple regression when being compared to the simple regressions.
(3) 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 (b)? Create a plot displaying the univariate regression coefficients from (a) on the x-axis, and the multiple regression coefficients from (b) on the y-axis. That is, each predictor is displayed as a single point in the plot. Its coefficient in a simple linear regression model is shown on the x-axis, and its coefficient estimate in the multiple linear regression model is shown on the y-axis. What does this plot tell you about the various predictors?

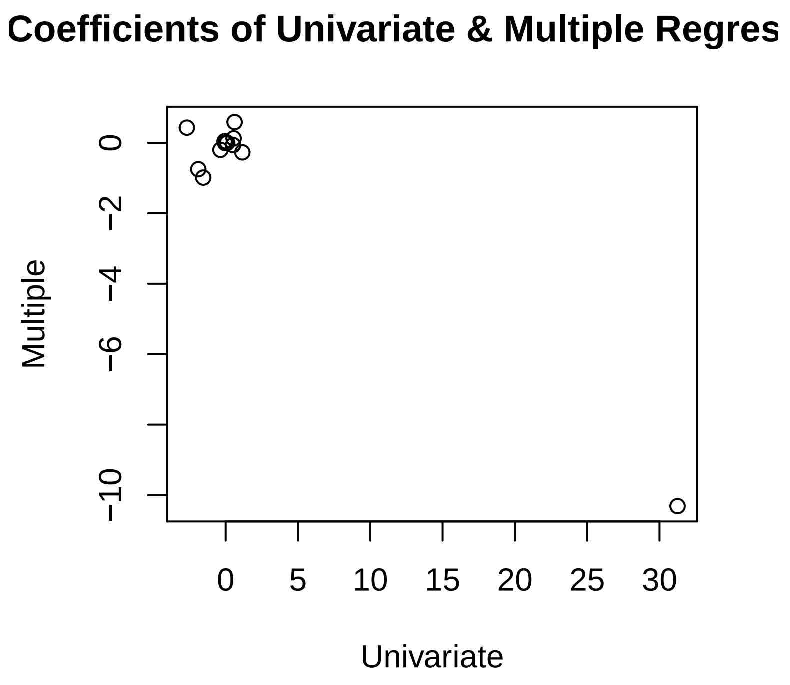
```r
CoefSimple = vector("numeric", 0)
CoefSimple = c(LinearReg_zn$coefficient[2],    LinearReg_indus$coefficient[2],
               LinearReg_chas$coefficient[2],  LinearReg_nox$coefficient[2],
               LinearReg_rm$coefficient[2],    LinearReg_age$coefficient[2],
               LinearReg_dis$coefficient[2],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

```
##          zn       indus        chas         nox          rm         age
## -0.07393498  0.50977633 -1.89277655 31.24853120 -2.68405122  0.10778623
##         dis         rad         tax     ptratio       black       lstat
## -1.55090168  0.61791093  0.02974225  1.15198279 -0.03627964  0.54880478
##        medv
## -0.36315992
```

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

```
##            zn         indus          chas           nox            rm
##   0.044855215  -0.063854824  -0.749133611 -10.313534912   0.430130506
##           age           dis           rad           tax       ptratio
##   0.001451643  -0.987175726   0.588208591  -0.003780016  -0.271080558
##         black         lstat          medv
##  -0.007537505   0.126211376  -0.198886821
```

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



(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?

```
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)
```

Looking at the p-calue, we can know: (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 quandratic 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, β0 = −7, β1 = 0.1, β2 = 1, β3 = -.04.

a. (5%) Estimate the probability that a student who studies for 32 h, has a PSQI score of 12 and has an undergrad GPA of 3.0 gets an A in the class. Show your work.

$\hat{y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 = -7 + 0.1X_1 + X_2 - 0.04X_3$

And $X_1 = 32,\ X_2 = 3.0, X_3 = 12$

=>$\hat{y} = -7 + 3.2 + 3.0 - 0.48 = -1.28$

=>$P(X) = \frac{e^y}{1+e^y} = \frac{e^{-1.28}}{1+e^{-1.28}}$ =21.76%

b. (5%) How many hours would the student in part (a) need to study to have a 50 % chance of getting an A in the class? Show your work.

Assume the student needs to study H hours.

Then we can get: $\hat{y} = -7 + 0.1H + 3.0 - 0.48 = 0.1H - 4.48$

$So,\ P(X) = \frac{e^{(0.1H-4.48)}}{1+e^{(0.1H-4.48)}} = 50\%$

$=> e^{0.1H-4.48} = 1$

$=> H = 44.8$

c. (5%) How many hours would a student with a 3.0 GPA and a PSQI score of 3 need to study to have a 50 % chance of getting an A in the class? Show your work.

Assume the student needs to study H hours.

Then we can get: $\hat{y} = -7 + 0.1H + 3.0 - 0.12 = 0.1H - 4.12$

$So,\ P(X) = \frac{e^{(0.1H-4.12)}}{1+e^{(0.1H-4.12)}} = 50\%$

$=> e^{0.1H-4.12} = 1$

$=> H = 41.2$

4 For this question, you will a naïve Bayes model to classify newspaper articles by their section. You will be provided a set of news articles (http://scads.eecs.wsu.edu/index.php/datasets) collected from the Guardian (a British newspaper). The articles are cleared of major confounding factors, such as HTML tags, but it is up to you to check the articles for other problems and to prepare them for classification.

```
library(stringi)
library(textreadr)
library(tm)
Articles = read.csv("GuardianArticles.csv", header = TRUE)
Articles$section = as.factor(Articles$section)
str (Articles)
```

```
## 'data.frame':    11458 obs. of  6 variables:
##  $ X      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ id     : chr  "artanddesign/2018/oct/10/norman-fosters-bloomberg-office-in-london-wins-stirling-pr
##  $ title  : chr  "Norman Foster's Bloomberg office in London wins Stirling prize" "Ray Smith obituary
##  $ body   : chr  "one of the most environmentally friendly office buildings ever conceived has been r
##  $ wc     : int  968 445 808 770 1498 1145 1512 476 493 799 ...
##  $ section: Factor w/ 6 levels "artanddesign",..: 1 1 1 1 1 1 1 1 1 1 ...
```

a. Tokenization

```
TokenDTM = Corpus(VectorSource(Articles$body)) %>%
           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: 11458, terms: 4027)>>
## Non-/sparse entries: 2618665/43522701
## Sparsity           : 94%
## Maximal term length: 29
## Weighting          : term frequency (tf)
## Sample             :
##       Terms
## Docs   like new one peopl said say time will work year
##   2872   18  11  11    14    2  26   11   23   13   29
##   3842   20   9  20     5    2   6   12    5    1   31
##   5464   15   4  21    18   12  17   18    3   11    2
##   5606   24   5  24    14    8  10   12    3   17   13
##   6971    8   9  18    10    1   1   12   13    8    9
##   7008   20  29  10    15   20  24   25    5   42   14
##   8173   17  15  22    24   15   2   16   12   29   20
##   869    18  17  18    11    0  14    5   17   11    9
##   9715   27  23  21    18     3   4   13     2   32   14
##   9770    5   5  20     2     0   1   14    26     1   49
```

```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)]
```

```
##      art beat best can chanc close couldnt everi fact favourit field good hard
## 9891   1    1    1   2     1     1       1     1    1        2     2    1    1
##      hope just last light like make media month one open reason relat rise said
## 9891    3    2    2     1    1    1     1     1   1    1      1     1    1    4
##      say two view water well went will win winner won year artist career earli
## 9891   1   2    1     1    4    1    2   2      2   4    1      1      1     1
##      form highlight long now ran remain start surviv though apart certain charg
## 9891    1         1    1   1   1      1     3      1      1     1       1     1
##      fit joint left much offer part see social still take yet bit come get give
## 9891   2     2    2    2     1    1   1      1     1    2   1   1    1   3    3
##      got happen hors mount need point quit sinc think although becam day hes
## 9891   1      1    4     1    2     1    1    1     1        1     1   4   2
##      job lucki meanwhil old reliev result week add albeit behind chang definit
## 9891   1     1        1   5      1      1    5   1      1      3     1       1
##      even king slight sustain angel began care end follow hill inch leg morn
## 9891    1    1      1       1     2     1    1   1      1    1    1   2    1
##      necessari relief run simpli that suffer agre board charli decid derbi
## 9891         1      1   3      1    3      1    1     1      1     1     4
##      maintain royal sent aliv away must sign team attempt concern juli june
## 9891        1     3    1    1    1    1    1    1       1       2    1    1
##      realli stay tri appear cut explos passag return sure deep next weve friday
## 9891      1    1   1      1   3      1      1      2    1    1    2    1      1
##      money profession wont entri fee interest oper saturday tuesday free condit
## 9891     1          1    1     2   1        1    1        3       1    2      1
##      obvious proper extent final pleas race stone clean finish roll spot
## 9891       1      1      1     2     2    6     1     1      1    1    1
##      trainer classic trip improv welcom disappoint initi monitor zone owner
## 9891       2       1    1      1      1          1     1       1    1     1
##      desert diamond outcom argument path foot cup gear irish warrior eas brook
## 9891      1       2      1        1    1    1   1    1     4       1   1     1
##      adam harri prevent glori stall rough stake stump injuri runner incid billi
## 9891    1     2       1     1     3     1     2     1      1      2     1     1
##      nap ascot
## 9891   1     5
```

b. Classification

(1). Reduce feature sets and remove correlated features

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

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

```r
library(e1071)

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(Articles$section, p = Proportion, list = FALSE, times = 1)

    # Proportion Of Article categories in Train Data
    TrainArticles = Articles[Index,]
    TrainLabels   = TrainArticles$section
    prop.table(table(TrainArticles$section))
    #Proportion of Article categories in Test Data
    TestArticles = Articles[-Index,]
    TestLabels   = TestArticles$section
    prop.table(table(TestArticles$section))

    #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 = naiveBayes(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)
}

ConfMatrix = GetConfMatrix (0.8, 10)
Accuracy   = ConfMatrix$overall['Accuracy']*100
sprintf("Proportion = 0.8, Accuracy = %.2f", Accuracy)
```

```
## [1] "Proportion = 0.8, Accuracy = 81.52"
```

(2). Get best spliting 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
    }
}
```

```
## [1] "Proportion = 0.60, Accuracy = 81.82"
## [1] "Proportion = 0.70, Accuracy = 82.53"
## [1] "Proportion = 0.90, Accuracy = 80.93"
```

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

```
## [1] "We get best accuracy = 82.53 with Proportion = 0.70"
```

(3). Get best low-freq by accuracy

```
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
    }
}
```

```
## [1] "LowFreq = 1, Accuracy = 83.35"
## [1] "LowFreq = 5, Accuracy = 82.65"
## [1] "LowFreq = 20, Accuracy = 82.33"
## [1] "LowFreq = 40, Accuracy = 82.42"
## [1] "LowFreq = 60, Accuracy = 82.10"
## [1] "LowFreq = 80, Accuracy = 81.19"
```

```
sprintf("We get best accuracy = %.2f with lowFreq = %.2f and spliting proportion = %.2f",
         Accuracy, BestFreq, BsetProp)
```

```
## [1] "We get best accuracy = 83.35 with lowFreq = 1.00 and spliting proportion = 0.70"
```

(3). Show the confusion matrix

```
print (ConfMatrix)
```

```
## Confusion Matrix and Statistics
##
##                 Reference
## Prediction       artanddesign business culture sport technology world
##   artanddesign            512        1      86     2          6    14
##   business                 11      503      21    14         70    45
##   culture                  48        3     409    14         26    25
##   sport                  3        2       0   482          3     2
##   technology            12       26      25     3        471    19
##   world                 13       19      24    16         19   486
##
## Overall Statistics
##
##                Accuracy : 0.8335
##                  95% CI : (0.8206, 0.8458)
##     No Information Rate : 0.1744
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8001
##
##  Mcnemar's Test P-Value : 1.62e-14
##
## Statistics by Class:
##
##                      Class: artanddesign Class: business Class: culture
## Sensitivity                       0.8548          0.9079         0.7239
## Specificity                       0.9616          0.9441         0.9596
## Pos Pred Value                    0.8245          0.7575         0.7790
## Neg Pred Value                    0.9691          0.9816         0.9464
## Prevalence                        0.1744          0.1613         0.1645
## Detection Rate                    0.1491          0.1464         0.1191
## Detection Prevalence              0.1808          0.1933         0.1528
## Balanced Accuracy                 0.9082          0.9260         0.8417
##                      Class: sport Class: technology Class: world
## Sensitivity                0.9077            0.7916       0.8223
## Specificity                0.9966            0.9701       0.9680
## Pos Pred Value             0.9797            0.8471       0.8423
## Neg Pred Value             0.9834            0.9569       0.9633
## Prevalence                 0.1546            0.1732       0.1721
## Detection Rate             0.1403            0.1371       0.1415
## Detection Prevalence       0.1432            0.1619       0.1680
## Balanced Accuracy          0.9521            0.8808       0.8952
```