STT481Hw2

Sam Isken

October 5, 2019

FS19 STT481: Homework 2 (Due: Friday, October 18th, beginning of the class.)

1. (10 pts) Finish the swirl course Regression Models. Finish Sections 1-13. You can install and go to the course by using the following command lines.

#library(swirl)  
#install\_course("Regression\_Models")  
#swirl()

I have included the pictures of my Swirl completion in the attached folder (zip file)

1. (10 pts) Download the csv file sat.csv and read the description of the dataset below.

sat <- read.csv("sat.csv")  
head(sat)

## state sat takers income years public expend rank  
## 1 Iowa 1088 3 326 16.79 87.8 25.60 89.7  
## 2 SouthDakota 1075 2 264 16.07 86.2 19.95 90.6  
## 3 NorthDakota 1068 3 317 16.57 88.3 20.62 89.8  
## 4 Kansas 1045 5 338 16.30 83.9 27.14 86.3  
## 5 Nebraska 1045 5 293 17.25 83.6 21.05 88.5  
## 6 Montana 1033 8 263 15.91 93.7 29.48 86.4

summary(sat)

## state sat takers income   
## Alabama : 1 Min. : 790.0 Min. : 2.00 Min. :208.0   
## Alaska : 1 1st Qu.: 889.2 1st Qu.: 6.25 1st Qu.:261.5   
## Arizona : 1 Median : 966.0 Median :16.00 Median :295.0   
## Arkansas : 1 Mean : 947.9 Mean :26.22 Mean :294.0   
## California: 1 3rd Qu.: 998.5 3rd Qu.:47.75 3rd Qu.:325.0   
## Colorado : 1 Max. :1088.0 Max. :69.00 Max. :401.0   
## (Other) :44   
## years public expend rank   
## Min. :14.39 Min. :44.80 Min. :13.84 Min. :69.80   
## 1st Qu.:15.91 1st Qu.:76.92 1st Qu.:19.59 1st Qu.:74.03   
## Median :16.36 Median :80.80 Median :21.61 Median :80.85   
## Mean :16.21 Mean :81.20 Mean :22.97 Mean :79.99   
## 3rd Qu.:16.76 3rd Qu.:88.25 3rd Qu.:26.39 3rd Qu.:85.83   
## Max. :17.41 Max. :97.00 Max. :50.10 Max. :90.60   
##

In 1982, average SAT scores were published with breakdowns of state-by-state performance in the United States. The average SAT scores varied considerably by state, with mean scores falling between 790 (South Carolina) to 1088 (Iowa). Two researchers examined compositional and demographic variables to examine to what extent these characteristics were tied to SAT scores. The variables in the data set were:

state: state name

sat: mean SAT score (verbal and quantitative combined)

takers: percentage of total eligible students (high school seniors) in the state who took the exam

income: median income of families of test takers, in hundreds of dollars.

years: average number of years that test takers had in social sciences, natural sciences, and humanities (combined)

public: percentage of test takers who attended public schools

expend: state expenditure on secondary schools, in hundreds of dollars per student

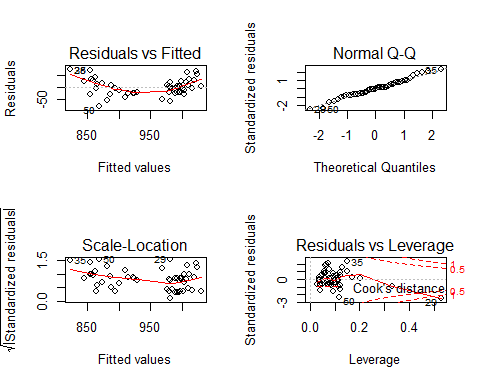
rank: median percentile of ranking of test takers within their secondary school classes. Possible values range from 0-99, with 99th percentile students being the highest achieving.

Fit a model with the sat as the response and expend, income, public and takers as predictors. Perform regression diagnostics on this model to answer the following questions. Display any plots that are relevent

SAT\_Model1 <- lm(sat~expend+income+public+takers ,data=sat)  
summary(SAT\_Model1)

##   
## Call:  
## lm(formula = sat ~ expend + income + public + takers, data = sat)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -79.139 -23.673 -1.953 20.554 72.827   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1083.1366 84.9776 12.746 < 2e-16 \*\*\*  
## expend 3.1332 1.0589 2.959 0.00491 \*\*   
## income -0.2533 0.1929 -1.313 0.19582   
## public -0.5656 0.6012 -0.941 0.35177   
## takers -3.3093 0.3692 -8.965 1.42e-11 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 34.66 on 45 degrees of freedom  
## Multiple R-squared: 0.7803, Adjusted R-squared: 0.7607   
## F-statistic: 39.94 on 4 and 45 DF, p-value: 2.896e-14

par(mfrow = c(2, 2))  
plot(SAT\_Model1)

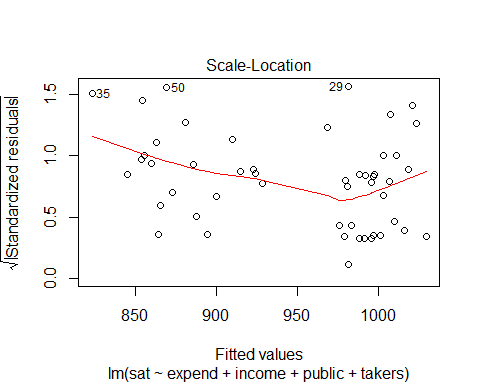


1. Check the constant variance assumption for the errors.

ncvTest(SAT\_Model1)

## Non-constant Variance Score Test   
## Variance formula: ~ fitted.values   
## Chisquare = 3.309361, Df = 1, p = 0.068886

plot(SAT\_Model1,3)



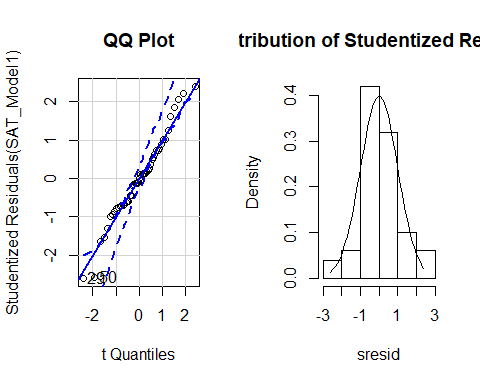
From this we see our errors are relatively homoskedastic around the line with 3 outliers (25,50,29). This could be improved by transforming out outcome variable but we can say that the assumption for constant error variance us fuffilled enough to proceed with our model.

1. Check the normality assumption.

#Display both plots at the same time   
par(mfrow = c(1, 2))  
#Create a QQ plot to see the root - standardized residuals   
qqPlot(SAT\_Model1, main="QQ Plot")

## [1] 29 50

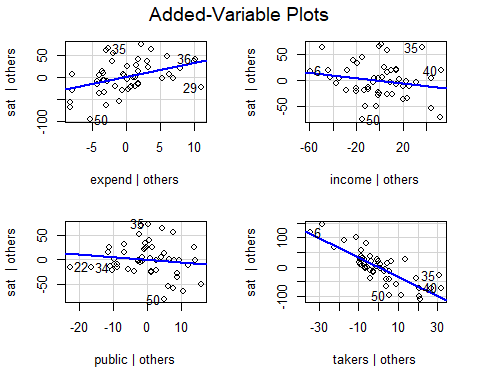
#Compute studentized residuals: Like standardized residuals, these are normalized to unit variance, but the Studentized version is fitted ignoring the current data point. (They are sometimes called jackknifed residuals).  
sresid <- studres(SAT\_Model1)  
  
#Plot studentized residuals and compare how they fall in comparison to a normal distribution  
hist(sresid, freq=FALSE,  
 main="Distribution of Studentized Residuals")  
xfit<-seq(min(sresid),max(sresid),length=40)  
yfit<-dnorm(xfit)  
lines(xfit, yfit)



As we can see from these plots our residuals are distributed roughly normal. Since our Normal Q-Q plot is relatively on the line we can assume normality of the residuals (shown by both regular and studentized residuals).

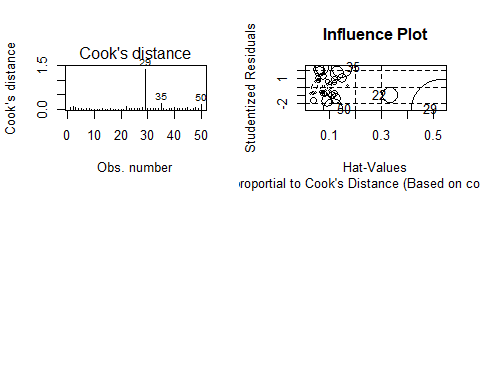
1. Check for large leverage points.

#Display 2 plots   
par(mfrow = c(2, 2))  
  
#added variable plots: These functions construct added-variable, also called partial-regression, plots for linear and generalized linear models  
avPlots(SAT\_Model1)



# Cook's D plot  
#Set the cutoff value for Cook's Distance as : 4/(n-k-1)  
cutoff <- 4/((nrow(sat)-length(SAT\_Model1$coefficients)-2))  
plot(SAT\_Model1, which=4, cook.levels=cutoff)  
# Influence Plot  
influencePlot(SAT\_Model1, id.method="identify", main="Influence Plot", sub="Circle size is proportial to Cook's Distance (Based on computed cutoff value)" )

## StudRes Hat CookD  
## 22 -0.9981088 0.3329581 0.09946233  
## 29 -2.6092480 0.5294350 1.35685279  
## 35 2.3979347 0.1511534 0.18523035  
## 50 -2.5766150 0.1163300 0.15533113



As we can see points 22, 29, 35 and 50 are considered large leverage points. This makes sense as all of these points (except 35 & 22) were considered outliers so it is logical that are they are having undue influence on the model

1. Check for the outliers.

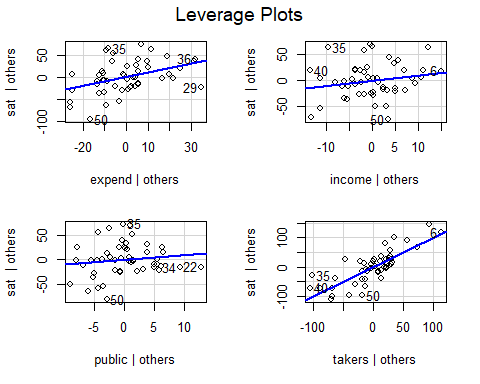
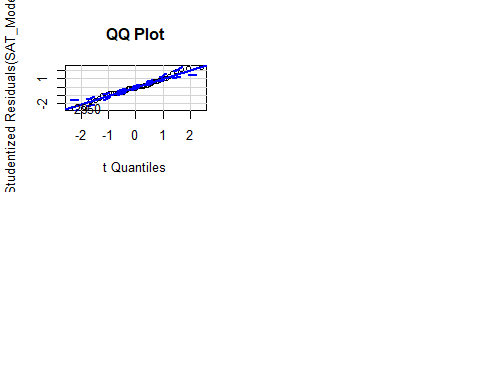
#Find outliers, plot standardized residuals, find leverage points   
outlierTest(SAT\_Model1)

## No Studentized residuals with Bonferroni p < 0.05  
## Largest |rstudent|:  
## rstudent unadjusted p-value Bonferroni p  
## 29 -2.609248 0.012353 0.61764

par(mfrow = c(2, 2))  
qqPlot(SAT\_Model1, main="QQ Plot")

## [1] 29 50

leveragePlots(SAT\_Model1)



As we can see points 29 and 50 are outliers.

1. Check the structure of the relationship between the predictors and the response.

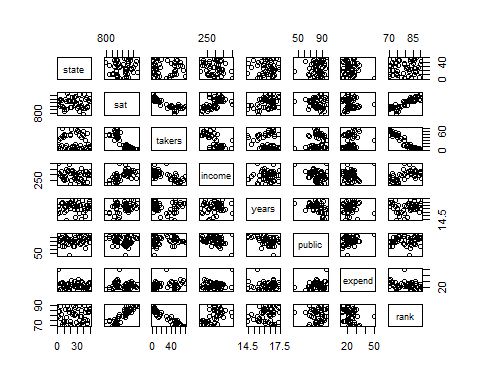
summary(SAT\_Model1)

##   
## Call:  
## lm(formula = sat ~ expend + income + public + takers, data = sat)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -79.139 -23.673 -1.953 20.554 72.827   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1083.1366 84.9776 12.746 < 2e-16 \*\*\*  
## expend 3.1332 1.0589 2.959 0.00491 \*\*   
## income -0.2533 0.1929 -1.313 0.19582   
## public -0.5656 0.6012 -0.941 0.35177   
## takers -3.3093 0.3692 -8.965 1.42e-11 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 34.66 on 45 degrees of freedom  
## Multiple R-squared: 0.7803, Adjusted R-squared: 0.7607   
## F-statistic: 39.94 on 4 and 45 DF, p-value: 2.896e-14

The only predictors that are significant are ‘expend’ and ‘takers’.Thus, the structure is that this model does not appear to be a great method of predicting SAT scores.

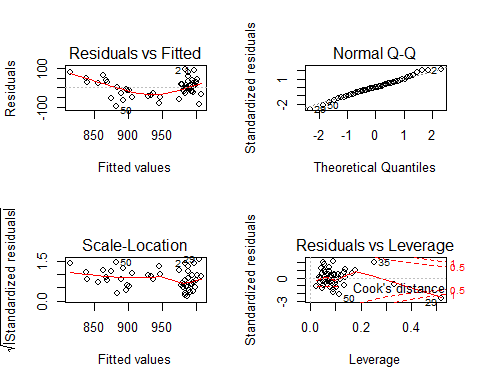
1. Use pairs function in R to see the relationship between sat and other predictors.

pairs(sat)



You can see takers appears to have a quadratic relationship with sat. Include this quadratic effect in your current model and perform the regression diagnostics. Check the structure of the relationship between the predictors and the response again.

takers2 <- sat$takers^2  
SAT\_Model2 <- lm(sat~expend+income+public+takers2 ,data=sat)  
par(mfrow = c(2, 2))  
plot(SAT\_Model2)



summary(SAT\_Model2)

##   
## Call:  
## lm(formula = sat ~ expend + income + public + takers2, data = sat)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -92.309 -28.063 -1.835 27.869 91.315   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 952.108989 110.259635 8.635 4.18e-11 \*\*\*  
## expend 1.584423 1.364969 1.161 0.252   
## income 0.096563 0.248311 0.389 0.699   
## public -0.294323 0.803357 -0.366 0.716   
## takers2 -0.038738 0.007419 -5.221 4.39e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 45.65 on 45 degrees of freedom  
## Multiple R-squared: 0.6188, Adjusted R-squared: 0.5849   
## F-statistic: 18.26 on 4 and 45 DF, p-value: 5.633e-09

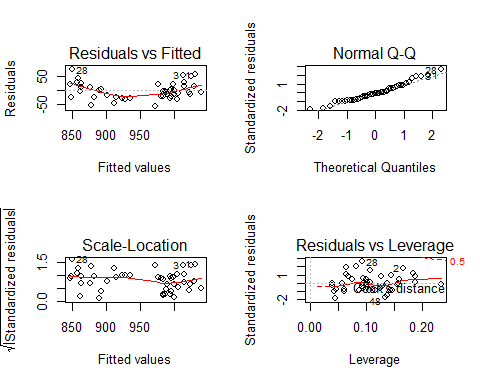
This made our QQ plot perform better (closer to a normal distribution) and caused expend to become insignificant. Thus, the only significant predictor in this model is takers^2.

1. Using the model in (a)-(e) (no quadratic effect), remove the large leverage points you found and perform the regression diagnostics. Check for large leverage points again.

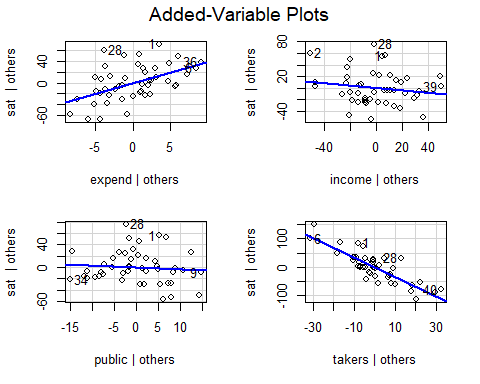
data\_rm <- c(22, 29, 35,50)  
SAT\_data2 <- sat[-data\_rm,]  
SAT\_Model3 <- lm(sat~expend+income+public+takers ,data=SAT\_data2)  
summary(SAT\_Model3)

##   
## Call:  
## lm(formula = sat ~ expend + income + public + takers, data = SAT\_data2)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -54.875 -21.679 -1.783 15.968 75.348   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1038.2393 89.1391 11.647 1.39e-14 \*\*\*  
## expend 3.8929 1.0576 3.681 0.000671 \*\*\*  
## income -0.2021 0.1940 -1.042 0.303627   
## public -0.3601 0.5985 -0.602 0.550731   
## takers -3.3838 0.3500 -9.669 3.91e-12 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 30.1 on 41 degrees of freedom  
## Multiple R-squared: 0.828, Adjusted R-squared: 0.8113   
## F-statistic: 49.36 on 4 and 41 DF, p-value: 3.809e-15

par(mfrow = c(2, 2))  
plot(SAT\_Model3)



#Display 2 plots   
par(mfrow = c(2, 2))  
  
#added variable plots: These functions construct added-variable, also called partial-regression, plots for linear and generalized linear models  
avPlots(SAT\_Model3)



# Cook's D plot  
#Set the cutoff value for Cook's Distance as : 4/(n-k-1)  
cutoff <- 4/((nrow(SAT\_data2)-length(SAT\_Model3$coefficients)-2))  
plot(SAT\_Model3, which=4, cook.levels=cutoff)  
# Influence Plot  
influencePlot(SAT\_Model3, id.method="identify", main="Influence Plot", sub="Circle size is proportial to Cook's Distance (Based on computed cutoff value)" )

## Warning in plot.window(...): "id.method" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "id.method" is not a graphical parameter

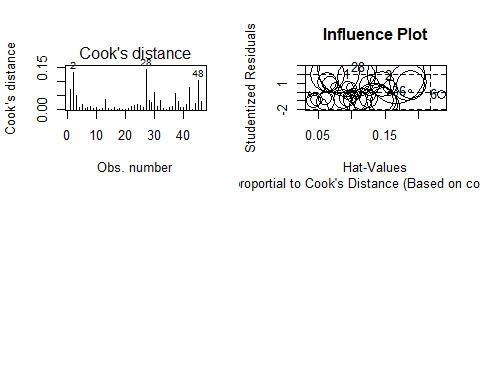
## Warning in axis(side = side, at = at, labels = labels, ...): "id.method" is  
## not a graphical parameter  
  
## Warning in axis(side = side, at = at, labels = labels, ...): "id.method" is  
## not a graphical parameter

## Warning in box(...): "id.method" is not a graphical parameter

## Warning in title(...): "id.method" is not a graphical parameter

## Warning in plot.xy(xy.coords(x, y), type = type, ...): "id.method" is not a  
## graphical parameter

## StudRes Hat CookD  
## 1 2.0812481 0.08149061 0.0710840495  
## 2 1.8840576 0.16615115 0.1331782607  
## 6 -0.2267100 0.23357118 0.0032068956  
## 28 2.8471633 0.09323455 0.1420751717  
## 36 0.1209622 0.18778705 0.0006932506



1. Comment on which model we should use. The model in (f) or the model in (a)-(e) with the removal of large leverage points that you did in (g)?

The removal of these outliers actually caused us to end up with more outliers. Thus, we should use our original model.

1. (10 pts) Question 2 in Section 4.7.
2. It was stated in the text that classifying an observation to the class for which (4.12) is largest is equivalent to classifying an observation to the class for which (4.13) is largest. Prove that this is the case. In other words, under the assumption that the observations in the kth class are drawn from a $(\_k,^2) $ distribution, the Bayes Classifier assigns an observation to the class for which the discriminant function is maximized.

In order to implement the Bayes Classifier we have to find the class (k) for which

is maximized. To do this we take the log of $ p\_{k}(x)$

We then set thus resulting in:

Since the last summation is independant of k we can manipulate our statement to be:

From here we can find the largest k for which:

1. (10 pts) Question 5 in Section 4.7.
2. We now examine the differences between LDA and QDA.
3. If the Bayes decision boundary is linear, do we expect LDA or QDA to perform better on the training set? On the test set?

We expect the QDA to perform better than the LDA on the training set because it is more flexible. We expect the LDA to perform better than the QDA on the on the test set because it is optimized using the Baye’s Decision boundary shown to maximize the classification in #2.

1. If the Bayes decision boundary is non-linear, do we expect LDA or QDA to perform better on the training set? On the test set?

Since LDA, given the name, assumes Multivariate normality, we expect QDA to out perform it on both sets.

1. In general, as the sample size n increases, do we expect the test prediction accuracy of QDA relative to LDA to improve, decline, or be unchanged? Why?

Because the flexibility of QDA > LDA we use LDA as n-> infinity. Since LDA assumes homogeneity of variance and QDA does not the increasing sample size is not a problem for QDA and our accuracy will increase for our test prediction.

1. True or False: Even if the Bayes decision boundary for a given problem is linear, we will probably achieve a superior test error rate using QDA rather than LDA because QDA is flexible enough to model a linear decision boundary. Justify your answer.

False. Because LDA assumes Multivariate normality and homogeneity of residuals it will fit a linear Baye’s decision boundary better. QDA could overfit this due to not recognizing the linearity.

1. (5 pts) Question 6 in Section 4.7.
2. Suppose we collect data for a group of students in a statistics class with variables =hours studied, =undergrad GPA, and Y = receive an A. We ï¬t a logistic regression and produce estimated coeï¬cient,
3. Estimate the probability that a student who studies for 40h and has an undergrad GPA of 3.5 gets an A in the class.

probx <- ((exp(1))^(-6+(.05\*40)+3.5))/(1+(exp(1))^(-6+(.05\*40)+3.5))  
probx

## [1] 0.3775407

Thus, the estimated probability a student scores an A given 40 hr study and a 3.5 GPA is .3775407 .

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

We could do this through algebra, but, We have R! Let us simply use our probx and plug in a few values.

probx2 <- ((exp(1))^(-6+(.05\*45)+3.5))/(1+(exp(1))^(-6+(.05\*45)+3.5))  
probx2 <- ((exp(1))^(-6+(.05\*45)+3.5))/(1+(exp(1))^(-6+(.05\*45)+3.5))  
probx2 <- ((exp(1))^(-6+(.05\*50)+3.5))/(1+(exp(1))^(-6+(.05\*50)+3.5))  
probx2

## [1] 0.5

Thus, a student would need to study 50 hours.

1. (5 pts) Question 9 in Section 4.7.
2. This problem has to do with odds.
3. On average, what fraction of people with an odds of 0.37 of defaulting on their credit card payment will in fact default?

To use odds we use the funcion:

odds:

after transformation this becomes:

Thus, 27% percent will default (on average, according to our model).

1. Suppose that an individual has a 16% chance of defaulting on her credit card payment. What are the odds that she will default?

The same logic applies:

Thus, 19% percent chance of default (on average, according to our model).

1. (10 pts) Download the csv file arthritis.csv. These data from Koch & Edwards (1988) from a double-blind clinical trial investigating a new treatment for rheumatoid arthritis. The data frame has 84 observations and 4 variables, which are:

Treatment: factor indicating treatment (Placebo, Treated). Sex: factor indicating sex (Female, Male). Age: age of patient. Improved: factor indicating treatment outcome (No, Yes).

1. Fit a logistic regression with the Improved as the response and Treatment, Sex and Age as predictors

arthritis <- read.csv("arthritis.csv")  
head(arthritis)

## Treatment Sex Age Improved  
## 1 Treated Male 27 Yes  
## 2 Treated Male 29 No  
## 3 Treated Male 30 No  
## 4 Treated Male 32 Yes  
## 5 Treated Male 46 Yes  
## 6 Treated Male 58 Yes

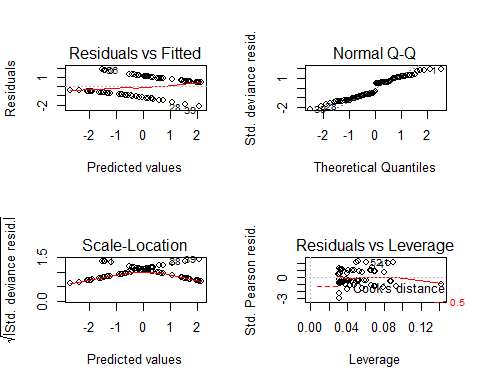
summary(arthritis)

## Treatment Sex Age Improved  
## Placebo:43 Female:59 Min. :23.00 No :42   
## Treated:41 Male :25 1st Qu.:46.00 Yes:42   
## Median :57.00   
## Mean :53.36   
## 3rd Qu.:63.00   
## Max. :74.00

arth\_Model1 <- glm(Improved~Treatment+Sex+Age,data=arthritis,family = binomial)  
par(mfrow = c(2, 2))  
summary(arth\_Model1)

##   
## Call:  
## glm(formula = Improved ~ Treatment + Sex + Age, family = binomial,   
## data = arthritis)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.10833 -0.91158 0.05362 0.91681 1.84659   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.01546 1.16777 -2.582 0.00982 \*\*  
## TreatmentTreated 1.75980 0.53650 3.280 0.00104 \*\*  
## SexMale -1.48783 0.59477 -2.502 0.01237 \*   
## Age 0.04875 0.02066 2.359 0.01832 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 116.449 on 83 degrees of freedom  
## Residual deviance: 92.063 on 80 degrees of freedom  
## AIC: 100.06  
##   
## Number of Fisher Scoring iterations: 4

plot(arth\_Model1)

 (b) Use log odds to interpret the coefficients

Given someone is treated (TreatmentTreated=1) we expect a 1.76 increase in log odds of improved health (with all other variables held constant)

Given someone is Male (SexMale=1) we expect a -1.49 decrease in log odds of improved health (with all other variables held constant)

Given a 1 unit increase in age (in yearS) we expect a .049 increase in log odds of improved health (with all other variables held constant)

1. Use odds to interpret the coefficients

Given someone is treated (TreatmentTreated=1) we expect a increase in odds of improved health (with all other variables held constant)

Given someone is Male (SexMale=1) we expect a decrease in odds of improved health (with all other variables held constant)

Given a 1 unit increase in age (in yearS) we expect a e^{.049} increase in odds of improved health (with all other variables held constant)

1. Construct 95% confidence intervals for the coefficients.

confint(arth\_Model1)

## Waiting for profiling to be done...

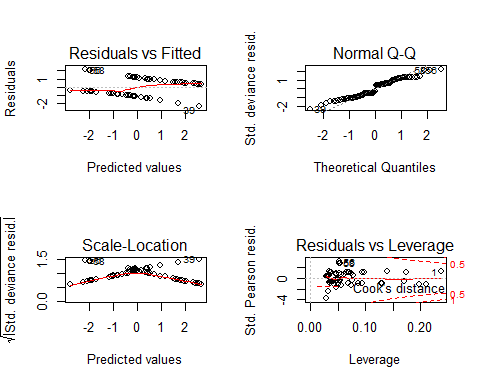
## 2.5 % 97.5 %  
## (Intercept) -5.477304188 -0.84351926  
## TreatmentTreated 0.750803551 2.87506538  
## SexMale -2.729719456 -0.37239953  
## Age 0.009951561 0.09194283

1. Add the interaction Sex:Age in your model. For this model, for a one year increase in age, how much does the log odds of some improvement (versus none) increase? Explain it separately with respect to Sex.

arth\_Model2 <- glm(Improved~Treatment+Sex+Age+Sex:Age,data=arthritis,family = binomial)  
par(mfrow = c(2, 2))  
summary(arth\_Model2)

##   
## Call:  
## glm(formula = Improved ~ Treatment + Sex + Age + Sex:Age, family = binomial,   
## data = arthritis)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.30304 -0.88850 0.01037 0.91087 2.12913   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -4.55477 1.56009 -2.920 0.00351 \*\*  
## TreatmentTreated 1.79705 0.55996 3.209 0.00133 \*\*  
## SexMale 2.75066 2.34871 1.171 0.24155   
## Age 0.07734 0.02775 2.787 0.00532 \*\*  
## SexMale:Age -0.07945 0.04313 -1.842 0.06545 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 116.45 on 83 degrees of freedom  
## Residual deviance: 88.64 on 79 degrees of freedom  
## AIC: 98.64  
##   
## Number of Fisher Scoring iterations: 4

plot(arth\_Model2)

 A 1 year increase in age, with all other variables held fixed, leads to a .07734 increase in log odds of improvements, independant of sex and an additional -.07945 decrease in log odds of improvements IF the candidate is male. This is due to the interaction variable.

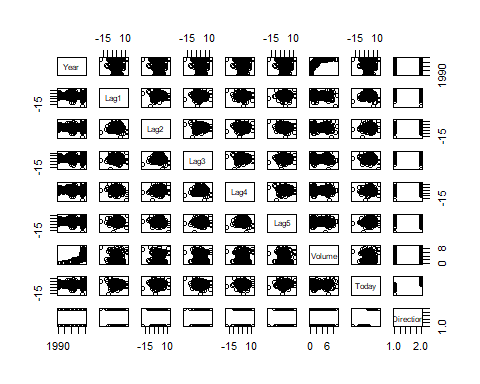
1. (15 pts) Question 10 in Section 4.7.
2. This question should be answered using the Weekly data set, which is part of the ISLR package. This data is similar in nature to the Smarket data from this chapters lab, except that it contains 1,089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.
3. Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?

library(ISLR)

## Warning: package 'ISLR' was built under R version 3.5.3

Weekly

pairs(Weekly)



summary(Weekly)

## Year Lag1 Lag2 Lag3   
## Min. :1990 Min. :-18.1950 Min. :-18.1950 Min. :-18.1950   
## 1st Qu.:1995 1st Qu.: -1.1540 1st Qu.: -1.1540 1st Qu.: -1.1580   
## Median :2000 Median : 0.2410 Median : 0.2410 Median : 0.2410   
## Mean :2000 Mean : 0.1506 Mean : 0.1511 Mean : 0.1472   
## 3rd Qu.:2005 3rd Qu.: 1.4050 3rd Qu.: 1.4090 3rd Qu.: 1.4090   
## Max. :2010 Max. : 12.0260 Max. : 12.0260 Max. : 12.0260   
## Lag4 Lag5 Volume   
## Min. :-18.1950 Min. :-18.1950 Min. :0.08747   
## 1st Qu.: -1.1580 1st Qu.: -1.1660 1st Qu.:0.33202   
## Median : 0.2380 Median : 0.2340 Median :1.00268   
## Mean : 0.1458 Mean : 0.1399 Mean :1.57462   
## 3rd Qu.: 1.4090 3rd Qu.: 1.4050 3rd Qu.:2.05373   
## Max. : 12.0260 Max. : 12.0260 Max. :9.32821   
## Today Direction   
## Min. :-18.1950 Down:484   
## 1st Qu.: -1.1540 Up :605   
## Median : 0.2410   
## Mean : 0.1499   
## 3rd Qu.: 1.4050   
## Max. : 12.0260

Volume is positively correleted with year.

1. Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors.

Weekly\_Model1 <- glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume, data=Weekly,family=binomial)

Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

summary(Weekly\_Model1)

##   
## 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

Lag2 is the only significant predictor.

1. 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.

probs <- predict(Weekly\_Model1, type="response")  
preds <- rep("Down", 1089)  
preds[probs > 0.5] = "Up"  
table(preds, Weekly$Direction)

##   
## preds Down Up  
## Down 54 48  
## Up 430 557

Investors are much more likely to bet on a valuation increasing (and end up overvaluing an option in the market) vs. undervaluing a stock that value actually goes up. However, over 50% of predictions are correct (430 “false positives”). This is statistically not bad but I would argue dangerous when actually applied to the markets.

1. 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).

#Create training data  
Weekly\_training <- Weekly[Weekly$Year<2009,]  
Weekly\_test <- Weekly[Weekly$Year>2008,]  
Weekly\_training\_glm <- glm(Direction~Lag2, data= Weekly\_training, family = "binomial")  
summary(Weekly\_training\_glm)

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

#Confusion Matrix  
probs <- predict(Weekly\_training\_glm, type="response")  
preds <- rep("Down", 985)  
preds[probs > 0.5] = "Up"  
table(preds, Weekly\_training$Direction)

##   
## preds Down Up  
## Down 23 20  
## Up 418 524

correctpred <- (524+23)/985  
correctpred

## [1] 0.5553299

1. Repeat (d) using LDA.

Weekly\_training\_lda <- lda(Direction~Lag2, data=Weekly\_training)  
Weekly\_training\_lda

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

#Confusion Matrix  
pred <- predict(Weekly\_training\_lda,newdata = Weekly\_test, type="response")  
class <- pred$class  
table(class,Weekly\_test$Direction)

##   
## class Down Up  
## Down 9 5  
## Up 34 56

correctpred <- (56+9)/(56+5+9+34)  
correctpred

## [1] 0.625

1. Repeat (d) using QDA.

Weekly\_training\_qda <- qda(Direction~Lag2, data=Weekly\_training)  
Weekly\_training\_qda

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

#Confusion Matrix  
pred <- predict(Weekly\_training\_qda,newdata = Weekly\_test, type="response")  
class <- pred$class  
table(class,Weekly\_test$Direction)

##   
## class Down Up  
## Down 0 0  
## Up 43 61

correctpred <- (43+61)/(43+61)  
correctpred

## [1] 1

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

library(class)  
Weekly\_training\_predictor <- cbind(Weekly\_training$Lag2)  
Weekly\_test\_predictor <- cbind(Weekly\_test$Lag2)  
Weekly\_training\_outcome <- cbind(Weekly\_training$Direction)  
Weekly\_Model\_knn <- knn(Weekly\_training\_predictor, Weekly\_test\_predictor, Weekly\_training\_outcome, k=1)  
table(Weekly\_Model\_knn, Weekly\_test$Direction)

##   
## Weekly\_Model\_knn Down Up  
## 1 21 30  
## 2 22 31

correctpred <- (31+21)/(21+22+30+31)  
correctpred

## [1] 0.5

1. Which of these methods appears to provide the best results on this data?

The QDA method appears to perform the best! With 100% accuracy.

1. Experiment with different combinations of predictors, including possible transformations and interactions, for each of the methods. Report the variables, method, and associated confusion matrix that appears to provide the best results on the held out data. Note that you should also experiment with values for K in the KNN classifier

#EDA  
cor(Weekly[,1:8])

## Year Lag1 Lag2 Lag3 Lag4  
## Year 1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923  
## Lag1 -0.03228927 1.000000000 -0.07485305 0.05863568 -0.071273876  
## Lag2 -0.03339001 -0.074853051 1.00000000 -0.07572091 0.058381535  
## Lag3 -0.03000649 0.058635682 -0.07572091 1.00000000 -0.075395865  
## Lag4 -0.03112792 -0.071273876 0.05838153 -0.07539587 1.000000000  
## Lag5 -0.03051910 -0.008183096 -0.07249948 0.06065717 -0.075675027  
## Volume 0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617  
## Today -0.03245989 -0.075031842 0.05916672 -0.07124364 -0.007825873  
## Lag5 Volume Today  
## Year -0.030519101 0.84194162 -0.032459894  
## Lag1 -0.008183096 -0.06495131 -0.075031842  
## Lag2 -0.072499482 -0.08551314 0.059166717  
## Lag3 0.060657175 -0.06928771 -0.071243639  
## Lag4 -0.075675027 -0.06107462 -0.007825873  
## Lag5 1.000000000 -0.05851741 0.011012698  
## Volume -0.058517414 1.00000000 -0.033077783  
## Today 0.011012698 -0.03307778 1.000000000

library("Hmisc")

## Warning: package 'Hmisc' was built under R version 3.5.3

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

## Loading required package: ggplot2

##   
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':  
##   
## format.pval, units

res2 <- rcorr(as.matrix(Weekly[,1:8]))  
res2

## Year Lag1 Lag2 Lag3 Lag4 Lag5 Volume Today  
## Year 1.00 -0.03 -0.03 -0.03 -0.03 -0.03 0.84 -0.03  
## Lag1 -0.03 1.00 -0.07 0.06 -0.07 -0.01 -0.06 -0.08  
## Lag2 -0.03 -0.07 1.00 -0.08 0.06 -0.07 -0.09 0.06  
## Lag3 -0.03 0.06 -0.08 1.00 -0.08 0.06 -0.07 -0.07  
## Lag4 -0.03 -0.07 0.06 -0.08 1.00 -0.08 -0.06 -0.01  
## Lag5 -0.03 -0.01 -0.07 0.06 -0.08 1.00 -0.06 0.01  
## Volume 0.84 -0.06 -0.09 -0.07 -0.06 -0.06 1.00 -0.03  
## Today -0.03 -0.08 0.06 -0.07 -0.01 0.01 -0.03 1.00  
##   
## n= 1089   
##   
##   
## P  
## Year Lag1 Lag2 Lag3 Lag4 Lag5 Volume Today   
## Year 0.2871 0.2709 0.3225 0.3048 0.3143 0.0000 0.2845  
## Lag1 0.2871 0.0135 0.0531 0.0187 0.7874 0.0321 0.0133  
## Lag2 0.2709 0.0135 0.0124 0.0541 0.0167 0.0047 0.0509  
## Lag3 0.3225 0.0531 0.0124 0.0128 0.0454 0.0222 0.0187  
## Lag4 0.3048 0.0187 0.0541 0.0128 0.0125 0.0439 0.7964  
## Lag5 0.3143 0.7874 0.0167 0.0454 0.0125 0.0535 0.7166  
## Volume 0.0000 0.0321 0.0047 0.0222 0.0439 0.0535 0.2754  
## Today 0.2845 0.0133 0.0509 0.0187 0.7964 0.7166 0.2754

#Chose to add predictors closest to significant   
  
#QDA  
Weekly\_training\_qda <- qda(Direction~Lag2+Lag1+Lag4, data=Weekly\_training)  
Weekly\_training\_qda

## Call:  
## qda(Direction ~ Lag2 + Lag1 + Lag4, data = Weekly\_training)  
##   
## Prior probabilities of groups:  
## Down Up   
## 0.4477157 0.5522843   
##   
## Group means:  
## Lag2 Lag1 Lag4  
## Down -0.03568254 0.289444444 0.15925624  
## Up 0.26036581 -0.009213235 0.09220956

#Confusion Matrix  
pred <- predict(Weekly\_training\_qda,newdata = Weekly\_test, type="response")  
class <- pred$class  
table(class,Weekly\_test$Direction)

##   
## class Down Up  
## Down 9 20  
## Up 34 41

#KNN  
Weekly\_training\_predictor <- cbind(Weekly\_training$Lag2,Weekly\_training$Lag1,Weekly\_training$Lag4)  
Weekly\_test\_predictor <- cbind(Weekly\_test$Lag2,Weekly\_training$Lag1,Weekly\_training$Lag4)

## Warning in cbind(Weekly\_test$Lag2, Weekly\_training$Lag1,  
## Weekly\_training$Lag4): number of rows of result is not a multiple of vector  
## length (arg 1)

Weekly\_training\_outcome <- cbind(Weekly\_training$Direction)  
Weekly\_Model\_knn <- knn(Weekly\_training\_predictor, Weekly\_test\_predictor, Weekly\_training\_outcome, k=1)  
#table(Weekly\_Model\_knn, Weekly\_test$Direction)

I stopped here as we can see that adding predictors increased our error rate and did not help us in our prediction.

1. (15 pts) Question 11 in Section 4.7.
2. In this problem, you will develop a model to predict whether a given car gets high or low gas mileage based on the Auto data set.
3. Create a binary variable, mpg01, that contains a 1 if mpg contains a value above its median, and a 0 if mpg contains a value below its median. You can compute the median using the median() function. Note you may ï¬nd it helpful to use the data.frame() function to create a single data set containing both mpg01 and the other Auto variables.

median(Auto$mpg)

## [1] 22.75

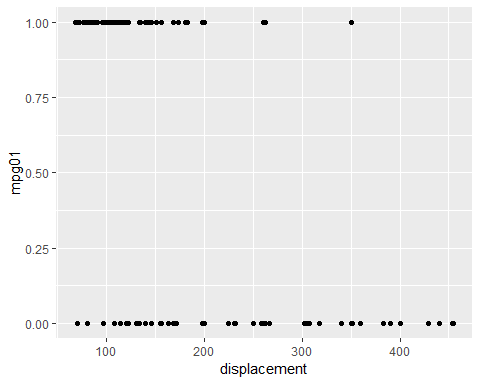
Auto$mpg01 <- 0   
Auto$mpg01[Auto$mpg>median(Auto$mpg)] <- 1  
head(Auto,30)

## mpg cylinders displacement horsepower weight acceleration year origin  
## 1 18 8 307 130 3504 12.0 70 1  
## 2 15 8 350 165 3693 11.5 70 1  
## 3 18 8 318 150 3436 11.0 70 1  
## 4 16 8 304 150 3433 12.0 70 1  
## 5 17 8 302 140 3449 10.5 70 1  
## 6 15 8 429 198 4341 10.0 70 1  
## 7 14 8 454 220 4354 9.0 70 1  
## 8 14 8 440 215 4312 8.5 70 1  
## 9 14 8 455 225 4425 10.0 70 1  
## 10 15 8 390 190 3850 8.5 70 1  
## 11 15 8 383 170 3563 10.0 70 1  
## 12 14 8 340 160 3609 8.0 70 1  
## 13 15 8 400 150 3761 9.5 70 1  
## 14 14 8 455 225 3086 10.0 70 1  
## 15 24 4 113 95 2372 15.0 70 3  
## 16 22 6 198 95 2833 15.5 70 1  
## 17 18 6 199 97 2774 15.5 70 1  
## 18 21 6 200 85 2587 16.0 70 1  
## 19 27 4 97 88 2130 14.5 70 3  
## 20 26 4 97 46 1835 20.5 70 2  
## 21 25 4 110 87 2672 17.5 70 2  
## 22 24 4 107 90 2430 14.5 70 2  
## 23 25 4 104 95 2375 17.5 70 2  
## 24 26 4 121 113 2234 12.5 70 2  
## 25 21 6 199 90 2648 15.0 70 1  
## 26 10 8 360 215 4615 14.0 70 1  
## 27 10 8 307 200 4376 15.0 70 1  
## 28 11 8 318 210 4382 13.5 70 1  
## 29 9 8 304 193 4732 18.5 70 1  
## 30 27 4 97 88 2130 14.5 71 3  
## name mpg01  
## 1 chevrolet chevelle malibu 0  
## 2 buick skylark 320 0  
## 3 plymouth satellite 0  
## 4 amc rebel sst 0  
## 5 ford torino 0  
## 6 ford galaxie 500 0  
## 7 chevrolet impala 0  
## 8 plymouth fury iii 0  
## 9 pontiac catalina 0  
## 10 amc ambassador dpl 0  
## 11 dodge challenger se 0  
## 12 plymouth 'cuda 340 0  
## 13 chevrolet monte carlo 0  
## 14 buick estate wagon (sw) 0  
## 15 toyota corona mark ii 1  
## 16 plymouth duster 0  
## 17 amc hornet 0  
## 18 ford maverick 0  
## 19 datsun pl510 1  
## 20 volkswagen 1131 deluxe sedan 1  
## 21 peugeot 504 1  
## 22 audi 100 ls 1  
## 23 saab 99e 1  
## 24 bmw 2002 1  
## 25 amc gremlin 0  
## 26 ford f250 0  
## 27 chevy c20 0  
## 28 dodge d200 0  
## 29 hi 1200d 0  
## 30 datsun pl510 1

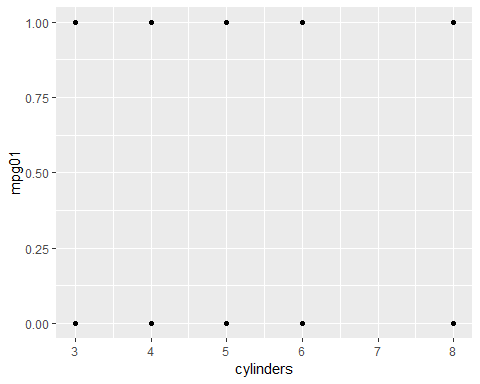
1. Explore the data graphically in order to investigate the association between mpg01 and the other features. Which of the other features seem most likely to be useful in predicting mpg01? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.

lapply(list("displacement","cylinders","horsepower","weight","acceleration","year","origin"),   
 function(i) ggplot(Auto, aes\_string(x=i, y="mpg01")) + geom\_point())

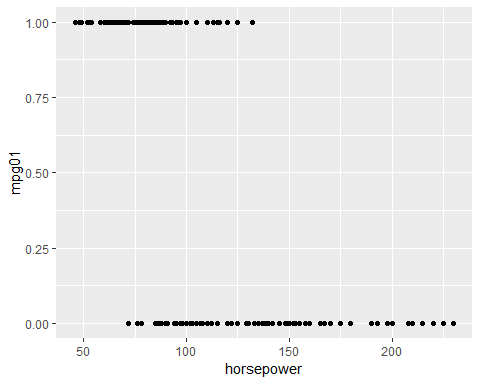
## [[1]]



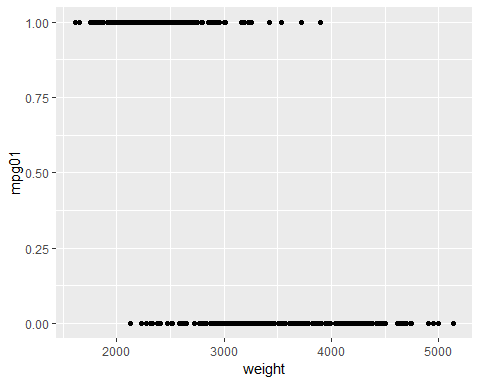
##   
## [[2]]



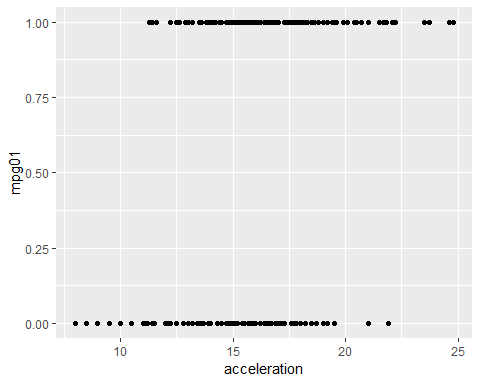
##   
## [[3]]



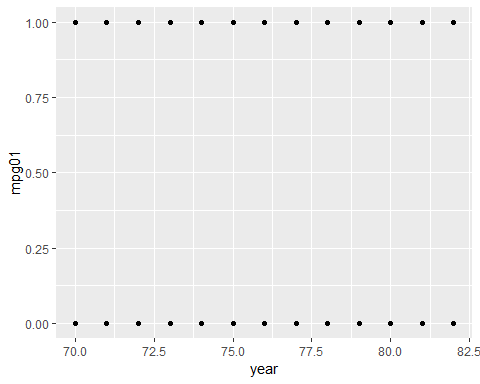
##   
## [[4]]



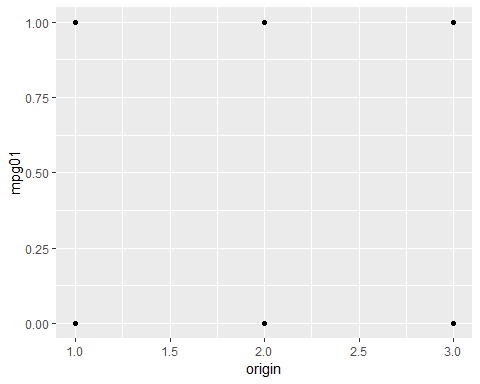
##   
## [[5]]



##   
## [[6]]



##   
## [[7]]



-Cars with lower displacement, on average, have a higher mpg -Cylinders and mpg have no real relationship -Cars with lower horsepower tend to have higher mpg -Lighter cars have a higher mpg -As acceleration increases, on avg, so does mpg

1. Split the data into a training set and a test set.

To do this I used a new package I found: caTools

library(caTools)

set.seed(123) # set seed to ensure you always have same random numbers generated  
sample = sample.split(Auto,SplitRatio = 0.75) # splits the data in the ratio mentioned in SplitRatio. After splitting marks these rows as logical TRUE and the the remaining are marked as logical FALSE  
train1 =subset(Auto,sample ==TRUE) # creates a training dataset named train1 with rows which are marked as TRUE  
test1=subset(Auto, sample==FALSE)

1. Perform LDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

Auto\_training\_lda <- lda(mpg01~acceleration+horsepower+weight, data=train1)  
Auto\_training\_lda

## Call:  
## lda(mpg01 ~ acceleration + horsepower + weight, data = train1)  
##   
## Prior probabilities of groups:  
## 0 1   
## 0.4981818 0.5018182   
##   
## Group means:  
## acceleration horsepower weight  
## 0 14.61971 128.94891 3586.978  
## 1 16.37536 79.21739 2346.471  
##   
## Coefficients of linear discriminants:  
## LD1  
## acceleration 0.0299487831  
## horsepower -0.0002462394  
## weight -0.0017314158

#Confusion Matrix  
pred <- predict(Auto\_training\_lda,newdata = test1, type="response")  
class <- pred$class  
table(class,test1$mpg01)

##   
## class 0 1  
## 0 51 2  
## 1 8 56

correctpred <- (56+51)/(10+56+51)  
correctpred

## [1] 0.9145299

The test error is (1-correctpred)

1-correctpred

## [1] 0.08547009

1. Perform QDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

Auto\_training\_qda <- qda(mpg01~acceleration+horsepower+weight, data=train1)  
Auto\_training\_qda

## Call:  
## qda(mpg01 ~ acceleration + horsepower + weight, data = train1)  
##   
## Prior probabilities of groups:  
## 0 1   
## 0.4981818 0.5018182   
##   
## Group means:  
## acceleration horsepower weight  
## 0 14.61971 128.94891 3586.978  
## 1 16.37536 79.21739 2346.471

#Confusion Matrix  
pred <- predict(Auto\_training\_qda,newdata = test1, type="response")  
class <- pred$class  
table(class,test1$mpg01)

##   
## class 0 1  
## 0 52 3  
## 1 7 55

correctpred <- (55+52)/(55+52+10)  
correctpred

## [1] 0.9145299

error\_rate <- 1-correctpred  
error\_rate

## [1] 0.08547009

1. Perform logistic regression on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

Auto\_training\_glm <- glm(mpg01~acceleration+horsepower+weight, family = "binomial", data=train1)  
summary(Auto\_training\_glm)

##   
## Call:  
## glm(formula = mpg01 ~ acceleration + horsepower + weight, family = "binomial",   
## data = train1)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.19516 -0.26238 0.09789 0.42866 2.92539   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 12.4643477 3.1080838 4.010 6.06e-05 \*\*\*  
## acceleration -0.0102397 0.1458716 -0.070 0.9440   
## horsepower -0.0484565 0.0228826 -2.118 0.0342 \*   
## weight -0.0026908 0.0006848 -3.930 8.51e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 381.23 on 274 degrees of freedom  
## Residual deviance: 165.50 on 271 degrees of freedom  
## AIC: 173.5  
##   
## Number of Fisher Scoring iterations: 7

#Confusion Matrix  
probs <- predict(Auto\_training\_glm, type="response")  
preds <- rep("0", 275)  
preds[probs > 0.5] = "1"  
table(preds, train1$mpg01)

##   
## preds 0 1  
## 0 116 14  
## 1 21 124

correctpred <- (124+116)/(124+116+21+14)  
correctpred

## [1] 0.8727273

errorrate <- 1-correctpred  
errorrate

## [1] 0.1272727

1. Perform KNN on the training data, with several values of K, in order to predict mpg01. Use only the variables that seemed most associated with mpg01 in (b). What test errors do you obtain? Which value of K seems to perform the best on this data set?

library(class)  
#k=1  
Auto\_training\_predictor <- cbind(train1$acceleration,train1$horsepower,train1$weight)  
Auto\_test\_predictor <- cbind(test1$acceleration,test1$horsepower,test1$weight)  
Auto\_training\_outcome <- cbind(train1$mpg01)  
Auto\_Model\_knn <- knn(Auto\_training\_predictor, Auto\_test\_predictor, Auto\_training\_outcome, k=1)  
  
print("K=1")

## [1] "K=1"

table(Auto\_Model\_knn, test1$mpg01)

##   
## Auto\_Model\_knn 0 1  
## 0 51 7  
## 1 8 51

#k=2  
Auto\_training\_predictor <- cbind(train1$acceleration,train1$horsepower,train1$weight)  
Auto\_test\_predictor <- cbind(test1$acceleration,test1$horsepower,test1$weight)  
Auto\_training\_outcome <- cbind(train1$mpg01)  
Auto\_Model\_knn <- knn(Auto\_training\_predictor, Auto\_test\_predictor, Auto\_training\_outcome, k=2)  
  
print("K=2")

## [1] "K=2"

table(Auto\_Model\_knn, test1$mpg01)

##   
## Auto\_Model\_knn 0 1  
## 0 53 7  
## 1 6 51

#k=3  
Auto\_training\_predictor <- cbind(train1$acceleration,train1$horsepower,train1$weight)  
Auto\_test\_predictor <- cbind(test1$acceleration,test1$horsepower,test1$weight)  
Auto\_training\_outcome <- cbind(train1$mpg01)  
Auto\_Model\_knn <- knn(Auto\_training\_predictor, Auto\_test\_predictor, Auto\_training\_outcome, k=3)  
  
print("K=3")

## [1] "K=3"

table(Auto\_Model\_knn, test1$mpg01)

##   
## Auto\_Model\_knn 0 1  
## 0 54 4  
## 1 5 54

#k=5  
Auto\_training\_predictor <- cbind(train1$acceleration,train1$horsepower,train1$weight)  
Auto\_test\_predictor <- cbind(test1$acceleration,test1$horsepower,test1$weight)  
Auto\_training\_outcome <- cbind(train1$mpg01)  
Auto\_Model\_knn <- knn(Auto\_training\_predictor, Auto\_test\_predictor, Auto\_training\_outcome, k=5)  
  
print("K=5")

## [1] "K=5"

table(Auto\_Model\_knn, test1$mpg01)

##   
## Auto\_Model\_knn 0 1  
## 0 54 5  
## 1 5 53

#K=10  
Auto\_training\_predictor <- cbind(train1$acceleration,train1$horsepower,train1$weight)  
Auto\_test\_predictor <- cbind(test1$acceleration,test1$horsepower,test1$weight)  
Auto\_training\_outcome <- cbind(train1$mpg01)  
Auto\_Model\_knn <- knn(Auto\_training\_predictor, Auto\_test\_predictor, Auto\_training\_outcome, k=10)  
  
print("K=10")

## [1] "K=10"

table(Auto\_Model\_knn, test1$mpg01)

##   
## Auto\_Model\_knn 0 1  
## 0 54 4  
## 1 5 54

#k=21  
Auto\_training\_predictor <- cbind(train1$acceleration,train1$horsepower,train1$weight)  
Auto\_test\_predictor <- cbind(test1$acceleration,test1$horsepower,test1$weight)  
Auto\_training\_outcome <- cbind(train1$mpg01)  
Auto\_Model\_knn <- knn(Auto\_training\_predictor, Auto\_test\_predictor, Auto\_training\_outcome, k=21)  
  
print("K=21")

## [1] "K=21"

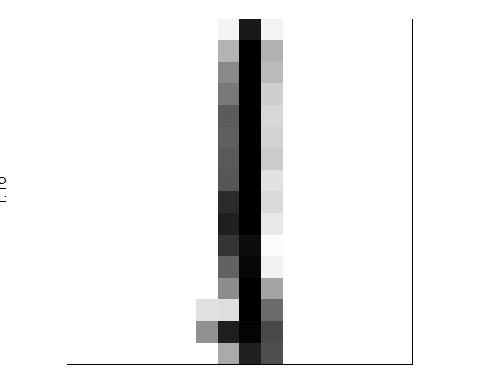
table(Auto\_Model\_knn, test1$mpg01)

##   
## Auto\_Model\_knn 0 1  
## 0 54 3  
## 1 5 55

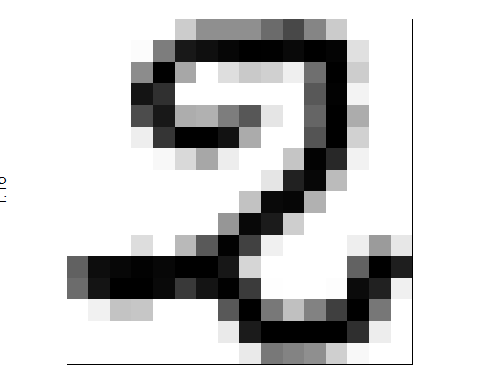
K=21 performed the best with thee samples, but not significantly different from K=5=3=10.

1. (10 pts) Download the csv files zipcode\_train.csv and zipcode\_test.csv. Load and visualize the files using the following command lines.

train.dat <- read.csv("zipcode\_train.csv")  
train.dat$Y <- as.factor(train.dat$Y)  
test.dat <- read.csv("zipcode\_test.csv")  
test.dat$Y <- as.factor(test.dat$Y)  
COLORS <- c("white", "black")  
CUSTOM\_COLORS <- colorRampPalette(colors = COLORS)  
vis <- function(i){  
par(pty = "s", mar = c(1, 1, 1, 1), xaxt = "n", yaxt = "n")  
z <- matrix(as.numeric(train.dat[i,1:256]), 16, 16)  
image(1:16,1:16,z[,16:1], col = CUSTOM\_COLORS(256))  
}  
vis(2) # hand written 1 (from 1 to 1005)



vis(1500) # hand written 2 (from 1006 to 1736)



head(test.dat)

## p1 p2 p3 p4 p5 p6 p7 p8 p9 p10  
## 1 -0.996 0.572 0.396 0.063 -0.506 -0.847 -1.000 -1.000 -1.000 -1.000  
## 2 -1.000 -1.000 0.469 0.413 1.000 1.000 0.462 -0.116 -0.937 -1.000  
## 3 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -0.586 0.693 1.000 0.802  
## 4 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000  
## 5 -1.000 -1.000 -1.000 -0.831 0.047 0.140 0.947 0.813 0.012 -0.768  
## 6 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -0.665 0.603 1.000 0.646  
## p11 p12 p13 p14 p15 p16 p17 p18 p19 p20 p21 p22 p23  
## 1 -1.000 -1 -1 -1 -1 -1 -1 -0.391 0.974 1.000 1.000 0.954 0.356  
## 2 -1.000 -1 -1 -1 -1 -1 -1 -1.000 -0.393 0.822 0.840 0.996 1.000  
## 3 -0.524 -1 -1 -1 -1 -1 -1 -1.000 -1.000 -1.000 -1.000 -0.998 0.582  
## 4 -1.000 -1 -1 -1 -1 -1 -1 -1.000 -1.000 -1.000 -0.969 -0.286 0.487  
## 5 -1.000 -1 -1 -1 -1 -1 -1 -1.000 -0.563 0.715 1.000 1.000 1.000  
## 6 -0.836 -1 -1 -1 -1 -1 -1 -1.000 -1.000 -1.000 -1.000 -0.232 0.848  
## p24 p25 p26 p27 p28 p29 p30 p31 p32 p33 p34 p35 p36  
## 1 -0.470 -1.000 -1.000 -1.000 -1.000 -1 -1 -1 -1 -1 -1 -0.716 -0.170  
## 2 1.000 0.697 -0.597 -1.000 -1.000 -1 -1 -1 -1 -1 -1 -1.000 -1.000  
## 3 1.000 1.000 0.798 -0.446 -1.000 -1 -1 -1 -1 -1 -1 -1.000 -1.000  
## 4 0.934 0.856 -0.269 -0.869 -1.000 -1 -1 -1 -1 -1 -1 -1.000 -0.719  
## 5 1.000 1.000 0.976 0.039 -0.905 -1 -1 -1 -1 -1 -1 0.056 1.000  
## 6 0.915 0.585 1.000 0.683 -0.799 -1 -1 -1 -1 -1 -1 -1.000 -1.000  
## p37 p38 p39 p40 p41 p42 p43 p44 p45 p46 p47 p48  
## 1 0.307 0.851 1.000 0.955 -0.228 -1.000 -1.000 -1.000 -1 -1 -1 -1  
## 2 -1.000 -0.567 -0.405 0.376 0.919 0.945 -0.536 -1.000 -1 -1 -1 -1  
## 3 -1.000 -0.856 0.932 1.000 0.859 -0.771 -1.000 -1.000 -1 -1 -1 -1  
## 4 0.612 0.996 1.000 1.000 1.000 1.000 0.716 -0.600 -1 -1 -1 -1  
## 5 1.000 1.000 1.000 1.000 1.000 1.000 1.000 -0.034 -1 -1 -1 -1  
## 6 0.293 1.000 0.613 -0.781 -1.000 -0.081 1.000 0.081 -1 -1 -1 -1  
## p49 p50 p51 p52 p53 p54 p55 p56 p57 p58 p59  
## 1 -1 -1 -1.000 -1.000 -1.000 -0.975 -0.422 0.581 0.996 -0.129 -0.993  
## 2 -1 -1 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -0.660 0.678 0.858  
## 3 -1 -1 -1.000 -1.000 -1.000 -0.432 1.000 1.000 0.079 -1.000 -1.000  
## 4 -1 -1 -0.997 0.415 1.000 1.000 1.000 1.000 1.000 1.000 1.000  
## 5 -1 -1 0.538 1.000 1.000 1.000 0.692 -0.081 -0.067 0.932 1.000  
## 6 -1 -1 -1.000 -0.592 0.937 0.774 -0.744 -1.000 -1.000 -0.881 0.851  
## p60 p61 p62 p63 p64 p65 p66 p67 p68 p69 p70 p71  
## 1 -1.000 -1.000 -1 -1 -1 -1 -1 -1.00 -1.000 -1.000 -1.000 -1.000  
## 2 -0.704 -1.000 -1 -1 -1 -1 -1 -1.00 -1.000 -1.000 -1.000 -1.000  
## 3 -1.000 -1.000 -1 -1 -1 -1 -1 -1.00 -1.000 -1.000 -0.518 1.000  
## 4 0.537 -1.000 -1 -1 -1 -1 -1 -1.00 0.372 1.000 0.884 0.089  
## 5 0.843 -0.911 -1 -1 -1 -1 -1 -0.16 0.942 0.716 0.125 -0.849  
## 6 0.828 -0.923 -1 -1 -1 -1 -1 -1.00 -0.478 0.970 -0.492 -1.000  
## p72 p73 p74 p75 p76 p77 p78 p79 p80 p81 p82 p83 p84  
## 1 -0.867 0.494 1.000 -0.366 -1.000 -1.000 -1 -1 -1 -1 -1 -1 -1.000  
## 2 -1.000 -1.000 -0.561 0.911 0.663 -0.968 -1 -1 -1 -1 -1 -1 -1.000  
## 3 1.000 -0.413 -1.000 -1.000 -1.000 -1.000 -1 -1 -1 -1 -1 -1 -1.000  
## 4 -0.383 -0.741 0.211 1.000 0.998 -0.640 -1 -1 -1 -1 -1 -1 -0.959  
## 5 -1.000 -0.855 0.805 1.000 1.000 -0.101 -1 -1 -1 -1 -1 -1 -0.706  
## 6 -1.000 -1.000 -1.000 0.273 1.000 -0.398 -1 -1 -1 -1 -1 -1 -1.000  
## p85 p86 p87 p88 p89 p90 p91 p92 p93 p94 p95 p96 p97  
## 1 -1.000 -1.000 -1 -1 -0.826 0.967 0.631 -0.99 -1.000 -1 -1 -1 -1  
## 2 -1.000 -1.000 -1 -1 -1.000 -1.000 0.014 1.00 -0.174 -1 -1 -1 -1  
## 3 -1.000 -0.469 1 1 -0.512 -1.000 -1.000 -1.00 -1.000 -1 -1 -1 -1  
## 4 -0.419 -0.723 -1 -1 -1.000 0.563 1.000 1.00 -0.370 -1 -1 -1 -1  
## 5 -0.803 -1.000 -1 -1 -0.381 1.000 1.000 1.00 0.164 -1 -1 -1 -1  
## 6 -1.000 -1.000 -1 -1 -1.000 -1.000 -0.425 1.00 0.175 -1 -1 -1 -1  
## p98 p99 p100 p101 p102 p103 p104 p105 p106 p107 p108 p109  
## 1 -1 -1 -1 -1 -1.000 -1 -1 -0.938 0.915 1.000 -0.386 -1.000  
## 2 -1 -1 -1 -1 -1.000 -1 -1 -1.000 -1.000 -0.995 0.648 0.964  
## 3 -1 -1 -1 -1 -0.235 1 1 -0.352 -1.000 -1.000 -1.000 -1.000  
## 4 -1 -1 -1 -1 -1.000 -1 -1 -0.973 0.488 1.000 1.000 -0.198  
## 5 -1 -1 -1 -1 -1.000 -1 -1 0.149 1.000 1.000 1.000 -0.295  
## 6 -1 -1 -1 -1 -1.000 -1 -1 -1.000 -1.000 -0.865 0.992 0.740  
## p110 p111 p112 p113 p114 p115 p116 p117 p118 p119 p120 p121  
## 1 -1.000 -1 -1 -1 -1 -1 -1 -1 -1.000 -1.000 -1.000 -0.999  
## 2 -0.777 -1 -1 -1 -1 -1 -1 -1 -1.000 -1.000 -1.000 -1.000  
## 3 -1.000 -1 -1 -1 -1 -1 -1 -1 -0.602 0.998 1.000 -0.376  
## 4 -1.000 -1 -1 -1 -1 -1 -1 -1 -1.000 -1.000 -1.000 -0.404  
## 5 -1.000 -1 -1 -1 -1 -1 -1 -1 -1.000 -1.000 -0.722 0.915  
## 6 -0.992 -1 -1 -1 -1 -1 -1 -1 -1.000 -1.000 -1.000 -1.000  
## p122 p123 p124 p125 p126 p127 p128 p129 p130 p131 p132  
## 1 0.629 1 -0.039 -1.000 -1.000 -1 -1 -1.000 -1 -1.00 -1.000  
## 2 -1.000 -1 0.392 1.000 0.062 -1 -1 -1.000 -1 -1.00 -1.000  
## 3 -1.000 -1 -1.000 -1.000 -1.000 -1 -1 -1.000 -1 -1.00 -1.000  
## 4 0.996 1 0.980 -0.605 -1.000 -1 -1 -0.825 -1 -0.89 -0.548  
## 5 1.000 1 0.968 -0.512 -1.000 -1 -1 -1.000 -1 -1.00 -1.000  
## 6 -1.000 -1 0.929 1.000 -0.929 -1 -1 -1.000 -1 -1.00 -1.000  
## p133 p134 p135 p136 p137 p138 p139 p140 p141 p142  
## 1 -1.000 -1.000 -1.000 -1.000 -1.000 0.283 1.000 -0.022 -1.000 -1.000  
## 2 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -0.227 1.000 0.297  
## 3 -1.000 -0.902 0.928 1.000 -0.204 -1.000 -1.000 -1.000 -1.000 -1.000  
## 4 -0.548 -0.548 -0.892 -0.727 0.746 1.000 1.000 0.783 -0.886 -1.000  
## 5 -1.000 -1.000 -1.000 0.190 1.000 1.000 1.000 0.445 -0.986 -1.000  
## 6 -1.000 -0.331 0.216 0.500 0.500 -0.260 -0.918 0.899 0.918 -0.899  
## p143 p144 p145 p146 p147 p148 p149 p150 p151 p152 p153  
## 1 -1 -1 -1.000 -1.000 -1.000 -0.956 -0.450 0.274 0.283 0.283 0.232  
## 2 -1 -1 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000  
## 3 -1 -1 -1.000 -1.000 -1.000 -1.000 -1.000 -0.997 0.679 1.000 -0.229  
## 4 -1 -1 0.236 -0.724 0.595 1.000 1.000 1.000 0.780 0.845 1.000  
## 5 -1 -1 -0.981 0.404 0.838 0.838 0.684 -0.027 0.735 0.953 1.000  
## 6 -1 -1 -1.000 -1.000 -1.000 -1.000 0.033 1.000 1.000 1.000 1.000  
## p154 p155 p156 p157 p158 p159 p160 p161 p162 p163  
## 1 0.399 1.000 0.051 -1.000 -1.000 -1.000 -1 -1.000 -1.000 -0.582  
## 2 -1.000 -1.000 -0.436 1.000 0.878 -0.961 -1 -1.000 -1.000 -1.000  
## 3 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1 -1.000 -1.000 -1.000  
## 4 1.000 0.824 -0.704 -1.000 -1.000 -1.000 -1 -0.418 0.751 1.000  
## 5 1.000 0.974 -0.559 -1.000 -1.000 -1.000 -1 -0.456 0.999 1.000  
## 6 1.000 0.750 1.000 0.467 -1.000 -1.000 -1 -1.000 -1.000 -0.954  
## p164 p165 p166 p167 p168 p169 p170 p171 p172 p173 p174  
## 1 0.716 1 1.0 1.000 1 1.000 1.000 1.000 0.283 -0.987 -1.000  
## 2 -1.000 -1 -1.0 -1.000 -1 -1.000 -1.000 -1.000 -0.364 1.000 0.981  
## 3 -1.000 -1 -1.0 0.461 1 0.128 -1.000 -1.000 -1.000 -1.000 -1.000  
## 4 1.000 1 1.0 1.000 1 1.000 0.986 -0.491 -1.000 -1.000 -1.000  
## 5 1.000 1 1.0 1.000 1 1.000 0.971 -0.161 -1.000 -1.000 -1.000  
## 6 0.289 1 0.1 -0.809 -1 -0.816 0.400 1.000 1.000 0.165 -1.000  
## p175 p176 p177 p178 p179 p180 p181 p182 p183 p184  
## 1 -1.000 -1 -1.000 -0.230 0.842 1.000 1.000 0.675 -0.022 -0.068  
## 2 -0.855 -1 -1.000 -1.000 -0.407 0.357 0.600 0.562 -0.480 -1.000  
## 3 -1.000 -1 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 0.313 1.000  
## 4 -1.000 -1 -0.982 0.743 1.000 1.000 1.000 1.000 1.000 1.000  
## 5 -1.000 -1 -0.114 1.000 1.000 1.000 1.000 1.000 1.000 1.000  
## 6 -1.000 -1 -1.000 -1.000 -0.863 1.000 0.613 -1.000 -1.000 -1.000  
## p185 p186 p187 p188 p189 p190 p191 p192 p193 p194  
## 1 1.000 1.000 1.000 1.000 0.417 -0.929 -1.00 -1 -1.000 0.428  
## 2 -1.000 -1.000 -1.000 -0.113 1.000 0.918 -0.89 -1 -1.000 -0.843  
## 3 0.325 -1.000 -1.000 -1.000 -1.000 -1.000 -1.00 -1 -1.000 -1.000  
## 4 1.000 1.000 0.326 -0.543 -0.965 -1.000 -1.00 -1 -1.000 -0.299  
## 5 1.000 0.998 0.204 -0.845 -1.000 -1.000 -1.00 -1 0.275 1.000  
## 6 -1.000 -0.363 1.000 1.000 -0.137 -1.000 -1.00 -1 -1.000 -1.000  
## p195 p196 p197 p198 p199 p200 p201 p202 p203 p204  
## 1 1.000 0.764 -0.277 -0.916 -0.101 0.768 1.000 0.975 -0.224 -0.145  
## 2 0.951 1.000 1.000 1.000 0.966 0.104 -0.914 -1.000 -0.436 0.778  
## 3 -1.000 -1.000 -1.000 -1.000 0.141 1.000 0.595 -0.999 -1.000 -1.000  
## 4 0.964 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000  
## 5 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.777  
## 6 -0.436 1.000 0.061 -1.000 -1.000 -0.352 0.541 1.000 1.000 1.000  
## p205 p206 p207 p208 p209 p210 p211 p212 p213 p214  
## 1 0.923 0.642 -0.256 -1.000 -1.000 0.719 1.000 -0.163 0.039 0.824  
## 2 1.000 0.749 -0.972 -1.000 -1.000 -0.917 0.917 1.000 1.000 0.886  
## 3 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000 -1.000  
## 4 0.596 0.485 0.229 -0.576 -1.000 -1.000 -0.152 0.896 1.000 1.000  
## 5 -0.472 -0.946 -0.955 -1.000 -0.595 0.997 1.000 1.000 1.000 1.000  
## 6 0.561 -1.000 -1.000 -1.000 -1.000 -1.000 -0.870 0.986 0.750 1.000  
## p215 p216 p217 p218 p219 p220 p221 p222 p223 p224 p225  
## 1 1.000 1.000 0.940 0.030 -1.000 -1 -0.749 -0.016 0.040 -0.996 -1  
## 2 0.807 1.000 0.828 0.760 0.951 1 1.000 -0.243 -1.000 -1.000 -1  
## 3 -0.351 1.000 0.731 -0.988 -1.000 -1 -1.000 -1.000 -1.000 -1.000 -1  
## 4 0.903 0.143 -0.753 0.348 0.909 1 1.000 1.000 1.000 0.448 -1  
## 5 1.000 0.693 0.675 0.929 1.000 1 1.000 1.000 0.938 -0.286 -1  
## 6 1.000 1.000 1.000 0.638 -0.014 1 0.995 -0.611 -1.000 -1.000 -1  
## p226 p227 p228 p229 p230 p231 p232 p233 p234 p235  
## 1 0.554 1.000 1.000 1.000 1.000 0.791 0.325 -0.673 -1.000 -1.000  
## 2 -0.999 0.432 1.000 1.000 0.973 0.925 1.000 1.000 1.000 1.000  
## 3 -1.000 -1.000 -1.000 -1.000 -1.000 -0.843 0.851 0.977 -0.593 -1.000  
## 4 -1.000 -1.000 -0.634 0.325 0.593 -0.245 -1.000 -1.000 -1.000 -0.735  
## 5 -0.196 1.000 1.000 1.000 1.000 0.066 -0.914 -1.000 -0.634 0.097  
## 6 -1.000 -1.000 -0.305 0.722 1.000 0.652 0.398 -0.232 -0.986 -0.582  
## p236 p237 p238 p239 p240 p241 p242 p243 p244 p245  
## 1 -1.000 -1.000 -1.000 -1.00 -1.000 -1 -0.605 0.718 0.972 0.398  
## 2 0.793 -0.136 -0.969 -1.00 -1.000 -1 -1.000 -0.979 -0.114 0.552  
## 3 -1.000 -1.000 -1.000 -1.00 -1.000 -1 -1.000 -1.000 -1.000 -1.000  
## 4 0.000 0.160 0.160 -0.38 -0.867 -1 -1.000 -1.000 -1.000 -1.000  
## 5 0.763 1.000 1.000 1.00 0.338 -1 -0.996 0.226 1.000 0.936  
## 6 1.000 1.000 -0.543 -1.00 -1.000 -1 -1.000 -1.000 -1.000 -1.000  
## p246 p247 p248 p249 p250 p251 p252 p253 p254 p255  
## 1 0.165 -0.668 -1.000 -1.000 -1.000 -1.00 -1.000 -1.000 -1.000 -1.000  
## 2 1.000 1.000 1.000 1.000 0.270 -0.28 -0.855 -1.000 -1.000 -1.000  
## 3 -1.000 -1.000 -0.601 0.592 0.219 -1.00 -1.000 -1.000 -1.000 -1.000  
## 4 -1.000 -1.000 -1.000 -1.000 -1.000 -1.00 -1.000 -1.000 -1.000 -1.000  
## 5 -0.221 -0.915 -1.000 -1.000 -1.000 -1.00 -0.866 -0.672 0.131 0.135  
## 6 -1.000 -1.000 -1.000 -1.000 -1.000 -1.00 0.720 0.711 -0.932 -1.000  
## p256 Y  
## 1 -1.000 2  
## 2 -1.000 2  
## 3 -1.000 1  
## 4 -1.000 2  
## 5 -0.318 2  
## 6 -1.000 2

In the dataset, you have 1736 training images and 462 test images, where each image is a handwritten digit and it can be either 1 or 2. The description of columns is below:

p1-256: the gray scale from -1 to 1. Y: the digit, which is either 1 or 2.

1. Perform logistic regression, LDA, and KNN models.

print("LDA")

## [1] "LDA"

test.dat2 <- test.dat[,-c(16,32)]  
train.dat2 <- train.dat[,-c(16,32)]   
# Remove co linear variables   
image\_training\_lda <- lda(Y~., data=train.dat2)  
  
#Confusion Matrix  
pred <- predict(image\_training\_lda,newdata = test.dat2, type="response")  
class <- pred$class  
table(class,test.dat2$Y)

##   
## class 1 2  
## 1 259 2  
## 2 5 196

correctpred <- (196+259)/(196+259+7)  
correctpred

## [1] 0.9848485

print("glm")

## [1] "glm"

image\_training\_glm <- lda(Y~., data=train.dat2, family=binomial)  
  
#Confusion Matrix  
pred <- predict(image\_training\_glm,newdata = test.dat, type="response")  
class <- pred$class  
table(class,test.dat$Y)

##   
## class 1 2  
## 1 259 2  
## 2 5 196

correctpred <- (196+259)/(196+259+7)  
correctpred

## [1] 0.9848485

print("KNN, K=1")

## [1] "KNN, K=1"

training\_predictordf <- train.dat[,1:256]  
test\_predictordf <- test.dat[,1:256]  
  
image\_training\_outcome <- cbind(train.dat$Y)  
image\_Model\_knn <- knn(training\_predictordf, test\_predictordf, image\_training\_outcome, k=1)  
  
  
print("K=1")

## [1] "K=1"

table(image\_Model\_knn, test.dat$Y)

##   
## image\_Model\_knn 1 2  
## 1 260 2  
## 2 4 196

1. Using the test.dat, which of these methods appears to provide the best results on the test data?

The Knn with k=1 performs the best and has the lowest error rate