Homework 3

Hyeonjin Lee

3/14/2021

Introduction to Statistical Learning in R Homework 3 (Chapter 4):

Question 4

> When the number of features p is large, there tends to be a deterioration in the performance of KNN and other local approaches that perform prediction using only observations that are near the test observation for which a prediction must be made. This phenomenon is known as the curse of dimensionality, and it ties into the fact that curse of dinon-parametric approaches often perform poorly when p is large. We mensionality will now investigate this curse.

a) Suppose that we have a set of observations, each with measurements on p = 1 feature, X. We assume that X is uniformly (evenly) distributed on [0, 1]. Associated with each observation is a response value. Suppose that we wish to predict a test observation's response using only observations that are within 10 % of the range of X closest to that test observation. For instance, in order to predict the response for a test observation with X = 0.6, we will use observations in the range [0.55, 0.65]. On average, what fraction of the available observations will we use to make the prediction?

> We can use the following expression to find the average fraction of available observations.

$$\int_{.05}^{.95} 10dx + \int_{.0}^{.05} (100x + 5)dx + \int_{..95}^{1} (105 - 100x)dx$$

> which equals: 9 + .375 + .375 = 9.75%.

b) Now suppose that we have a set of observations, each with measurements on p = 2 features, X1 and X2. We assume that (X1, X2) are uniformly distributed on [0, 1] × [0, 1]. We wish to predict a test observation's response using only observations that are within 10 % of the range of X1 and within 10 % of the range of X2 closest to that test observation. For instance, in order to predict the response for a test observation with X1 = 0.6 and X2 = 0.35, we will use observations in the range [0.55, 0.65] for X1 and in the range [0.3, 0.4] for X2. On average, what fraction of the available observations will we use to make the prediction?

> We can assume X1 and X2 to be independent. Therefore, the fraction of available observations we use to make the prediction is simply 9.75% * 9.75% or .951%.

c) Now suppose that we have a set of observations on p = 100 features. Again the observations are uniformly distributed on each feature, and again each feature ranges in value from 0 to 1. We wish to predict a test observation's response using observations within the 10 % of each feature's range that is closest to that test observation. What fraction of the available observations will we use to make the prediction?

> Now, the fraction of the available observations will be 9.75% ˆ 100 which is very close to 0%.

d) Using your answers to parts (a)–(c), argue that a drawback of KNN when p is large is that there are very few training observations "near" any given test observation

We can see from parts a - c that the fraction of observations is 9.75% to p power. Therefore, when p approaches infinity, the percentage approaches zero.

e) Now suppose that we wish to make a prediction for a test observation by creating a p-dimensional hypercube centered around the test observation that contains, on average, 10 % of the training observations. For p = 1, 2, and 100, what is the length of each side of the hypercube? Comment on your answer.

Note: A hypercube is a generalization of a cube to an arbitrary number of dimensions. When p = 1, a hypercube is simply a line segment, when p = 2 it is a square, and when p = 100 it is a 100-dimensional cube

We can denote l to be $l = .1^{(1/p)}$
p=1, l = .1
p = 2, l = $.1^{1/2}$
p = 100, l = $.1^{1/100}$

Question 9

This problem has to do with odds.

a) On average, what fraction of people with an odds of 0.37 of defaulting on their credit card payment will in fact default?

Odds of an event is the probability of y occurring divided by the probability of y not occurring (1-y). This can be written as the following:

$$\frac{p(y)}{1-p(y)} = .37$$

> We can rearrange the formula to find p(y) as:

$$p(y) = .37(1-p(y)) = .37/1.37 = .27$$

> Therefore, on average, 27% of people default on their credit card payment.

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

Using the same formula, we can now calculate the odds as the following:

$$\frac{p(y)}{1-p(y)} = \frac{.16}{1-.16} = .19$$

> The odds that this individual will default is .19%

Question 10

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 chapter's lab, except that it contains 1, 089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

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

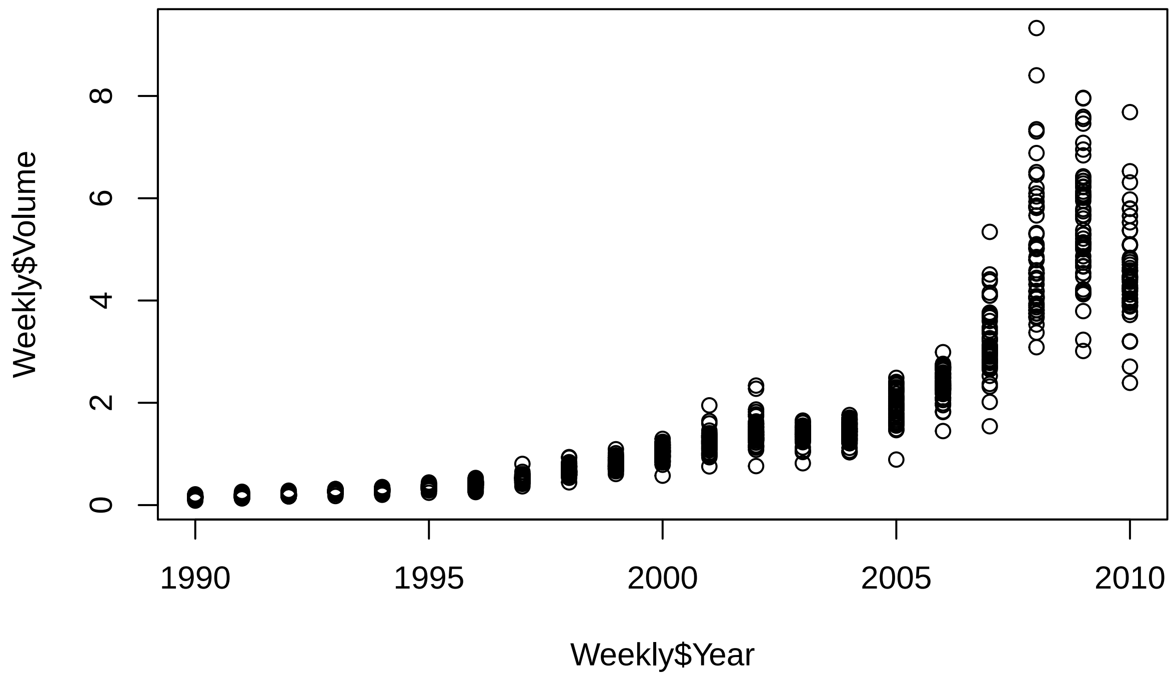
```
library(ISLR)
summary(Weekly)
```

```
##       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             
```

```
cor(Weekly[,-9])
```

```
##               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
```

```r
plot(Weekly$Year,Weekly$Volume)
```



> We can see from the correlation matrix that the only notable association is between Year and Volume. Looking at the graph of Year vs Volume, we can see that as the years increase, there is also an increase in volume.

b) Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

```r
fit1 <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data= Weekly, family = binomial)
summary(fit1)
```

```
##
## 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
```

Only Lag2 is statistically significant is the p-value is less than our alpha of .05.

c) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

```
glm_probs <- predict(fit1, type = "response")
glm_pred <- rep("Down", length(glm_probs))
glm_pred[glm_probs > .5] <- "Up"
table(glm_pred, Weekly$Direction)
```

```
##
## glm_pred Down  Up
##     Down   54  48
##     Up    430 557
```

From the matrix, we can see that the percentage of correct predictions on the training data is (54+557)/(1089) or 56.107%. Therefore, 43.89% is the training error rate. We can also see that for the weeks in which the market went up, our model is correct (557/(557+48)) or 92.07% of the time. For the weeks in which the market went down,the model is correct (54/54+430) or 11.16% of the time.

d)

Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag2 as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 and 2010).

```
train <- (Weekly$Year < 2009)
Weekly_910 <- Weekly[!train, ]
Direction_910 <- Weekly$Direction[!train]
fit2 <- glm(Direction ~ Lag2, data= Weekly, family = binomial, subset = train)
summary(fit2)
```

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

```r
glm_probs2 <- predict(fit2, Weekly_910, type = "response")
glm_pred2 <- rep("Down", length(glm_probs2))
glm_pred2[glm_probs2 > .5] <- "Up"
table(glm_pred2, Direction_910)
```

```
##          Direction_910
## glm_pred2 Down Up
##      Down    9  5
##      Up     34 56
```

From the confusion matrix, we can see that the percentage of correct redictions is (9+56)/104 or 62.5%. Therefore 37.5% is the test error rate. When the market went up, our model was correct (56/56+5) or 91.8% of the time. When the market went down, our model was correct only (9/9+34) or 20.93% of the time.

e) Repeat (d) using LDA

```r
library(MASS)
lda_fit <- lda(Direction ~ Lag2, data = Weekly, subset = train)
lda_fit
```

```
## Call:
## lda(Direction ~ Lag2, data = Weekly, subset = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
```

```
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
##
## Coefficients of linear discriminants:
##            LD1
## Lag2 0.4414162
```

```r
lda_pred <- predict(lda_fit, Weekly_910)
table(lda_pred$class, Direction_910)
```

```
##       Direction_910
##        Down Up
##   Down    9  5
##   Up     34 56
```

From the matrix, we can see that the percentage of correct predictions is 62.5% and thus the test error rate is 37.5% (using the same intuition from previous problems). We can also say that when the market goes up, our model is correct 91.8% of the time and when the market goes down, our model is correct 20.93% of the time. The results are close to the results from the logistic regression.

f) Repeat (d) using QDA

```r
qda_fit <- qda(Direction ~ Lag2, data = Weekly, subset = train)
qda_fit
```

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

```r
qda_pred <- predict(qda_fit, Weekly_910)
table(qda_pred$class, Direction_910)
```

```
##       Direction_910
##        Down Up
##   Down    0  0
##   Up     43 61
```

We can see from the matrix that the percentage of correct predictions is 58.65%. Therefore, 411.35% is the test error rate. For the weeks that the market goes up, our model is correct 100% of the time. For the weeks that the market goes down, our model is correct 0% of the time. However, we should note that our model chooses up the entier time and still receives a correctness of 58.65%.

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

```
library(class)
train_x <- as.matrix(Weekly$Lag2[train])
test_x <- as.matrix(Weekly$Lag2[!train])
train_Direction <- Weekly$Direction[train]
set.seed(1)
knn_pred <- knn(train_x, test_x, train_Direction, k =1)
table(knn_pred, Direction_910)
```

```
##         Direction_910
## knn_pred Down Up
##     Down   21 30
##     Up     22 31
```

From the matrix, we can see that the percentage of correct predictions is 50%. Therefore, the test error rate is also 50%. When the market goes up, the model is correct 50.82% of the time. When the market goes down, our model is correct 48.84% of the time.

h) Which of these methods appears to provide the best results on this data?

We can compare the test error rates of the models. We can see that logistic regression and LDA have the lowest error rates. Then it is QDA and KNN.

i) 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.

```
# We can try the with Lag1: Lag2

fit3 <- glm(Direction ~ Lag1:Lag2, data = Weekly, family = binomial, subset = train)
glm_probs3 <- predict(fit3, Weekly_910, type ="response")
glm_pred3 <- rep("Down", length(glm_probs3))
glm_pred3[glm_probs3 > .5] = "Up"
table(glm_pred3, Direction_910)
```

```
##          Direction_910
## glm_pred3 Down Up
##      Down    1  1
##      Up     42 60
```

```
mean(glm_pred3 == Direction_910)
```

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

```
# LDA with Lag1:Lag2

lda_fit2 <- lda(Direction ~ Lag1:Lag2, data = Weekly, subset = train)
lda_pred2 <- predict(lda_fit2, Weekly_910)
table(lda_pred2$class, Direction_910)
```

```
##       Direction_910
##        Down Up
##   Down    0  1
##   Up     43 60
```

```r
mean(lda_pred2$class == Direction_910)
```

```
## [1] 0.5769231
```

```r
# QDA with Lag1:Lag2

qda_fit2 <- qda(Direction ~ Lag1:Lag2, data = Weekly, subset = train)
qda_pred2 <- predict(qda_fit2, Weekly_910)
table(qda_pred2$class, Direction_910)
```

```
##       Direction_910
##        Down Up
##   Down   16 32
##   Up     27 29
```

```r
mean(qda_pred2$class == Direction_910)
```

```
## [1] 0.4326923
```

```r
# Knn k = 5

knn_pred2 <- knn(train_x, test_x, train_Direction, k = 5)
table(knn_pred2, Direction_910)
```

```
##          Direction_910
## knn_pred2 Down Up
##      Down   15 20
##      Up     28 41
```

```r
mean(knn_pred2 == Direction_910)
```

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

```r
# Knn k = 50
knn_pred3 <- knn(train_x, test_x, train_Direction, k = 50)
table(knn_pred3, Direction_910)
```

```
##          Direction_910
## knn_pred3 Down Up
##      Down   21 20
##      Up     22 41
```

```r
mean(knn_pred3 == Direction_910)
```

```
## [1] 0.5961538
```

Looking at all the models, we can see that the best performance came from the logistic regression and the LDA models when comparing test rate failures.