STATS 790 HW1

Hainan Xu

Question 1

Dr.Breiman presents two different modeling approaches: traditional data model and algorithmic model. He argues in favor of the latter approach, stating that it can produce more reliable results. He conducts a review of both methods and highlights the strengths and weaknesses of each. He also notes that, despite traditional statistical methods being the majority, the use of algorithmic models and machine learning will continue to grow and develop in the future. I think this paper is a very insightful work as it was written in 2001, where machine learning was not as popular as it is now.

Question 2

First, let's load the required data and packages. The data is available at ESL Mixuture Simulation.

```
library(ggplot2)
library(class)
load("ESL.mixture.rda")
```

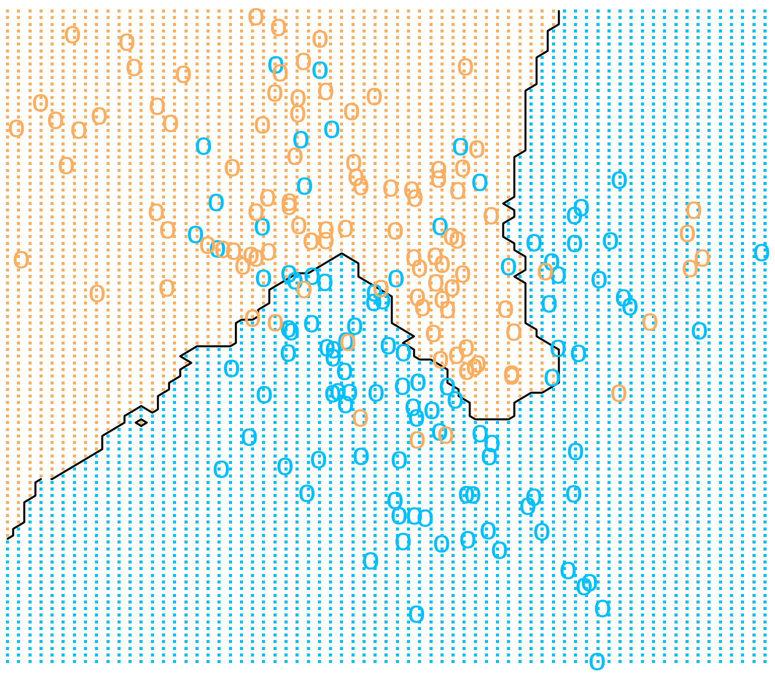
Then, we can generate the grid, and fit a knn model using the given dataset to perform classifications to the grid points.

```
training<-ESL.mixture$x
label<-ESL.mixture$y

#generate the grids
grid<-expand.grid(x=ESL.mixture$px1,y=ESL.mixture$px2)
#make predictions for the grid
grid$y_pred<-knn(train = training,
             test = grid,
             cl = label,
             k = 15,
             prob = TRUE)
```

A decision boundary can be generated using function `contour`. Reference: stack overflow.

```
m <-matrix(grid$y_pred,length(ESL.mixture$px1),length(ESL.mixture$px2))

contour(ESL.mixture$px1,ESL.mixture$px2,m,levels=0.5, labels="", xlab="", ylab="", main="F
points(grid, pch=".", cex=1, col=ifelse(grid$y_pred==1,"#FDAE61", "#00BFFF"))
points(training[,1],training[,2], pch="o", cex=1, col=ifelse(label==1,"#FDAE61", "#00BFFF"
```

Figure 2.2 15 Nearest Neighbor Classifier



Question 3

No, the mean absolute error (MAE) is not minimized by the mean. Instead, since we are minimizing an L1 norm, it is actually minimized by the median.

Both MSE and MAE can measure the differences between the predicted value and the true values, however, they pertain different property. MSE takes L2 norm, which means it is minimized by the mean, and it can be differentiated for the purpose of optimization. MAE is minimized by the median, which means it's more robust and insensitive to outliers compared to MSE.

Question 4

A linear smoother has the form,

$$\hat{\mu}(x) = \sum_i y_i \hat{w}_i(x_i, x)$$

The global means has the form,

$$\mu = \frac{1}{n}\sum_{i=1}^{n} y_i$$

Compare the two, we can deduce $\hat{w}(x_i, x) = \frac{1}{n}J_n$, where J_n is a $n \times n$ matrix of ones.

$$df\hat{\mu} = tr(\hat{w}(x_i, x)) = \frac{1}{n} \times n = 1$$

Thus it has one degree of freedom.

Question 5

Assume the total number of observations is n. If we consider k nearest neighbors regression(KNN) as a linear smoother, the y_i is decided by it's k nearest neighbors. Thus, we can consider that $\hat{y}_i = \frac{1}{k}\sum_{i=1}^{k} x_i$.

Therefore, the influence matrix of KNN is,

$$\hat{w}(x_i, x) = \begin{cases} \frac{1}{k}, x_i \text{ one of k nearest neighbors of } x \\ 0, \text{otherwise} \end{cases}$$

Since each values must be the nearest neighbor of itself, diagonal entries of $\hat{w}$ are $\frac{1}{k}$. Thus the degrees of freedom of KNN is $\frac{n}{k}$, When n=k, the degree of freedom is 1.

Question 6

First,let's load the required packages,

```
library(ggplot2)
library(R.utils)
library(dplyr)
library(lme4)
library(class)
library(gridExtra)
```

Linear regression:

```
train<-read.table('zip.train')
test<-read.table('zip.test')

#train<-read.table('./data/zip.train')
#test<-read.table('./data/zip.test')

train<-filter(train,train$V1=="2"|train$V1=="3")
test<-filter(test,test$V1=="2"|test$V1=="3")

lr.fit<-lm(V1~., data=train)

lr.train_pred<-predict(lr.fit,train[-1])%>%round()
lr.test_pred<-predict(lr.fit,test[-1])%>%round()

lr.train_error<- 1-sum(lr.train_pred==train$V1)/length(lr.train_pred)
lr.test_error<- 1-sum(lr.test_pred==test$V1)/length(lr.test_pred)

errors<-data.frame(lr.train_error,lr.test_error)
colnames(errors)<-c("train error","test error")
errors
```

```
  train error test error
1 0.005759539 0.04120879
```

KNN:

```
K<-c(1,3,5,7,15)
knn.train_errors<-list()
```

```r
knn.test_errors<-list()
for (k in K){
  knn.train_pred = knn(train[,-1],
                  train[,-1],
                  cl=train[,1],
                  k=k)
  knn.test_pred= knn(train[,-1],
                  test[,-1],
                  cl=train[,1],
                  k=k)
  knn.train_errors<-append(knn.train_errors,1-sum(knn.train_pred==train$V1)/length(knn.tra
  knn.test_errors<-append(knn.test_errors,1-sum(knn.test_pred==test$V1)/length(knn.test_pr

  }

knn.errors <- data.frame(unlist(K),unlist(knn.train_errors),unlist(knn.test_errors))
colnames(knn.errors)<-(c("K","train error","test error"))
knn.errors
```

```
   K train error test error
1  1 0.000000000 0.02472527
2  3 0.005039597 0.03021978
3  5 0.005759539 0.03021978
4  7 0.006479482 0.03296703
5 15 0.009359251 0.03846154
```

Comparison:

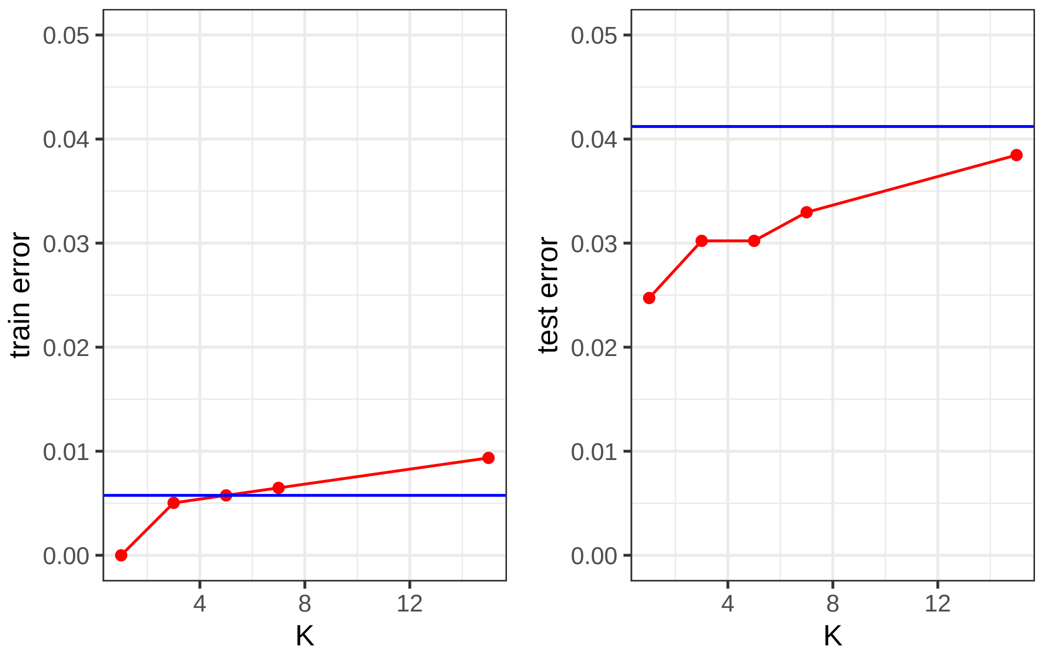
The red line represents the error rates of KNN. The blue line represents the error rate of linear regression.

Overall, the test error rate is higher than the training error rate. The error rate of KNN increases as k increases. The train error of KNN starts lower than the linear regression, and gradually increased to around 0.0094, which become higher than the error rate of linear regression.

For $K \in 1, 3, 5, 7, 15$,The test error of KNN is always lower than that of linear regression.

```r
p1<-ggplot(knn.errors,aes(x=K,y=`train error`))+ geom_point(color="red")+geom_line(color="

p2<-ggplot(knn.errors,aes(x=K,y=`test error`))+ geom_point(color="red")+geom_line(color="r
 # geom_point(knn.errors,aes(x=K,y=`train error`))
```

```r
grid.arrange(p1, p2, ncol=2)
```


train error
K
test error
K