STAT 983 **Homework 01** **Ramesh A. Kesawan**

Problem 1

1. Stein's Unbiased Risk Estimates (SURE): Suppose $Y \sim N(\beta, I_p)$, where $Y, \beta \in R^p, I_p$ is a p-dimensional identity matrix. Consider the estimator $\hat{\beta} = Y + g(Y)$, where $g(Y) = (g_1(Y), \cdots, g_p(Y))$ is weakly differentiable and $\sum_{i=1}^{p} \int \left| \frac{\partial}{\partial y_i} g_i(Y) \right| dY < \infty$ Prove,

$$E\left(\|\beta - \hat{\beta}\|^2\right) = E\left(p + 2\sum_{i=1}^{p} \frac{\partial}{\partial y_i} g_i(Y) + \|g(Y)\|^2\right)$$

Proof:

Consider the $E\left(\|\beta - \hat{\beta}\|^2\right)$ and let $\beta = (\beta_1, \beta_2, ..., \beta_p)$, $Y = (Y_1, Y_2, ..., Y_p)$

$$\begin{aligned}
E\left(\|\beta - \hat{\beta}\|^2\right) &= E\left(\|\beta - Y - g(Y)\|^2\right) \\
&= E\left[\sum_{i=1}^{p} \left(Y_i - \beta_i - g_i(Y)\right)^2\right] \\
&= E\left[\sum_{i=1}^{p} \left(Y_i - \beta_i\right)^2 + \sum_{i=1}^{p} g_i^2(Y) - 2\sum_{i=1}^{p} g_i(Y)\left(Y_i - \beta_i\right)\right] \\
&= E\left[\sum_{i=1}^{p} \left(Y_i - \beta_i\right)^2\right] + E\left[\sum_{i=1}^{p} g_i^2(Y)\right] - 2E\left[\sum_{i=1}^{p} g_i(Y)\left(Y_i - \beta_i\right)\right] \\
&= p + E\|g(Y)\|^2 - 2E[g(Y)]'(Y - \beta) \qquad \because E\left[\sum_{i=1}^{p} \left(Y_i - \beta_i\right)^2\right] = p \text{ ; from variance} \\
&= p + E\|g(Y)\|^2 - 2E\left[\sum_{i=1}^{p} \frac{\partial}{\partial Y_i} g_i(Y)\right] \qquad \because E[g(Y)]'(Y - \beta) = E\left[\sum_{i=1}^{p} \frac{\partial}{\partial Y_i} g_i(Y)\right] \text{ ; From Stein's identity} \\
&= E\left(p - 2\sum_{i=1}^{p} \frac{\partial}{\partial y_i} g_i(Y) + \|g(Y)\|^2\right)
\end{aligned}$$

2. Let $\hat{\theta}^{JS} = \left(1 - \frac{p-2}{\|Y\|^2}\right) Y$ be the JS estimator. $\hat{\theta}^{MLE} = Y$ be the MLE estimator. Prove that when $p \geq 3$

$$E\left(\left\|\beta - \hat{\beta}^{JS}\right\|^2\right) < E\left(\left\|\beta - \hat{\beta}^{MLE}\right\|^2\right)$$

Proof:

Note that $E\left(\left\|\beta - \hat{\beta}^{MLE}\right\|^2\right) = E\left(\|\beta - Y\|^2\right) = p$; from variance. From earlier result,

$$\begin{aligned}
E\left(\|\beta - \hat{\beta}^{JS}\|^2\right) &= E\left(p + \|g(Y)\|^2 - 2\sum_{i=1}^{p} \frac{\partial}{\partial y_i} g_i(Y)\right) \\
&= p + E\left(\|g(Y)\|^2\right) - 2E\left(\sum_{i=1}^{p} \frac{\partial}{\partial y_i} g_i(Y)\right)
\end{aligned}$$

$$
\begin{aligned}
&= p + E\left(\frac{(p-2)^2}{\|Y\|^2}\right) - 2E\left(\sum_{i=1}^{p}\frac{\partial}{\partial y_i}\left[\frac{(p-2)Y_i}{\|Y\|^2}\right]\right) \\
&= p + (p-2)^2 E\left(\frac{1}{\|Y\|^2}\right) - 2(p-2)E\left(\sum_{i=1}^{p}\left[\frac{\|Y\|^2 - 2Y_i^2}{\|Y\|^4}\right]\right) \\
&= p + (p-2)^2 E\left(\frac{1}{(\|Y\|^2}\right) - 2(p-2)^2 E\left(\frac{1}{\|Y\|^2}\right) \qquad \because \sum_{i=1}^{p} Y_i^2 = \|Y\|^2 \\
&= p - (p-2)^2 E\left(\frac{1}{(\|Y\|^2}\right) \\
&= E\left(\left\|\beta - \hat{\beta}^{MLE}\right\|^2\right) - (p-2)^2 E\left(\frac{1}{(\|Y\|^2}\right) \\
&< E\left(\left\|\beta - \hat{\beta}^{MLE}\right\|^2\right) \qquad \because -(p-2)^2 E\left(\frac{1}{(\|Y\|^2}\right) < 0, \quad \forall \beta \in R^P
\end{aligned}
$$

Problem 2 (R exercise)

Here we are supposed to build a Handwritten Text Recognition model to identify the handwritten digits which were scanned from envelopes by the U.S. Postal Service.

Data

The Data set consists of handwritten digits which are automatically scanned from the envelopes by the U.S. Postal Service. The images said to be deslanted and size normalized, resulting in 16 x 16 gray-scale images. Originally that there are 7291 training observations and 2007 test observations, eaco of them is a gray-scale images for digits 0-9. But for our case we have obtained subset of data and narrowed our task to classify the digits 2 and 7. As such our training set makes of 1376 records and 256 variables (16×16) and also another variable indicating class (2 or 7) of the observations.

Explanatory Data Analysis

So, in the data we have 256 variables each of them represents the gray-scale value of a pixel under the resolution of 16×16. For example figure 1 is the 16×16 gray-scale images of 10 randomly selected observations with the original label given below.

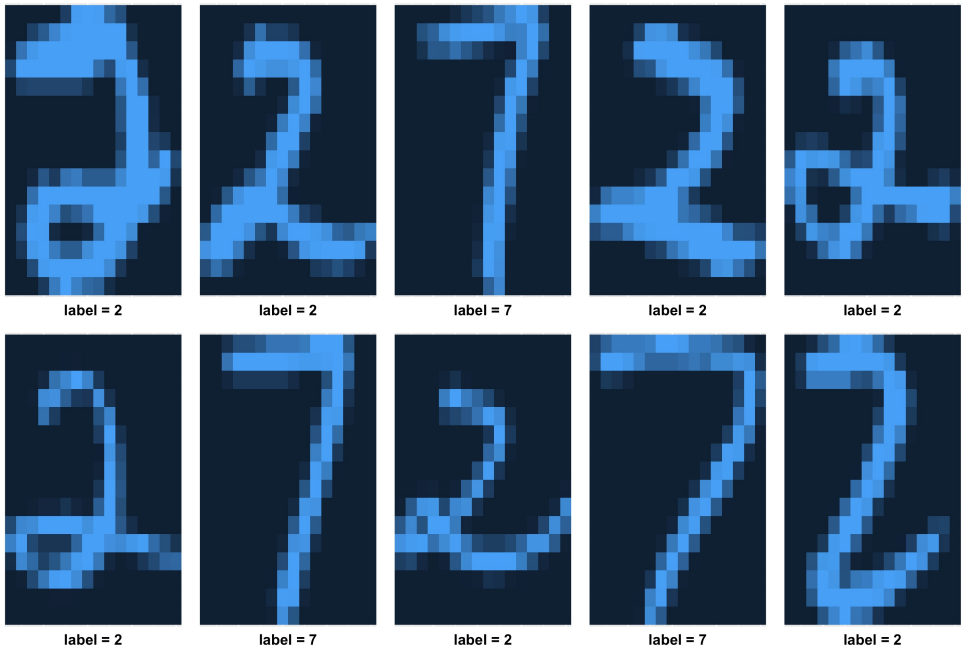
As we said earlier the data also consist of a variable which indicates the labels of the records (class variable) as either 2 or 7. The data set almost balanced in-terms of each classes, with **731** observations from class 2 and **645** from class 7.

Models

Here we discuss two models to classify the above observations to respective classes. One such is a parametric model and the other is a non-parametric model.

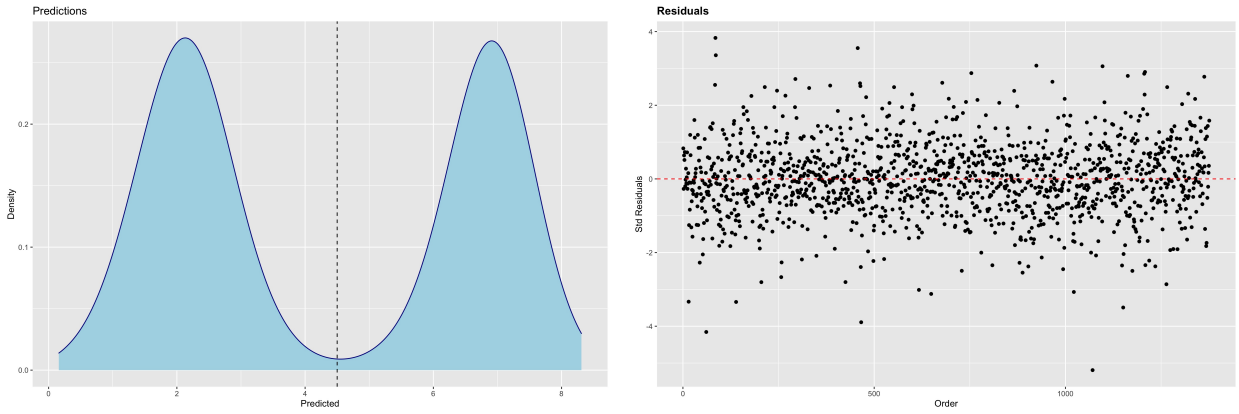
Linear Regression: Linear regression is defining a linear relationship between a scalar response and one or more predictor variables (also known as dependent and independent variables). In our model each of the pixel will be considered as the predictor variable where as the class variable will be the response variable. Having decided the predictor and response variables the model has been built on R with the aid of training set. Using the model we made the in-sample predictions. It is worth to note that LR predicts continuous output. There fore we came up with the decision rule with the help of empirically chosen threshold from

Figure 1: Gray-scale images of 10 observations



the predictions to classify the observations. If you see the below density plot, it is evident that a threshold of 4.5 would divide the predictions into two categories perfectly.

$$\hat{y} = \begin{cases} 2 & P(y|x) < 4.5 \\ 7 & \text{o.w} \end{cases}$$



(a) Distribution of predictions

(b) Residuals

Figure 2: Plots for Linear Regression

To measure the performance of the model, we initially got the value of coefficient of determination (adjusted R^2). That is **0.9293** and that implies the model's ability to explain the variation in the response values. Then we calculated the training mis-classification error. It is **0.000727**. While there are several ways

to check the adequacy of the model, we have used the simplest of all, that is to check the existence of non-random pattern in the residual. From the plot given above, it is evident that such a pattern is absent.

K Nearest Neighbors (KNN): The k-nearest neighbors algorithm (KNN) is a non-parametric classification method that classify the instances by assigning the label, which is most frequent among the k training samples nearest to that instance. With that idea we trained our model using four separate values of **k=1, 3, 5, 7, 15**. Once trained, the model is used to calculate the miss-classification errors for all k values.

Table 1: Training Miss-Classification errors

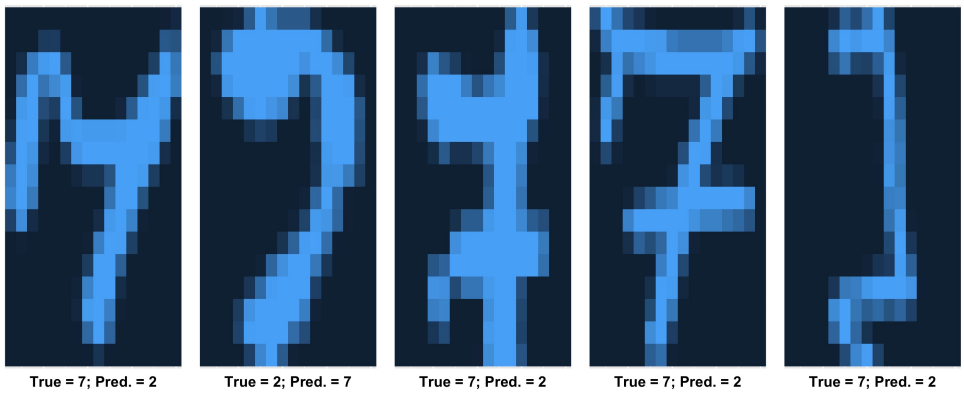
K	Error
1	0
3	0.01017
5	0.01235
7	0.01453
15	0.01744

Even though that you have a perfect miss-classification error when k=1, it is often the case of over-fit. Another important fact is that these errors aren't much differ from one to another. To obtain the best k values it is often recommended to do a cross validation.

Prediction and Conclusion

Linear Regression: The above build model is used to do the prediction on unseen data set(test set). Then miss- classification error is calculated. It turns out the error is **0.01739**.

Figure 3: Miss-Classified LR



. Above plot shows the miss-classified instances linear regression model. Each tile of the plot indicate the true value and the predicted one. Apart from these five instances the model was able to correctly classify the rest of the instances. Which is notably huge considering there are 345 observations.

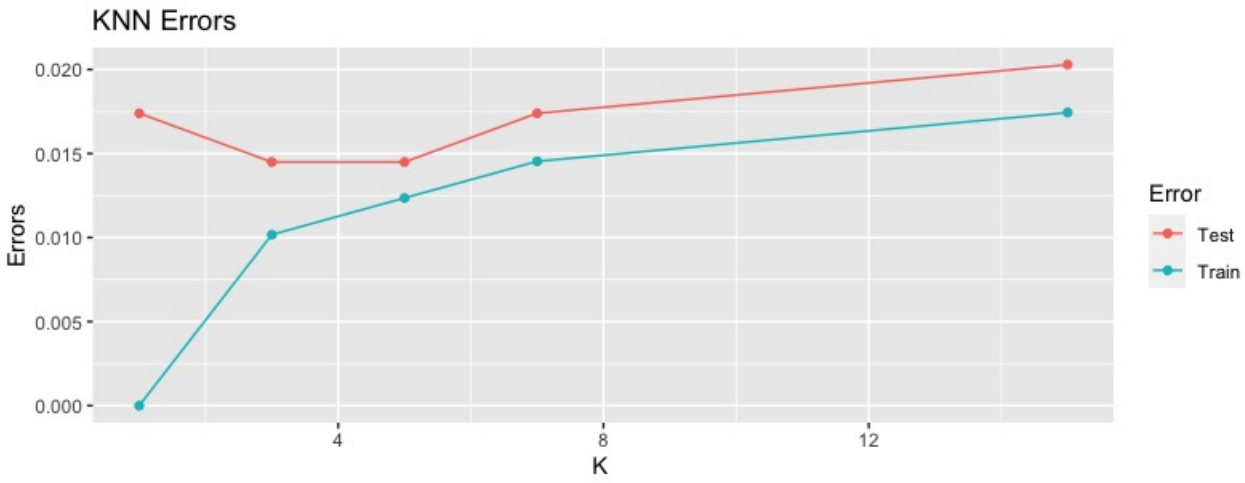
KNN: Same way the KNN is also trained on the train set and found the miss-classification error based on the test set. Below table shows the errors for each value of k.

Table 2: Test Miss-Classification errors

K	Error
1	0.01739
3	0.01449
5	**0.01449**
7	0.01739
15	0.02028

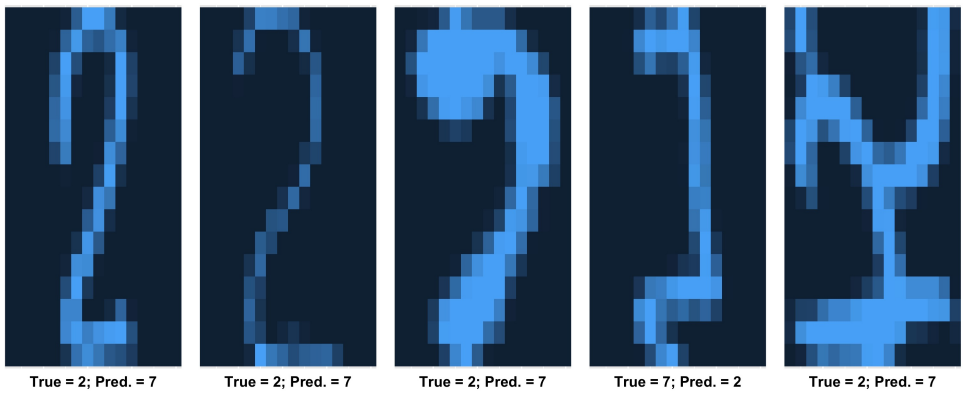
If we observe the values, we can notice that the test miss-classified error is significantly high compared the training error when k=1. It is the clear-cut evidence of over-fitting. To get the optimal value of K, one would plot the test and train errors and find the k which minimizes the both. Plot below given serves the exact purpose. It clear, that when k=5 we can see a reduction in both train and test error. So, our suggestion would be build a model using k=5 for further investigation and sometimes deploying.

Figure 4: KNN test and train errors



Having decided to use k=5, we have shown the only 5 miss-classified images from the test set. labels below each tile indicates the true and predicted labels.

Figure 5: Miss-Classified KNN(5)



Rcode

```
setwd("~/Documents/Class/STAT-983")

library(ggplot2)
library(gridExtra)
library(ggpubr)
library(reshape2)
library(class)

ziptrain <- read.table("zip.train.csv", sep = ",")
ziptrain27 <- subset(ziptrain, ziptrain[,1]==2 | ziptrain[,1]==7)

#### EDA

dim(ziptrain27)
#[1] 1376  257
round(cor(ziptrain27),2)
table(ziptrain27[,1])
#2   7
#731 645


n=10
plots <- vector("list", n)
set.seed(11)
rowIndex<-sample(1:length(ziptrain27),size = n)
for (i in 1:n){
  xVal = t(matrix(data.matrix(ziptrain27[rowIndex[i],-1]),byrow=TRUE,16,16)[16:1,])
  gg <- melt(data.frame(x=1:16,xVal),id="x")
  plots[[i]]<-ggplot(gg) + geom_raster(aes(x=x,y=variable,fill=value))+
    labs(x = paste0("label = ",ziptrain27[rowIndex[i],1]))+
    scale_x_continuous(expand=c(0,0))+
    scale_fill_continuous(type = "gradient")+
    guides(fill=FALSE)+
    theme(axis.text = element_blank(),
          axis.title.x=element_text(face = "bold"),
          axis.ticks=element_blank(),
          axis.title=element_blank())
}
pixelFigure=ggarrange(plotlist=plots,ncol = 5,nrow = 2)
annotate_figure(pixelFigure,top = text_grob("Images training sample",face = "bold",
    size = 16,hjust = 1.5,vjust=1))
ggsave("pixelFigure.jpg",plot = pixelFigure)


#### Models

# linear Regression
mod1 <- lm( V1 ~ . , data= ziptrain27)
regResults<-summary(mod1)
plot(mod1)
plot(regResults$residuals)
```

```
stdResid<-scale(regResults$residuals,center = T,scale = T)
ggplot(data=as.data.frame(stdResid), aes(y=stdResid,x=1:length(stdResid)))+
  geom_point(color="black")+
  labs(x="Order",y="Std Residuals",title = "Residuals")+
  geom_hline(yintercept = 0,linetype="dashed",colour="red")+
  theme(plot.title=element_text(face = "bold"))
ggsave("LRResidual.jpg",plot = last_plot())


pred1.train <- predict.lm(mod1, ziptrain27[,-1])
ggplot(data=as.data.frame(pred1.train), aes(x=pred1.train))+
  geom_density(color="darkblue", fill="lightblue")+
  labs(x="Predicted",y="Density",title = "Predictions")+
  geom_vline(xintercept = 4.5,
             linetype="dashed")
ggsave("LRPredictions.jpg",plot = last_plot())
y1pred.train <- 2 + 5*(pred1.train >= 4.5)
mean(y1pred.train != ziptrain27[,1])


# KNN
kk <- 1
xnew <- ziptrain27[,-1]
ypred2.train <- knn(ziptrain27[,-1], xnew, ziptrain27[,1], k=kk)
mean( ypred2.train != ziptrain27[,1])

trainKnnErrors<-c()
for (i in c(1, 3, 5, 7, 15)){
  xnew <- ziptrain27[,-1]
  ypred2.train <- knn(ziptrain27[,-1], xnew, ziptrain27[,1], k=i)
  trainKnnErrors<-c(trainKnnErrors,mean( ypred2.train != ziptrain27[,1]))
  print(mean( ypred2.train != ziptrain27[,1]))
}
trainKnnErrors
#0.00000000 0.01017442 0.01235465 0.01453488 0.01744186

#### Predictions

## Test Errors
ziptest <- read.table("zip.test.csv",sep = ",")
ziptest27 <- subset(ziptest, ziptest[,1]==2 | ziptest[,1]==7)

# Testing Error of LR
pred1.test <- predict.lm(mod1, ziptest27[,-1])
y1pred.test <- 2 + 5*(pred1.test >= 4.5)
missedLR<-which(y1pred.test != ziptest27[,1])

# Miss-classified plots LR
n=5
plots <- vector("list", n)=
  rowIndex<-missedLR[1:5]
for (i in 1:n){
  xVal = t(matrix(data.matrix(ziptest27[rowIndex[i],-1]),byrow=TRUE,16,16)[16:1,])
```

```
  gg <- melt(data.frame(x=1:16,xVal),id="x")
  plots[[i]]<-ggplot(gg) + geom_raster(aes(x=x,y=variable,fill=value))+
    labs(x = paste0("True = ",ziptest27[rowIndex[i],1],"; Pred. = ",y1pred.test[rowIndex[i]]))+
    scale_x_continuous(expand=c(0,0))+
    scale_fill_continuous(type = "gradient")+
    guides(fill=FALSE)+
    theme(axis.text = element_blank(),
          axis.title.x=element_text(face = "bold"),
          axis.ticks=element_blank(),
          axis.title=element_blank())
}
pixelFigure=ggarrange(plotlist=plots,ncol = 5,nrow = 1)
pixelFigure
ggsave("LR Miss.jpg",plot = pixelFigure)


# Testing error of KNN
testKnnErrors<-c()
for (i in c(1, 3, 5, 7, 15)){
  xnew2 <- ziptest27[,-1]
  ypred2.test <- knn(ziptrain27[,-1], xnew2, ziptrain27[,1], k=i)
  testKnnErrors<-c(testKnnErrors,mean( ypred2.test != ziptest27[,1]))
  print(mean( ypred2.test != ziptest27[,1]))
}

# Test and Train error of KNN
TTErrors<-data.frame(y=c(trainKnnErrors,testKnnErrors),
        x=c(1,3,5,7,15),Error=c(rep("Train",5),rep("Test",5)))
ggplot(data =TTErrors )+
  geom_point(aes(x=x,y=y,color=Error))+
  geom_line(aes(x=x,y=y,color=Error))+
  labs(y="Errors",x="K ",title = "KNN Errors")

ggsave("Knn erors.jpg",plot = last_plot())


xnew5 <- ziptest27[,-1]
ypred5.test <- knn(ziptrain27[,-1], xnew5, ziptrain27[,1], k=5)
print(mean( ypred5.test != ziptest27[,1]))
missedKNN5<-which((ypred5.test != ziptest27[,1]))


# Miss-classified plots KNN5
n=5
plots <- vector("list", n)
rowIndex<-missedKNN5[1:5]
for (i in 1:n){
  xVal = t(matrix(data.matrix(ziptest27[rowIndex[i],-1]),byrow=TRUE,16,16)[16:1,])
  gg <- melt(data.frame(x=1:16,xVal),id="x")
  plots[[i]]<-ggplot(gg) + geom_raster(aes(x=x,y=variable,fill=value))+
    labs(x = paste0("True = ",ziptest27[rowIndex[i],1],"; Pred. = ", ypred5.test[rowIndex[i]]))+
    scale_x_continuous(expand=c(0,0))+
    scale_fill_continuous(type = "gradient")+
```

```
    guides(fill=FALSE)+
    theme(axis.text = element_blank(),
          axis.title.x=element_text(face = "bold"),
          axis.ticks=element_blank(),
          axis.title=element_blank())
}
pixelFigure=ggarrange(plotlist=plots,ncol = 5,nrow = 1)
pixelFigure
ggsave("KNN5 Miss.jpg",plot = pixelFigure)
```