Jakob Lovato
Will Bliss
STAT 760
HW 1

1. We want to show that

$$\text{argmax}_i \hat{y}_i = \text{argmin}_k \|t_k - \hat{y}\| = \text{argmin}_k \|t_k - \hat{y}\|^2.$$

Since $\sum \hat{y}_i = 1$, it follows that $\max y_i \geq \frac{1}{K}$. Then, when $\text{argmax}_i \hat{y}_i \neq \text{argmax}_i \hat{y}_i'$, it follows that

$$\begin{aligned}
\|y - t_{k'}\|^2 - \|y - t_k\| &= y_k^2 + (y_{k'} - 1)^2 - \left(y_{k'}^2 + (y_k - 1)^2\right) \\
&= y_k^2 + y_{k'}^2 - 2y_{k'} + 1 - \left(y_{k'}^2 + y_k^2 - 2y_k + 1\right) \\
&= y_k^2 + y_{k'}^2 - 2y_{k'} + 1 - y_{k'}^2 - y_k^2 + 2y_k - 1 \\
&= 2y_k - 2y_{k'} \\
&= 2(y_k - y_{k'}) \\
&\geq 0
\end{aligned}$$

since $y_k \geq y_{k'}$. Thus,

$$\text{argmax}_i \hat{y}_i = \text{argmin}_k \|t_k - \hat{y}\|.$$

2. Consider N data points uniformly distributed: $X_1, X_2, X_3, \ldots, X_N$. Let d be the median distance to the closest data point from the origin. Let Z be the event of a randomly chosen data point having a distance to the origin larger than d. Then

$$\begin{aligned} P(Z) &= P(\|X_1\| > d) \cdot P(\|X_2\| > d) \cdot \ldots \cdot P(\|X_N\| > d) \\ &= \prod_{i=1}^{N} P(\|X_i\| > d) \\ &= \frac{1}{2}. \end{aligned}$$

Observe $\exists$ a constant value C_p where the volume contained in a p-dimensional sphere of radius r is

$$V(r) = C_p r^p.$$

Then

$$\begin{aligned} P(\|X_i\| > d) &= 1 - P(\|X_i\| \leq d) \\ &= 1 - \frac{C_p d^p}{C_p} \\ &= 1 - d^p \end{aligned}$$

using the volume's ratio. Thus, it follows that

$$\prod_{i=1}^{N} (1 - d^p) = \frac{1}{2}$$

$$\Rightarrow \quad (1-d^p)^N = \frac{1}{2}$$

$$\Rightarrow \quad 1 - d^p = \left(\frac{1}{2}\right)^{\frac{1}{N}}$$

$$\Rightarrow \quad d^p = 1 - \left(\frac{1}{2}\right)^{\frac{1}{N}}$$

$$\Rightarrow \quad d = \left(1 - \left(\frac{1}{2}\right)^{\frac{1}{N}}\right)^{\frac{1}{p}}.$$

Stat 760 Homework 1 Question 3

Will Bliss and Jakob Lovato

2/1/2022

Introduction

The dataset `zipcode` contains rows of data with 256 values per row, each value representing the darkness of a pixel in a 16x16 image of handwritten digits collected from the US Postal Service. We use this data to train classifiers to identify digits in a testing dataset. For the purposes of this code, we only classify the digits 2 and 3.

Code

```
#Read in Data
digitTrain <- read.table("/Users/jakoblovato/Desktop/Stat 760/HW 1/zip.train 2")
digitTrainTwoThree <- digitTrain[digitTrain[,1] == 2 | digitTrain[,1] == 3,]

digitTest <- read.table("/Users/jakoblovato/Desktop/Stat 760/HW 1/zip.test 2")
digitTestTwoThree <- digitTest[digitTest[,1] == 2 | digitTest[,1] == 3,]
```

```
###MANUAL Linear regression
betaZeroTrain <- matrix(1, ncol = 1, nrow(digitTrainTwoThree))
XTrain <- as.matrix(cbind(betaZeroTrain, digitTrainTwoThree[,2:257]))
XTXInverse <- solve(t(XTrain) %*% XTrain)
###digitTrainTwoThree$V1 is Y vector, would not let me assign it to a variable
betas <- XTXInverse %*% t(XTrain) %*% digitTrainTwoThree$V1

YHatTrain <- XTrain %*% betas
trainPredict <- ifelse(YHatTrain < 2.5, 2, 3)

#Compute training error
compareTrain <- data.frame(pred = trainPredict, actual = digitTrainTwoThree[,1])
errorTrain <- mean(compareTrain$actual != compareTrain$pred)
errorTrain
```

```
## [1] 0.005759539
```

```
#Repeat for Test data
betaZeroTest <- matrix(1, ncol = 1, nrow(digitTestTwoThree))
XTest <- as.matrix(cbind(betaZeroTest, digitTestTwoThree[,2:257]))

YHatTest <- XTest %*% betas
testPredict <- ifelse(YHatTest < 2.5, 2, 3)

compareTest <- data.frame(pred = testPredict, actual = digitTestTwoThree[,1])
errorTest <- mean(compareTest$actual != compareTest$pred)
```

```
errorTest
```

```
## [1] 0.04120879
```

We can see the error for the training data using linear regression is 0.0057595 and the error for the test data is 0.0412088. The training error is expectedly much lower than the training error.

```
###KNN Classifier
KNN <- function(training, classifying){
  A <- as.matrix(training[,-1])
  knn1 <- c()
  knn3 <- c()
  knn5 <- c()
  knn7 <- c()
  knn15 <-c()
  for(i in 1:nrow(classifying)){
    X <- t(matrix(unlist(replicate(nrow(A), classifying[i,2:257])), nrow = ncol(A)))
    distances <- diag((A - X) %*% t(A - X))
    knn1Indices <- which(distances %in% sort(distances)[1:1])
    knn1 <- c(knn1, ifelse(sum(digitTrainTwoThree[knn1Indices, 1]) / 1 < 2.5, 2, 3))
    knn3Indices <- which(distances %in% sort(distances)[1:3])
    knn3 <- c(knn3, ifelse(sum(digitTrainTwoThree[knn3Indices, 1]) / 3 < 2.5, 2, 3))
    knn5Indices <- which(distances %in% sort(distances)[1:5])
    knn5 <- c(knn5, ifelse(sum(digitTrainTwoThree[knn5Indices, 1]) / 5 < 2.5, 2, 3))
    knn7Indices <- which(distances %in% sort(distances)[1:7])
    knn7 <- c(knn7, ifelse(sum(digitTrainTwoThree[knn7Indices, 1]) / 7 < 2.5, 2, 3))
    knn15Indices <- which(distances %in% sort(distances)[1:15])
    knn15 <- c(knn15, ifelse(sum(digitTrainTwoThree[knn15Indices, 1]) / 15 < 2.5, 2, 3))
  }
  return(data.frame(knn1 = knn1, knn3 = knn3, knn5 = knn5, knn7 = knn7, knn15 = knn15))
}
```

```
#KNN on training data
knnTrain <- KNN(digitTrainTwoThree, digitTrainTwoThree)

knnTrainError <- matrix(c(mean(knnTrain$knn1 != digitTrainTwoThree[,1]),
                          mean(knnTrain$knn3 != digitTrainTwoThree[,1]),
                          mean(knnTrain$knn5 != digitTrainTwoThree[,1]),
                          mean(knnTrain$knn7 != digitTrainTwoThree[,1]),
                          mean(knnTrain$knn15 != digitTrainTwoThree[,1])), nrow = 1)
colnames(knnTrainError) <- c("1NN Error", "3NN Error", "5NN Error", "7NN Error",
                             "15NN Error")
knnTrainError
```

```
     1NN Error   3NN Error   5NN Error   7NN Error  15NN Error
[1,]         0 0.005039597 0.005759539 0.006479482 0.009359251
```

As one would expect, the 1NN error on training data is 0, as all data points align with themselves. The training error for 3NN is 0.005039597, error for 5NN is 0.005759539, error for 7NN is 0.006479482, and error for 15NN is 0.009359251.
Note that the errors for KNN on the training data is very close to the error for linear regression on the training data; in fact, the errors are equal for linear regression and 5NN. So the two classification methods appear comparable on training data.

```
#KNN on test data
knnTest <- KNN(digitTrainTwoThree, digitTestTwoThree)

knnTestError <- matrix(c(mean(knnTest$knn1 != digitTestTwoThree[,1]),
                         mean(knnTest$knn3 != digitTestTwoThree[,1]),
                         mean(knnTest$knn5 != digitTestTwoThree[,1]),
                         mean(knnTest$knn7 != digitTestTwoThree[,1]),
                         mean(knnTest$knn15 != digitTestTwoThree[,1])), nrow = 1)
colnames(knnTestError) <- c("1NN Error", "3NN Error", "5NN Error", "7NN Error",
                            "15NN Error")
knnTestError
```

```
      1NN Error  3NN Error  5NN Error  7NN Error 15NN Error
[1,] 0.02472527 0.03021978 0.03021978 0.03296703 0.03846154
```

The errors for the test data is notably larger than that of the training data, but still quite low. The training error for 1NN is 0.02472527, error for 3NN is 0.03021978, error for 5NN is 0.03021978, error for 7NN is 0.03296703, and error for 15NN is 0.03846154.
We see that the error for all KNN models on test data are lower than the error for linear regression on test data. It is notable that when we consider more nearest neighbors, the error increases. So 1NN has the lowest error and has about half the error of linear regression. 1NN having the lowest error surprised us, as one might expect this model to be over fit to the training data. See the confusion matrices below for comparison of correct/false classification for each of the KNN models:

```
#KNN1 Confusion Matrix
knn1confusion <- matrix(c(sum(which(knnTest$knn1 == 2) %in% which(digitTestTwoThree[,1] == 2)),
                      sum(which(knnTest$knn1 == 2) %in% which(digitTestTwoThree[,1] == 3)),
                      sum(which(knnTest$knn1 == 3) %in% which(digitTestTwoThree[,1] == 2)),
                      sum(which(knnTest$knn1 == 3) %in% which(digitTestTwoThree[,1] == 3))
                      ), nrow = 2, byrow = TRUE)
colnames(knn1confusion) <- c("True 2", "True 3")
rownames(knn1confusion) <- c("Pred 2", "Pred 3")
knn1confusion
```

```
       True 2 True 3
Pred 2    192      3
Pred 3      6    163
```

```
#KNN3ConfustionMatrix
knn3confusion <- matrix(c(sum(which(knnTest$knn3 == 2) %in% which(digitTestTwoThree[,1] == 2)),
                          sum(which(knnTest$knn3 == 2) %in% which(digitTestTwoThree[,1] == 3)),
                          sum(which(knnTest$knn3 == 3) %in% which(digitTestTwoThree[,1] == 2)),
                          sum(which(knnTest$knn3 == 3) %in% which(digitTestTwoThree[,1] == 3))
), nrow = 2, byrow = TRUE)
colnames(knn3confusion) <- c("True 2", "True 3")
rownames(knn3confusion) <- c("Pred 2", "Pred 3")
knn3confusion
```

```
       True 2 True 3
Pred 2    191      4
Pred 3      7    162
```

```
#KNN5ConfustionMatrix
knn5confusion <- matrix(c(sum(which(knnTest$knn5 == 2) %in% which(digitTestTwoThree[,1] == 2)),
                          sum(which(knnTest$knn5 == 2) %in% which(digitTestTwoThree[,1] == 3)),
                          sum(which(knnTest$knn5 == 3) %in% which(digitTestTwoThree[,1] == 2)),
```

```
                          sum(which(knnTest$knn5 == 3) %in% which(digitTestTwoThree[,1] == 3))
), nrow = 2, byrow = TRUE)
colnames(knn5confusion) <- c("True 2", "True 3")
rownames(knn5confusion) <- c("Pred 2", "Pred 3")
knn5confusion
```

```
       True 2 True 3
Pred 2    191      4
Pred 3      7    162
```

```
#KNN7ConfustionMatrix
knn7confusion <- matrix(c(sum(which(knnTest$knn7 == 2) %in% which(digitTestTwoThree[,1] == 2)),
                          sum(which(knnTest$knn7 == 2) %in% which(digitTestTwoThree[,1] == 3)),
                          sum(which(knnTest$knn7 == 3) %in% which(digitTestTwoThree[,1] == 2)),
                          sum(which(knnTest$knn7 == 3) %in% which(digitTestTwoThree[,1] == 3))
), nrow = 2, byrow = TRUE)
colnames(knn7confusion) <- c("True 2", "True 3")
rownames(knn7confusion) <- c("Pred 2", "Pred 3")
knn7confusion
```

```
       True 2 True 3
Pred 2    189      3
Pred 3      9    163
```

```
#KNN15ConfustionMatrix
knn15confusion <- matrix(c(sum(which(knnTest$knn15 == 2) %in% which(digitTestTwoThree[,1] == 2)),
                          sum(which(knnTest$knn15 == 2) %in% which(digitTestTwoThree[,1] == 3)),
                          sum(which(knnTest$knn15 == 3) %in% which(digitTestTwoThree[,1] == 2)),
                          sum(which(knnTest$knn15 == 3) %in% which(digitTestTwoThree[,1] == 3))
), nrow = 2, byrow = TRUE)
colnames(knn15confusion) <- c("True 2", "True 3")
rownames(knn15confusion) <- c("Pred 2", "Pred 3")
knn15confusion
```

```
       True 2 True 3
Pred 2    187      3
Pred 3     11    163
```