Mini Project 1

John Kenney

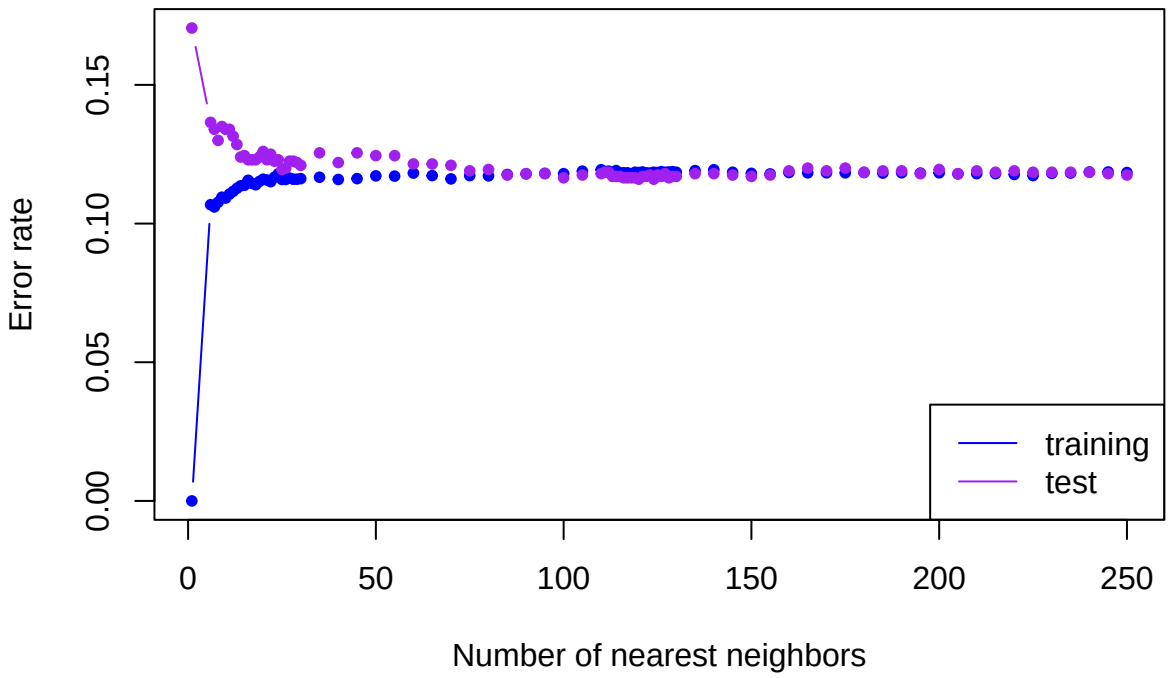
9/8/2021

Section 1

1.

(a)
From the r code shown in section 2 you can see that we model knn with k = 1-200 usually by increments of 5 except in certain areas when we increment by one on the zoom in on areas that we might think will be good to investigate further. Also from this we found k= 120 to have the lowest error for test data on the first run so I investigated this area further on the subsequent runs.

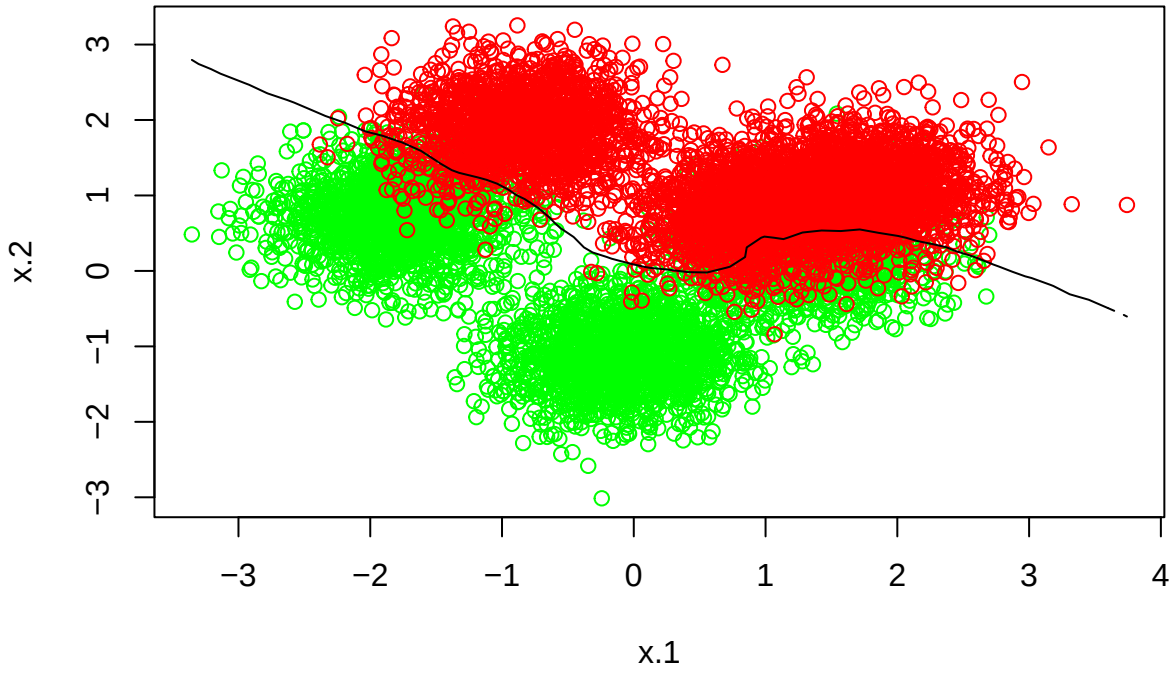
(b) The model is working as it should because we can see that the test error is creating a U shape which will give us a minimum and also agrees with the intuition that with a small k the generalization of the model is bad so the test error will be high. It is consistent with what i would expect.



(c) On fist running through the model I went by increments of 5 from 35 to 200 and after fitting found that k=120 had the lowest test error so I re ran the model the from 110 to 130 going by increments of 1. from This I found that k=124 had the same test error as k=120 but for drawing my decision boundary I stuck with k=120. The optimum k results can be found in the table below.

```
##      ks err.rate.train err.rate.test
## 120 120         0.1184         0.116
## 124 124         0.1185         0.116
```

(d) From what I can see the decision boundary shown below seems reasonable to me because going further away from the boundary we see that most of the points are correctly classified. On the lower right of the decision boundary from x.1 about 1 to 2 and x.2 about 0 to 1 it is not classifying the best I would like but from the overlap it looks that there are points underneath the red that we cant see are correctly classified. bottom line is that nearest to the decision boundary the harder it is the classify variables as can be seen in the plot below.



2 (a) MSE$\{\hat{f}(x_0)\}$ = (Bias$\{\hat{f}(x_0)\}$)^2 + var$\{\hat{f}(x_0)\}$
$E[(\hat{f}(x_0) - f(x_0))^2] = (E[\hat{f}(x_0)] - f(x_0))^2 + E[\hat{f}(x_0)^2]$ - $E[\hat{f}(x_0)]^2$
$E[\hat{f}(x_0)^2 - 2\hat{f}(x_0)f(x_0) + f(x_0)^2] = E[\hat{f}(x_0)]^2$ - $2E[\hat{f}(x_0)]f(x_0) + f(x_0)^2 + E[\hat{f}(x_0)^2]$ - $E[\hat{f}(x_0)]^2$
bc $E[\hat{f}(x_0)]^2$ - $E[\hat{f}(x_0)]^2 = 0$
$E[\hat{f}(x_0)^2]$ - $2E[\hat{f}(x_0)]f(x_0) + f(x_0)^2 = E[\hat{f}(x_0)^2]$ - $2E[\hat{f}(x_0)]f(x_0) + f(x_0)^2$

(b) $E[(\hat{Y_0} - Y_o)^2] = (\text{Bias}\{\hat{f}(x_0)\})^2 + \text{var}\{\hat{f}(x_0)\} + \sigma^2$
from a we know that Bias + Var is $E[(\hat{f}(x_0) - f(x_0))^2]$ = MSE
$E[(\hat{f}(x_0) - f(x_0) - \epsilon_0)^2] = E[(\hat{f}(x_0) - f(x_0))^2] + \sigma^2$

$E[(\hat{f}(x_0) - f(x_0))^2 - 2\epsilon_0(\hat{f}(x_0) - f(x_0)) + \epsilon_0^2] = E[(\hat{f}(x_0) - f(x_0))^2] + \sigma^2$
$E[(\hat{f}(x_0) - f(x_0))^2]$ - $2E[\epsilon_0]E[(\hat{f}(x_0) - f(x_0))] + E[\epsilon_0^2] = E[(\hat{f}(x_0) - f(x_0))^2] + \sigma^2$
since $E[\epsilon_0] = 0$ then $2E[\epsilon_0]E[(\hat{f}(x_0) - f(x_0))]$ because the training error and test error are independent
and $E[(\hat{f}(x_0) - f(x_0))^2] = \text{MSE}$
and lastly $E[\epsilon_0^2] = var(\epsilon_0) + E[\epsilon_0]^2$
$\text{MSE}\{\hat{f}(x_0)\} + var(\epsilon_0) + E[\epsilon_0]^2 = \text{MSE}\{\hat{f}(x_0)\} + \sigma^2$
since we know $var(\epsilon_0) = \sigma^2$
and that $E[\epsilon_0] = 0$ then $E[\epsilon_0]^2 = 0$
therefore
$\text{MSE}\{\hat{f}(x_0)\} + \sigma^2 = \text{MSE}\{\hat{f}(x_0)\} + \sigma^2$

section 2 can be found on the next page

Section 2
rcode

```
# here we load in the train and test data
library(class) # for knn
library(kableExtra) # for ktable
setwd("C:/Users/John/Documents/R/Programs/Stat4360/MiniProjects/Miniproject1/")
train_data <- read.csv("1-training_data.csv")
test_data <- read.csv("1-test_data.csv")


#plots the training data and colors the points that contain yes for
 #the response variable as green and red for no
plot(train_data[,1:2], xlab = "x.1", ylab = "x.2",
     col = ifelse(train_data[,3] == "yes", "green", "red"))

#<!-- # Fit KNN for several values of K -->
#creates a sequence 1 to 200 by 5 and a zoomed in look at
#around 120 by looking 10 places before and after to by increments
#of 1 to see if there is a better minimum
ks <- c(1,seq(6, 30, by = 1), seq(35, 105, by = 5),seq(110, 130, by = 1), seq(135, 250, by = 5))
nks <- length(ks)
err.rate.train <- numeric(length = nks)
err.rate.test <- numeric(length = nks)
names(err.rate.train) <- names(err.rate.test) <- ks
#creates vectors to store the errors for each k
for (i in seq(along = ks)) { # creates a model for each k
  #specified in ks and saves the error rate for the model
  #when predicting on the training set and test set
  set.seed(1)
  mod.train <- knn(train_data[,1:2], train_data[,1:2], train_data[,3], k = ks[i])
  set.seed(1)
  mod.test <- knn(train_data[,1:2], test_data[,1:2], train_data[,3], k = ks[i])
  err.rate.train[i] <- mean(mod.train != train_data[,3])
  err.rate.test[i] <- mean(mod.test != test_data[,3])
}



#creates a data frame of the k used and its error on training and test datasets
result <- data.frame(ks, err.rate.train, err.rate.test)
kbl(result, booktabs = T,
    caption = "Table of the error rate for each k on the train and test data") %>%
    kable_styling(latex_options = c("striped", "scale_down"))

# plots the error rate of each k on the training and test set
plot(ks, err.rate.train, xlab = "Number of nearest neighbors", ylab = "Error rate", type = "b",
     ylim = range(c(err.rate.train, err.rate.test)), col = "blue", pch = 20)
lines(ks, err.rate.test, type="b", col="purple", pch = 20)
legend("bottomright", lty = 1, col = c("blue", "purple"), legend = c("training", "test"))

# gets the minimum error found of the k's on the error test rate
result[err.rate.test == min(result$err.rate.test), ]
```

```r
# Setting up for creating a Decision boundary for optimal K
# creates a grid of that covers all the values of x.1 and x.2
n.grid <- 50
x1.grid <- seq(f = min(train_data[, 1]), t = max(train_data[, 1]), l = n.grid)
x2.grid <- seq(f = min(train_data[, 2]), t = max(train_data[, 2]), l = n.grid)
grid <- expand.grid(x1.grid, x2.grid)

# saves the k that has the minimum error on the test data set i got 2 values
#of the same min error but just picked 120 to use for making the decision boundary
k.opt <- result[err.rate.test == min(result$err.rate.test), ]$ks[1]
set.seed(1)
# trains the model using the optimum k and saves the proportion of the winning class
mod.opt <- knn(train_data[,1:2], grid, train_data[,3], k = k.opt, prob = T)
prob <- attr(mod.opt, "prob") # prob is voting fraction for winning class
prob <- ifelse(mod.opt == "yes", prob, 1 - prob) # now it is voting fraction for Direction == "yes"
prob <- matrix(prob, n.grid, n.grid)

#plots the training data while applying the decision boundary on top of the plot
plot(train_data[,1:2], col = ifelse(train_data[,3] == "yes", "green", "red"))
contour(x1.grid, x2.grid, prob, levels = 0.5, labels = "", xlab = "", ylab = "", main = "", add = T)
```