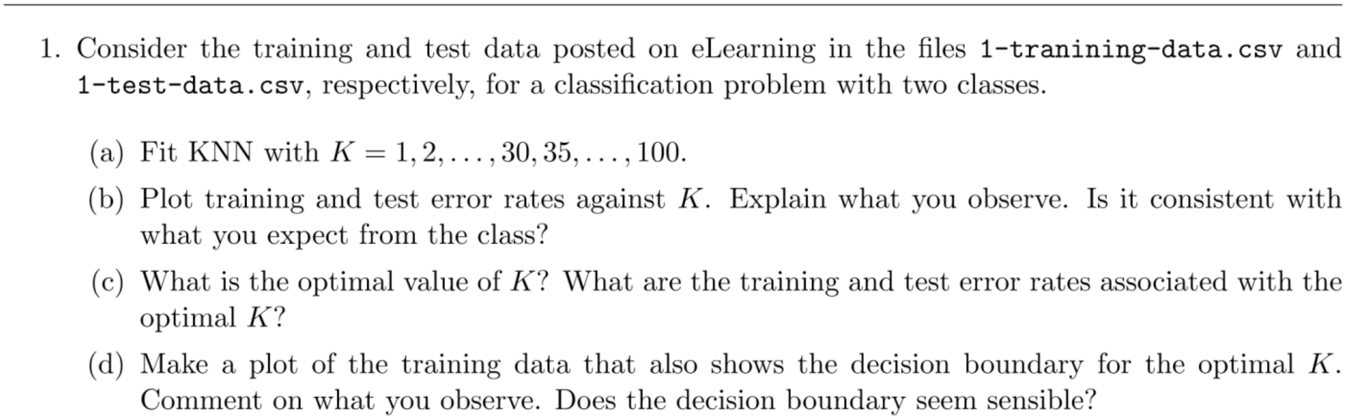


1. Consider the training and test data posted on eLearning in the files `1-tranining-data.csv` and `1-test-data.csv`, respectively, for a classification problem with two classes.

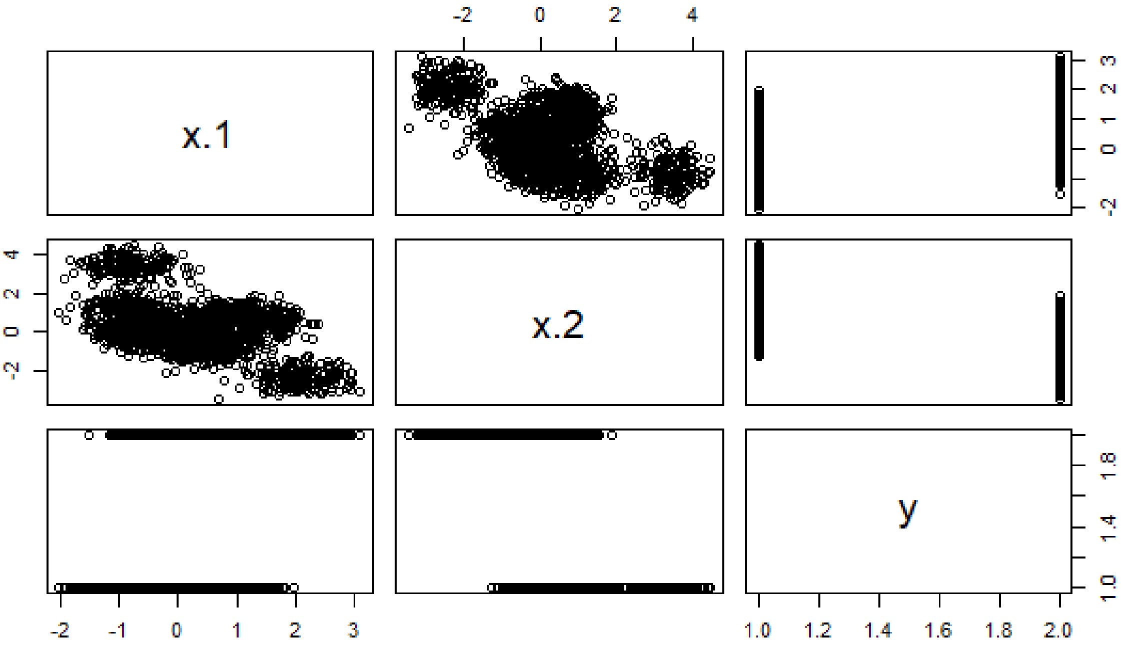
   (a) Fit KNN with $K = 1, 2, \ldots, 30, 35, \ldots, 100$.

   (b) Plot training and test error rates against K. Explain what you observe. Is it consistent with what you expect from the class?

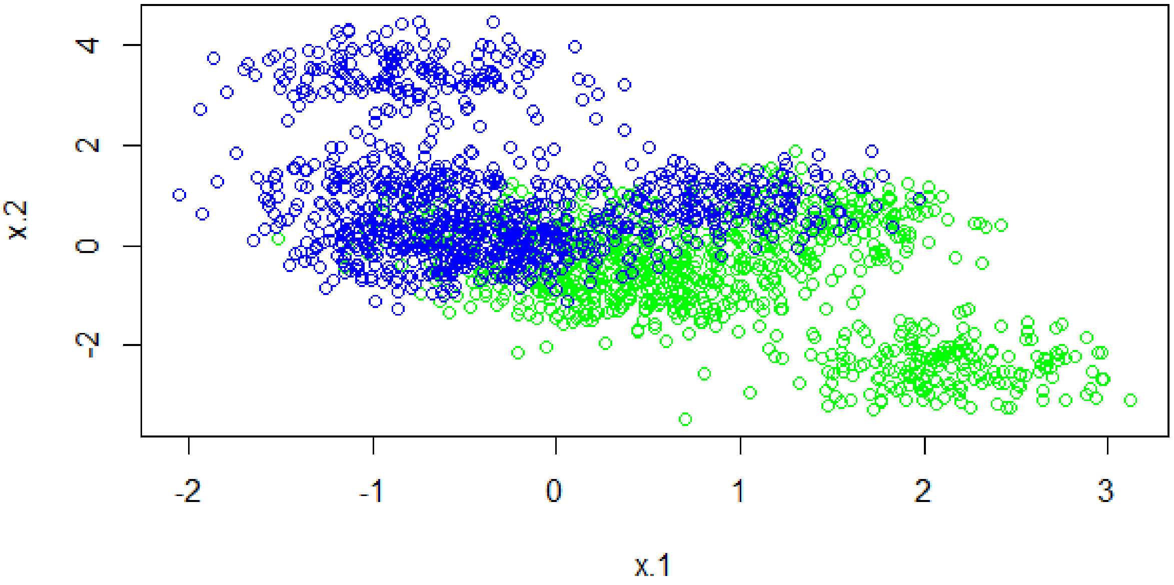
   (c) What is the optimal value of K? What are the training and test error rates associated with the optimal K?

   (d) Make a plot of the training data that also shows the decision boundary for the optimal K. Comment on what you observe. Does the decision boundary seem sensible?

```
#Name: LP Dangal
#Project: 1

Here, we need to separate two variables x1 and x2 using classification algori
thm KNN. All the codes below are for viewing structural, correlations and ove
rview of nature of data. After all the questions are addressed one by one.

> set.seed(170030)
> library(class)

> #downloadin training data and test data
> train_data <- read.csv("C:/Users/LPD/Desktop/Stat ML Pankaj/Project/project
1/1-training_data.csv")
>
> test_data<- read.csv("C:/Users/LPD/Desktop/Stat ML Pankaj/Project/project1/
1-test_data.csv")


> #overview of training and test data
> str(train_data)
'data.frame':   2000 obs. of  3 variables:
 $ x.1: num  1.444 0.373 1.902 1.019 1.981 ...
 $ x.2: num  0.131 -0.878 -2.515 -0.226 -2.894 ...
 $ y  : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...


> str(test_data)
'data.frame':   2000 obs. of  3 variables:
 $ x.1: num  0.802 1.918 0.866 0.647 1.931 ...
 $ x.2: num  -0.518 -2.935 -0.484 -0.879 0.448 ...
 $ y  : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...


> # scatterplot matrix of training data
> pairs(train_data)
```


x.1
x.2
y
-2
0
2
4
-2
-1
0
1
2
3
1.0
1.2
1.4
1.6
1.8
2.0

```
> #directly accesing column names without using $sign
> attach(train_data)


> #correlations between predictors
> cor(train_data[,-3])
           x.1        x.2
x.1  1.0000000 -0.5365476
x.2 -0.5365476  1.0000000


> #combining predictors of training datasets.
> train_data.x = cbind(x.1,x.2)


> # response variable for training datasets
> train_data.y = y


> #viewing top 6 rows of combined predictors
> head(train_data.x)
           x.1        x.2
[1,] 1.4439388  0.1314175
[2,] 0.3726628 -0.8782508
[3,] 1.9018229 -2.5146419
[4,] 1.0193237 -0.2262318
[5,] 1.9805583 -2.8940830
[6,] 1.8975980 -0.1363249


> # viewing response variable for training datasets
> head(train_data.y)
[1] yes yes yes yes yes yes
Levels: no yes

> dim(train_data.x)
[1] 2000    2



> #scatter plots between two predictors
> plot(train_data.x, xlab = "x.1", ylab = "x.2",
+      col = ifelse(train_data.y == "yes", "green", "blue"))
```



```
> #for not using $sign to access column
> attach(test_data)

> test_data.x = cbind(x.1,x.2)
> dim(test_data.x)
[1] 2000    2

> test_data.y = y
>
> head(test_data.y)
[1] yes yes yes yes yes yes
Levels: no yes

> table(test_data.y)
test_data.y
  no  yes
1000 1000
```

a) Fit KNN for several values of K

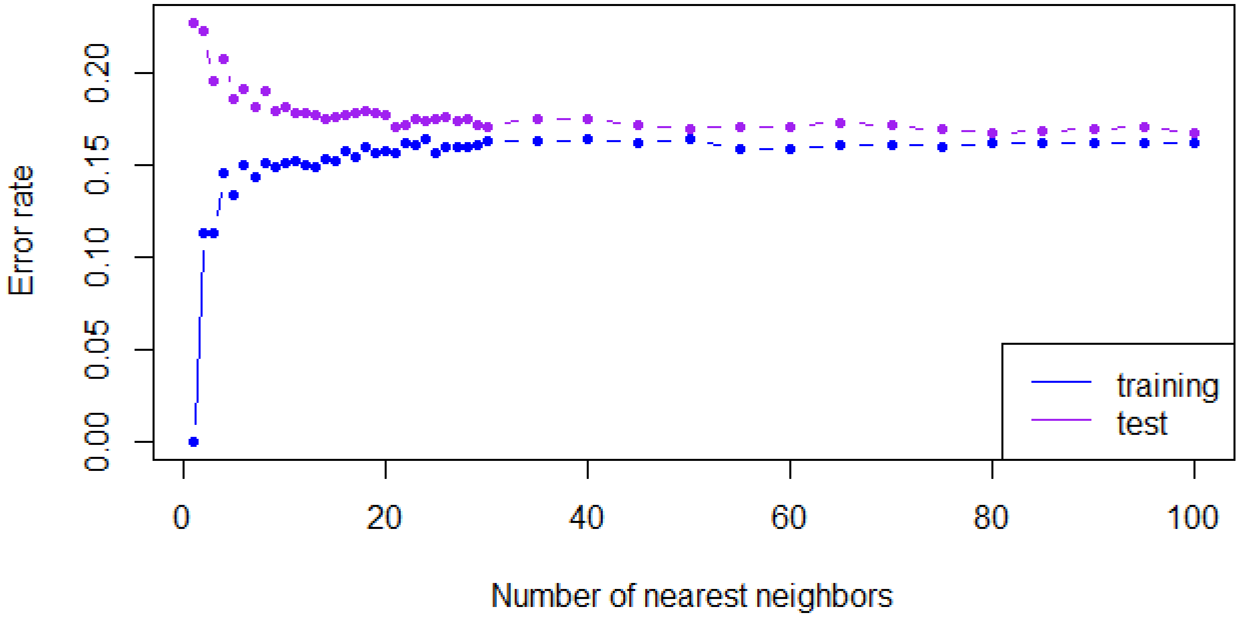
```
> # Fit KNN for several values of K

>  ks <- c(seq(1, 30, by = 1), seq(35, 100, by = 5))


> nks <- length(ks)
> nks
[1] 44
> err.rate.train <- numeric(length = nks)
> err.rate.train
 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
[30] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
> err.rate.test <- numeric(length = nks)
> names(err.rate.train) <- names(err.rate.test) <- ks
>
> for (i in seq(along = ks)) {
+   set.seed(170030)
+   mod.train <- knn(train_data.x, train_data.x, train_data.y, k = ks[i])
+   set.seed(170030)
+   mod.test <- knn(train_data.x, test_data.x, train_data.y, k = ks[i])
+   err.rate.train[i] <- 1 - sum(mod.train == train_data.y)/length(train_data
.y)
+   err.rate.test[i] <- 1 - sum(mod.test == test_data.y)/length(test_data.y)
+ }
```

b) # Codes for plotting train and test error

```
> 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"
```



We know that as K increases model becomes less flexible and finally become inflexible as large K. Also, we know that

Training MSE is very low and test MSE is very high for high flexible model i.e. K=1, graph agrees on it. As the flexibility Decreases i.e. k increases, training MSE goes towards zero and test MSE becomes minimum and increases again with U Shaped curve which is seen in above plot as a flatted u shaped. U shaped curve of test MSE is because of the tradeoff between bias and variance. It is observed that test MSE is minimum at K=80 and K=100. We can choose less flexible models if both have same test MSE.

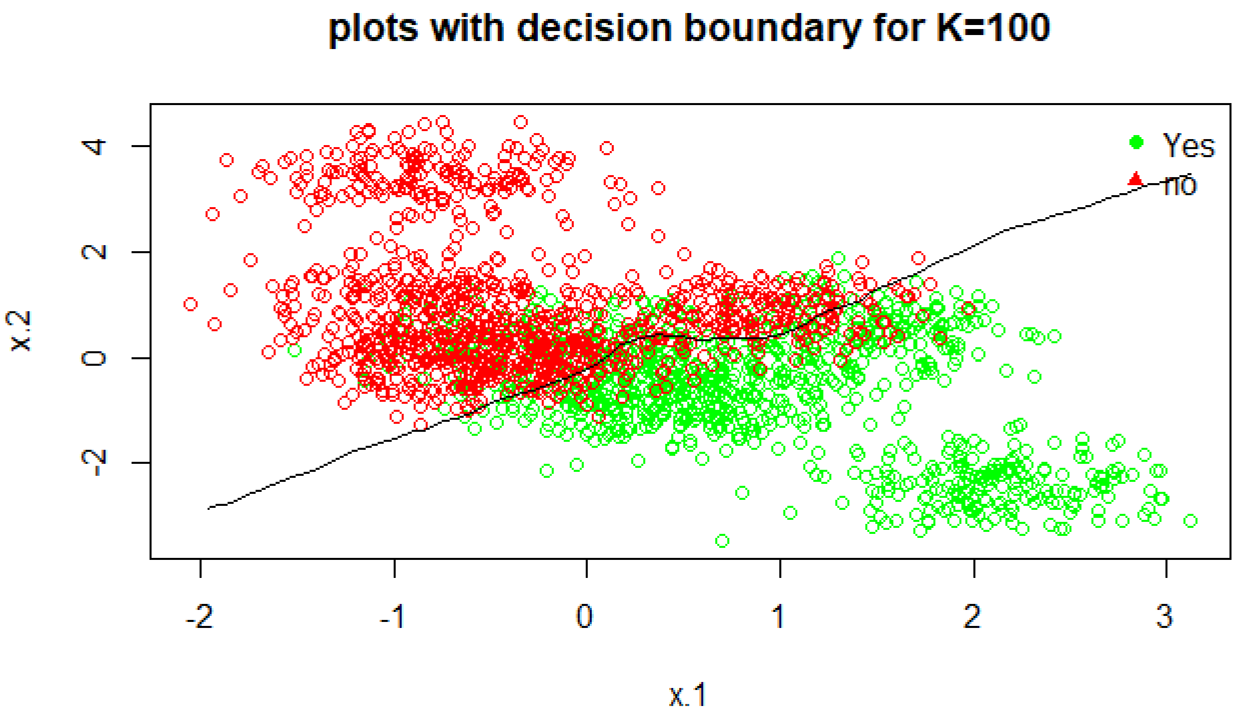
c) #Optimal value of K and training and test error rates associated it.

```
> result <- data.frame(ks, err.rate.train, err.rate.test)
> View(result)
>  result[err.rate.test == min(result$err.rate.test), ]
     ks err.rate.train err.rate.test
80   80          0.162         0.168
100 100          0.162         0.168
```

We can choose anyone as both have same test MSE. It is better to choose less flexible model so lets' pick K=100. So Optimal value of K is 100 and its' training error rate is 0.162 and test error rate is 0.168.

d) Plotting of training data with decision boundary for optimal K=100

```
> # Decision boundary for optimal K (not very informative here)
>
> n.grid <- 50
> x1.grid <- seq(f = min(train_data.x[, 1]), t = max(train_data.x[, 1]), l = n.grid)
> x2.grid <- seq(f = min(train_data.x[, 2]), t = max(train_data.x[, 2]), l = n.grid)
> grid <- expand.grid(x1.grid, x2.grid)
> #minimum test MSE from plot is 100
> k.opt <- 100
> set.seed(170030)
> mod.opt <- knn(train_data.x, grid, train_data.y, 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 Dire
== "yes"
> prob <- matrix(prob, n.grid, n.grid)
>
> plot(train_data.x, main="plots with decision boundary for K=100",col = ifelse(train_c
== "yes", "green", "red"))
>
> contour(x1.grid, x2.grid, prob, levels = 0.5, labels = "", xlab = "", ylab = "",
+         main = "", add = T)

> legend("topright",col=c("green","red"),legend=c("Yes","no"),bty="n",pch=c(16,17))
```



The black curve in circle of red and green showing the decision boundary for green (yes) and red(no). It is observed That some red and some greens are misclassified with test error rate 16.8%.