Machine Learning Assignment 1

Mowniesh Asokan(mowas455)

11/13/2020

Assignment 1 HandWritten Digit Recognition

1.Partitioning the data

```
#Load Data
dataset = read.csv("C:/Users/Mounish/Desktop/ML/Lab1/optdigits.csv")
cat("Dimension of Dataset is",dim(dataset),"\n")
```

```
## Dimension of Dataset is 3822 65
```

```
#Split the data into train, validation and test

set.seed(12345)
sample_train<- sample(seq_len(nrow(dataset)), size = floor(0.50*nrow(dataset)))
sample_valid<- sample(seq_len(nrow(dataset)), size = floor(0.25*nrow(dataset)))
sample_test <- sample(seq_len(nrow(dataset)), size = floor(0.25*nrow(dataset)))

train     <- dataset[sample_train, ]
validation<- dataset[sample_valid, ]
test      <- dataset[sample_test, ]
```

2.Fit the kknn model to the training data

Predict the validation and test

- Model is worked and trained by using the training data .
- In this nearest neighbor method estimate is formed over the average of K nearest data points and also the distance is calculated by using Euclidean distance formula.

1.Confusion Matrix

```
## Confusion matrix for the training data.
```

```
##
##       0   1   2   3   4   5   6   7   8   9
##    0 177   0   0   0   1   0   0   0   0   0
##    1   0 174   9   0   0   0   1   0   1   3
##    2   0   0 170   0   0   0   0   1   2   0
##    3   0   0   0 197   0   2   0   1   0   0
##    4   0   1   0   0 166   0   2   6   2   2
##    5   0   0   0   0   0 183   1   2   0  11
##    6   0   0   0   0   0   0 200   0   0   0
##    7   0   1   0   1   0   1   0 192   0   0
##    8   0  10   0   1   0   0   2   0 190   2
##    9   0   3   0   4   2   0   0   2   4 181
```

```
## Confusion matrix for the test data.
```

```
##
##        0   1   2   3   4   5   6   7   8   9
##    0 177   0   0   0   1   0   0   0   0   0
##    1   0 174   9   0   0   0   1   0   1   3
##    2   0   0 170   0   0   0   0   1   2   0
##    3   0   0   0 197   0   2   0   1   0   0
##    4   0   1   0   0 166   0   2   6   2   2
##    5   0   0   0   0   0 183   1   2   0  11
##    6   0   0   0   0   0   0 200   0   0   0
##    7   0   1   0   1   0   1   0 192   0   0
##    8   0  10   0   1   0   0   2   0 190   2
##    9   0   3   0   4   2   0   0   2   4 181
```

- In confusion matrix, rows are taken as actual Y and the columns are taken as predicted $\bar{Y}$

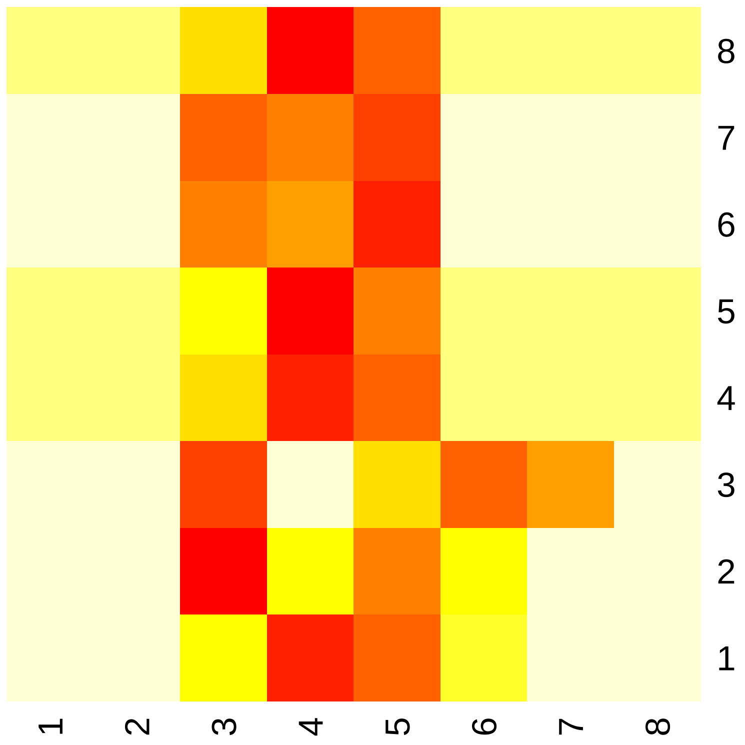
2.Missclassification Rate

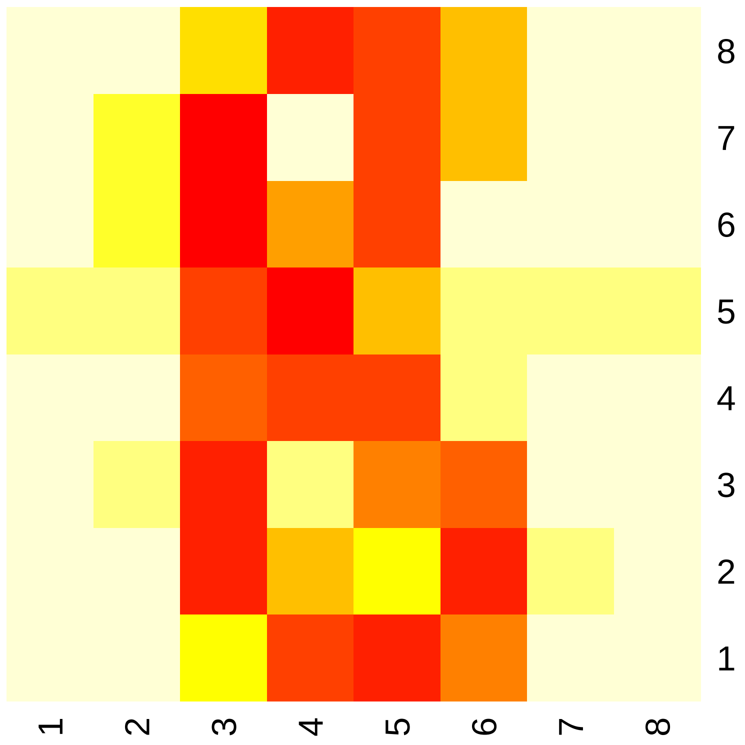
```
## [1] 0.04238619
```

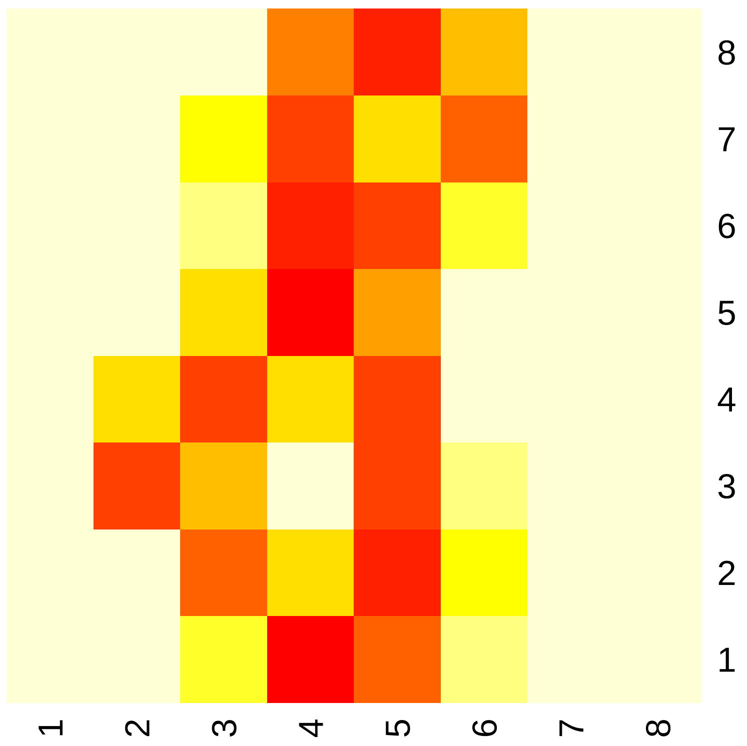
```
## [1] 0.05968586
```

According to value We get through the confusion matrix, the miss classification of actual 3 and 4 are high in rate and somewhat 9 is predicted wrongly as 5 and that the miss classification rate is affected by test and validation data.

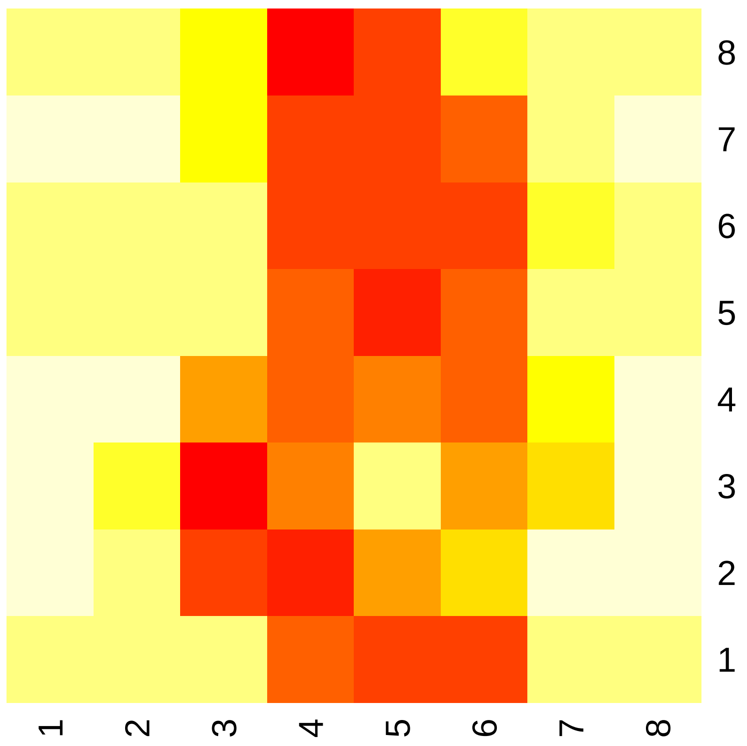
2.Identify any 2 cases of digit "8" in the training data which were easiest to classify and 3 cases that were hardest to classify

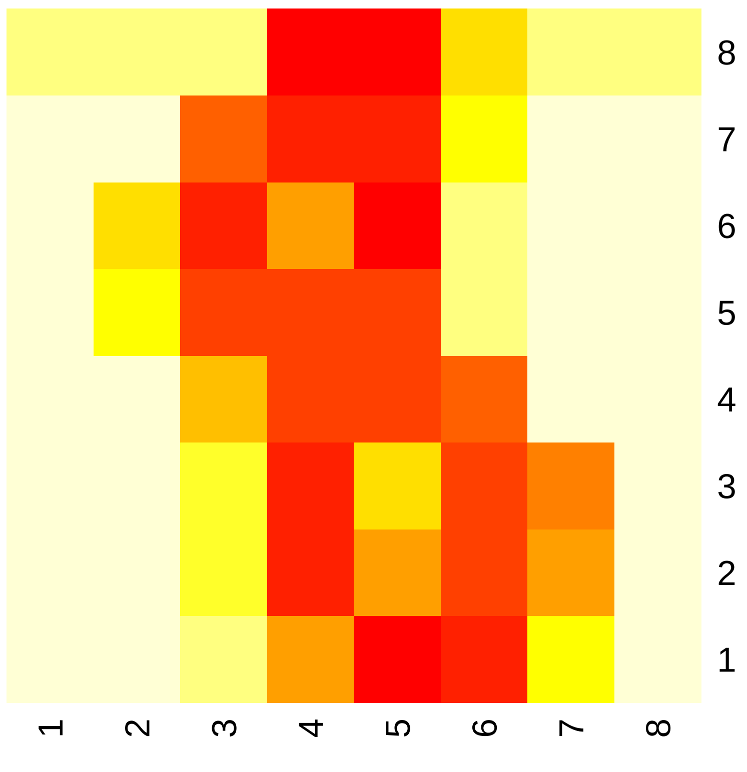






1
2
3
4
5
6
7
8


8
7
6
5
4
3
2
1
1
2
3
4
5
6
7
8

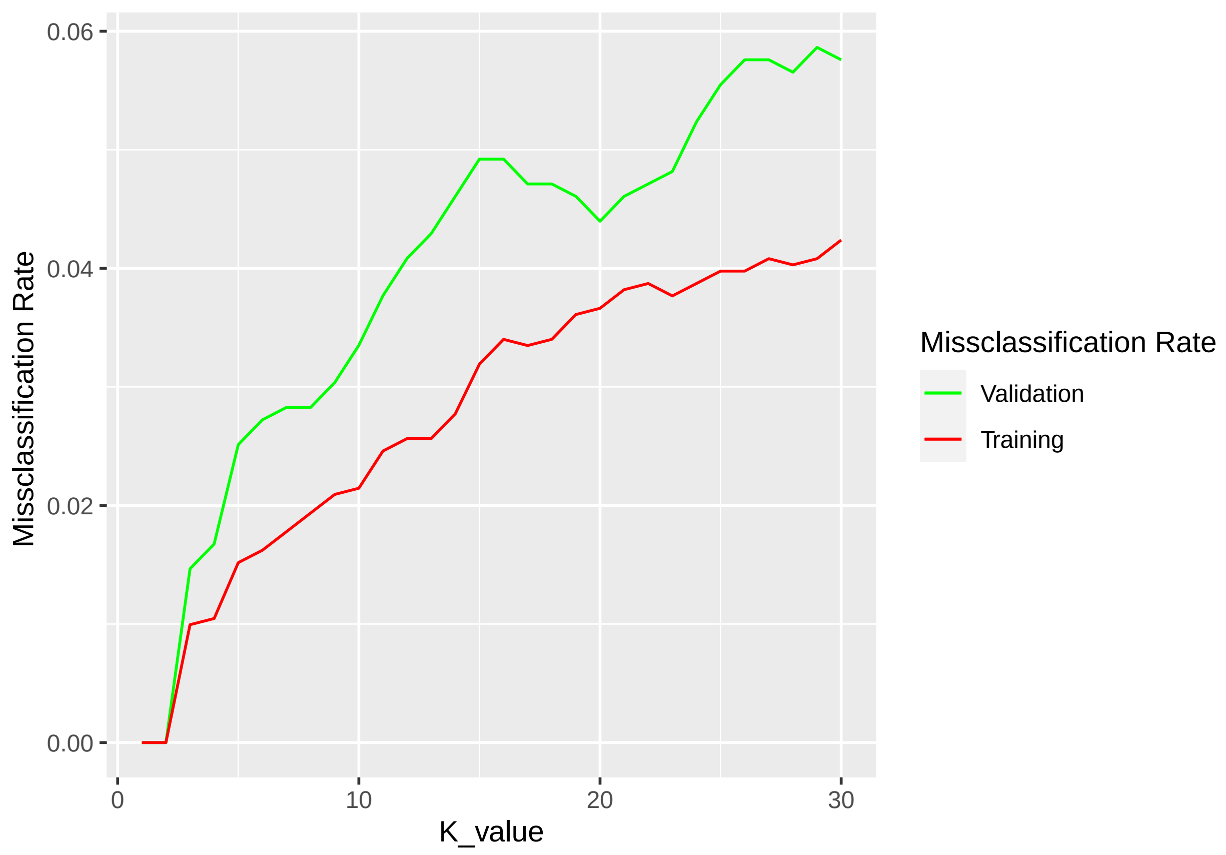


The easiest way to classify the 8 is 3 and the toughest way to classify the 8 is 9 and 4.

3.Find the optimal K value for KNN model for this dataset.(k=1 to 30)

```
##  [1] 0.000000000 0.000000000 0.009942439 0.010465725 0.015175301 0.016221873
##  [7] 0.017791732 0.019361591 0.020931450 0.021454736 0.024594453 0.025641026
## [13] 0.025641026 0.027734171 0.031920460 0.034013605 0.033490319 0.034013605
## [19] 0.036106750 0.036630037 0.038199895 0.038723182 0.037676609 0.038723182
## [25] 0.039769754 0.039769754 0.040816327 0.040293040 0.040816327 0.042386185
```

```
##  [1] 0.00000000 0.00000000 0.01465969 0.01675393 0.02513089 0.02722513
##  [7] 0.02827225 0.02827225 0.03036649 0.03350785 0.03769634 0.04083770
## [13] 0.04293194 0.04607330 0.04921466 0.04921466 0.04712042 0.04712042
## [19] 0.04607330 0.04397906 0.04607330 0.04712042 0.04816754 0.05235602
## [25] 0.05549738 0.05759162 0.05759162 0.05654450 0.05863874 0.05759162
```

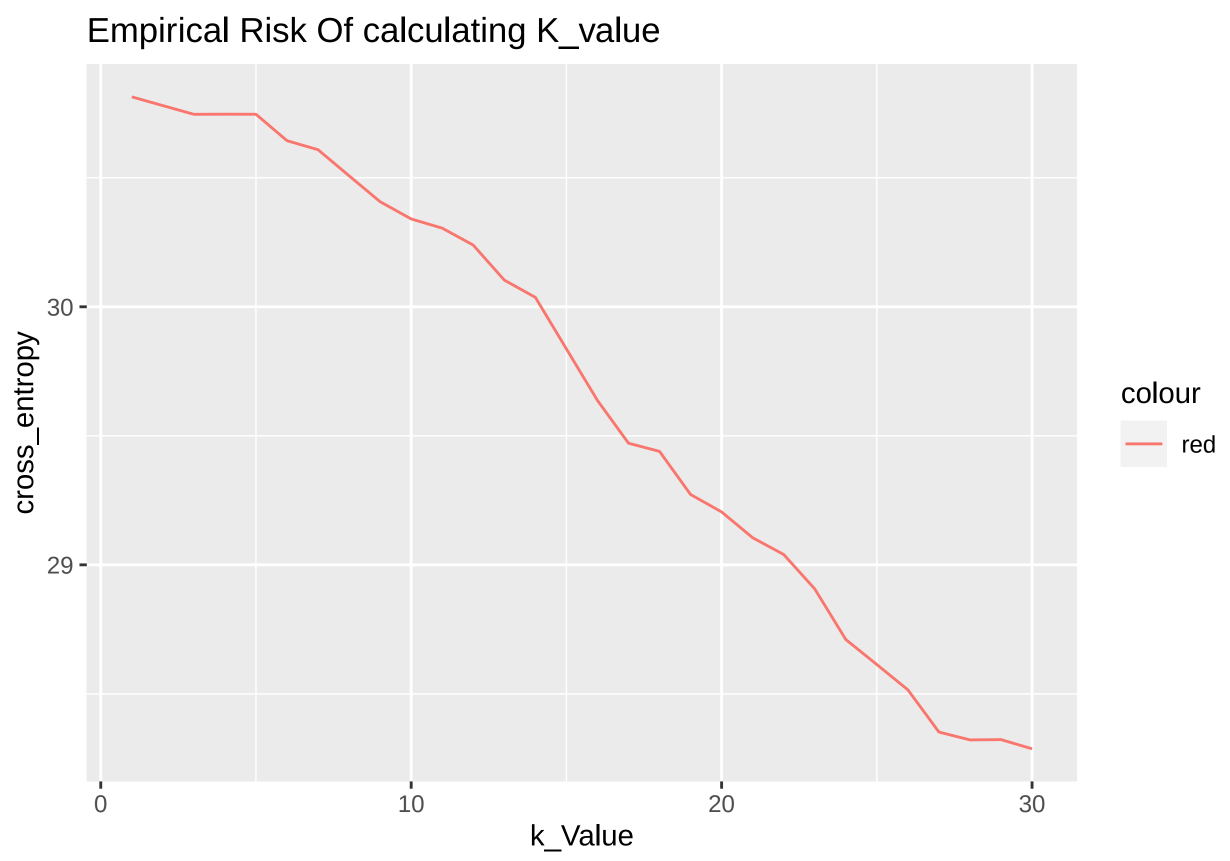


```
## calculated optimum value of K is:  1
```

Based on the varying K value to the model the optimal k value for this dataset is 1. In this hand written digit recognition dataset bias between the data is low and the variance among the data points is high.Miss classification rate for validation and training set is not equal for all the value of k(1 to 30). But the Error rate is equal in the k= 1 and 2.

5.Empirical risk for the validation data as cross-entropy

```
##  [1] 30.81365 30.77966 30.74592 30.74640 30.74618 30.64378 30.60867 30.50763
##  [9] 30.40723 30.34040 30.30476 30.23907 30.10344 30.03662 29.83690 29.63724
## [17] 29.47151 29.43961 29.27262 29.20510 29.10504 29.04018 28.90677 28.71073
## [25] 28.61364 28.51576 28.35212 28.32167 28.32275 28.28716
```



```
## calculated optimum value of K by using the cross entropy function is:  28.28716
```

In this case the entropy function , Empirical risk of the entropy value get slightly decreased when the value of k increased.So the log loss of this function is decreased depend only on the k value.

Appendix: All code for this report

```
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2)
library(glmnet)
library(psych)
library(tidyr)
library(kknn)
#Load Data
dataset = read.csv("C:/Users/Mounish/Desktop/ML/Lab1/optdigits.csv")
cat("Dimension of Dataset is",dim(dataset),"\n")

#Split the data into train, validation and test

set.seed(12345)
sample_train<- sample(seq_len(nrow(dataset)), size = floor(0.50*nrow(dataset)))
sample_valid<- sample(seq_len(nrow(dataset)), size = floor(0.25*nrow(dataset)))
sample_test <- sample(seq_len(nrow(dataset)), size = floor(0.25*nrow(dataset)))
```

```r
train     <- dataset[sample_train, ]
validation<- dataset[sample_valid, ]
test      <- dataset[sample_test, ]


knn.fit <- kknn(as.factor(X0.26)~., train=train,test=train ,k = 30,
                  kernel = "rectangular")

knn.fit_v <- kknn(as.factor(X0.26)~., train=validation,test=validation,k = 30,
                  kernel = "rectangular")

knn.fit_t <- kknn(as.factor(X0.26)~., train=test,test=test,k = 30,
                kernel = "rectangular")

pred.knn <- fitted(knn.fit)
pred.knn_v <- fitted(knn.fit_v)
pred.knn_t <- fitted(knn.fit_t)

c<-table(train$X0.26,as.factor(pred.knn)) # for train data
cat("Confusion matrix for the training data.")
print(c)
v<-table(test$X0.26,as.factor(pred.knn_t)) # for test data
cat("Confusion matrix for the test data.")
print(c)
missclassrate=function(y,y_i)
  {
  n=length(y)
  v<-1-(sum(diag(table(y,y_i)))/n)
  return(v)
}


missclassrate(train$X0.26,as.factor(pred.knn)) # for training data


missclassrate(test$X0.26,as.factor(pred.knn_t)) # for test data

v=data.frame(knn.fit$prob)

estm_pb <- colnames(v)[apply(v, 1, which.max)]

v$y<-train$X0.26

v$fit <- knn.fit$fitted

v$estm_pb <- estm_pb
###
y_8 <- v[v$y == 8,]
yhat_8 <- y_8[y_8$fit == 8,]
###
# Easy
easy <- as.numeric(row.names(yhat_8[order(-yhat_8[,9]),][1:2,]))
```

```r
# Tough
tougher <- as.numeric(row.names(yhat_8[order(yhat_8[,9]),][1:3,]))

col=heat.colors(12)

heatmap(t(matrix(unlist(train[easy[1],-65]), nrow=8)),Colv = NA, Rowv = NA,col=rev(heat.colors(12)))

heatmap(t(matrix(unlist(train[easy[2],-65]), nrow=8)), Colv = NA, Rowv = NA,col=rev(heat.colors(12)))


heatmap(t(matrix(unlist(train[tougher[1],-65]), nrow=8)), Colv = NA, Rowv = NA,col=rev(heat.colors(12))

heatmap(t(matrix(unlist(train[tougher[2],-65]), nrow=8)), Colv = NA, Rowv = NA,col=rev(heat.colors(12))

heatmap(t(matrix(unlist(train[tougher[3],-65]), nrow=8)), Colv = NA, Rowv = NA,col=rev(heat.colors(12))

k=1
k.optm=c()
y.optm=c()
for (i in 1:30){
  knn.fit <-kknn(as.factor(X0.26)~., train=train,test=train, k = i,kernel = "rectangular")

  knn.fit_v <-kknn(as.factor(X0.26)~., train=validation,test=validation, k = i,kernel = "rectangular")

  ypred = fitted(knn.fit)

  vpred = fitted(knn.fit_v)

  k.optm[i] = 1-(sum(diag(as.matrix(table(Actual = train$X0.26, Predicted = ypred))))/nrow(train))

  y.optm[i] = 1-(sum(diag(as.matrix(table(Actual = validation$X0.26, Predicted = vpred))))/nrow(validat

}
k.optm

y.optm


my.df  <- data.frame(K_Value = c(1:30), Training= c(k.optm), Validation = c(y.optm))

plot3<-ggplot( ) +
  geom_line(aes(x=my.df$K_Value,y=my.df$Validation,colour="green")) +
  geom_line(aes(x=my.df$K_Value,y=my.df$Training,colour="red"))+
  ylab("Missclassification Rate ") +xlab("K_value")+
scale_color_manual(name = "Missclassification Rate", labels = c("Validation ", "Training "),
                   values =c("green", "red"))


print(plot3)

optm_value_k <- which.min(y.optm - k.optm )

cat("calculated optimum value of K is: ", optm_value_k)
```

```r
rp <- function(i){
  n <- rep(0,10)
  n[i+1] <- 1
  return(I(n))
}
 er<-c()
 for (i in 1:30){
   knn.fit <-kknn(as.factor(X0.26)~., train=train,test=validation, k = i,kernel = "rectangular")

   x<- data.frame(knn.fit$prob)

   max_prob <- colnames(x)[apply(x ,1,which.max)]

   x$target <- validation$X0.26

   x$fit <- knn.fit$fitted

   x$max_prob <- max_prob

   x$binary <- (lapply(as.numeric(x$target)-1, rp))

   #cross entropy by using log

   for (val in 1:nrow(x)){

     x[val, "cross_entropy"] <- -sum(log(x[val,1:10]+1e-15)* x[[val, "binary"]])
   }

   er[i] <- mean(x$cross_entropy)
 }

 er

 df<-data.frame(cross_entropy=er,k_Value=c(1:30))

 plot4<-ggplot(df,aes(x=k_Value,y=cross_entropy,col="red" ))  +
        geom_line()+ggtitle("Empirical Risk Of calculating K_value")
 print(plot4)

best_optm_k_value <-min(er)

 cat("calculated optimum value of K by using the cross entropy function is: ", best_optm_k_value)
```