Machine Learning Lab 1

Farid Musayev, Kristina Levina, and Wuhao Wang

18 November, 2021

Contribution

All the group members have finished 3 assignments and helped each other with problems and reached a consensus on solutions to the problems. To be more specific:

Wuhao Wang is responsible for assignment 1 and its report.
Kristina Levina is responsible for assignment 2 and its report.
Farid Musayev is responsible for assignment 3 and its report.

Assignment1

Q1:

Load data and then split them into three part. And also rename the columns.

Q2:

1) digits 6 0 7 can be predicted very well both in training data and test data(over 98.8% correct rate).

2) digits 2 3 9 are also in good condition(over 95%).

3) digits 1 5, the performance not very good(over 90%).

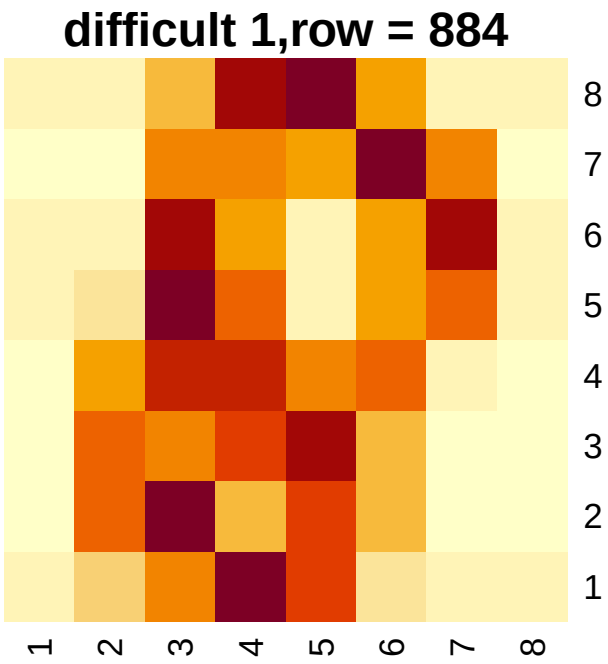
4) digits 4 8, not very good(lower than 90%).

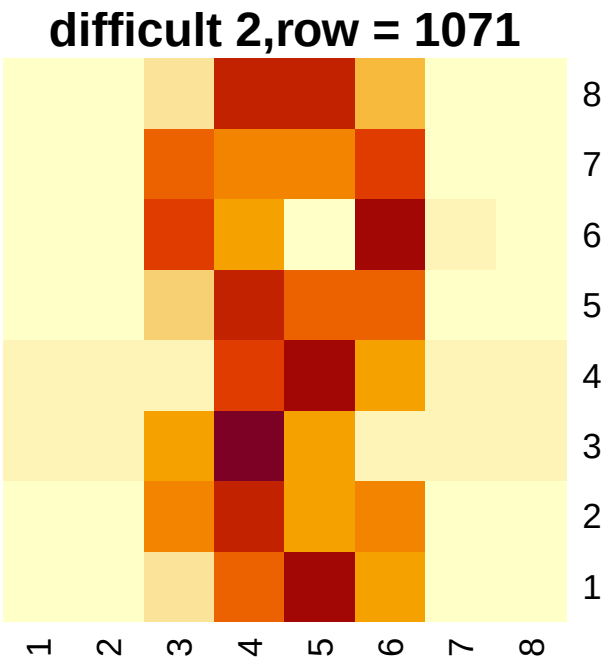
5) overall : average error rate is 0.045 in training data and 0.053 in test data. This model is not over fitting.

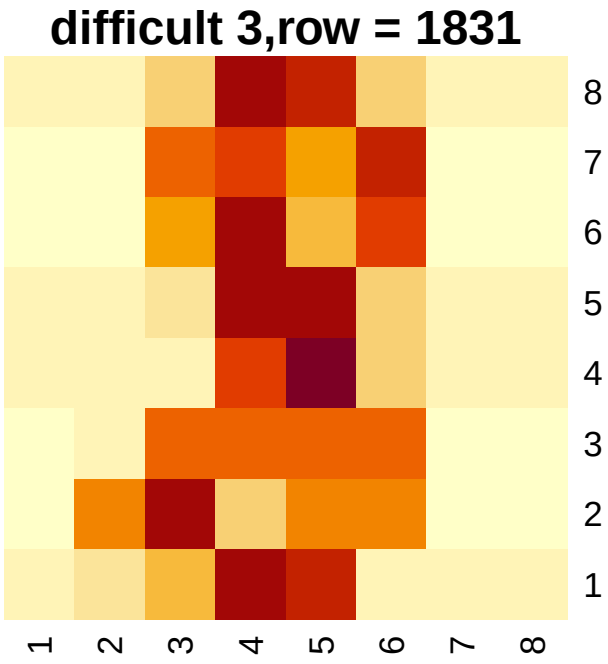
```
##                       0          1          2          3         4          5 6
## MisC rates 0.01282051 0.05813953 0.02970297 0.03603604 0.1376147 0.09708738 0
##                       7         8          9
## MisC rates 0.008928571 0.1025641 0.04494382
```

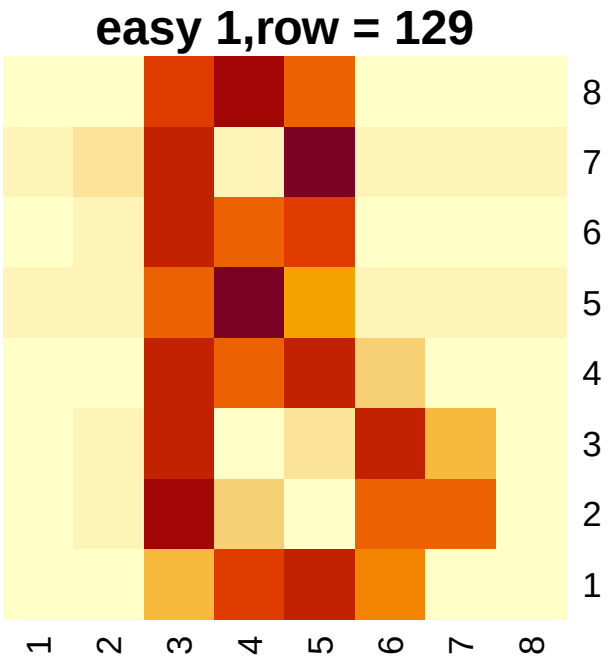
Q3:

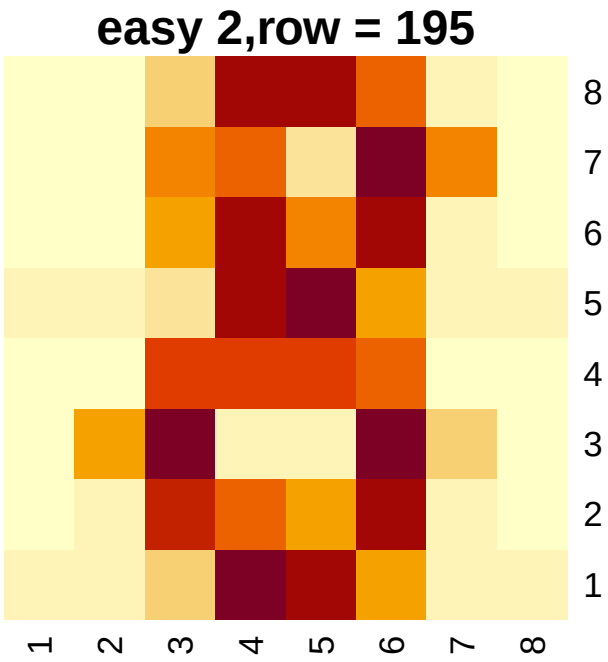
Using ***sort()*** function to find the correct prediction with highest probability and lowest probability. Then reshape them into 8*8 matrix. By viewing these observations in this way, is much easier to recognize.


difficult 1,row = 884
8
7
6
5
4
3
2
1
1 2 3 4 5 6 7 8


difficult 2,row = 1071
8
7
6
5
4
3
2
1
1 2 3 4 5 6 7 8


difficult 3,row = 1831
8
7
6
5
4
3
2
1
1 2 3 4 5 6 7 8


easy 1,row = 129
8
7
6
5
4
3
2
1
1 2 3 4 5 6 7 8



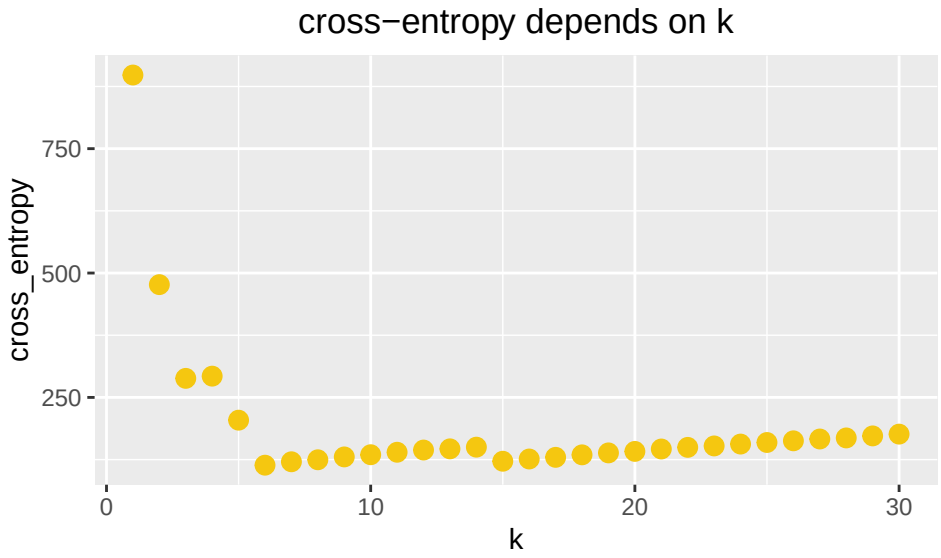
Q4:

Training data into a loop with k increasing from 1 to 30, the complexity grows when k increases. Overall, the error rate keeps increasing when k increases. For the valid data,the optimal k is k == 3, then error rate keeps increasing until k == 25. when k == 3, the error rate for training, valid and test data are around 0.012, 0.025 and 0.024.

###

Q5: Now, the best k is 6. for this problem, cross-entropy not only help us know more details about our

model.
For example, if an observation is '8' with prediction probability 0,0,0,0,0,0,0,1,0 another observation is also '8' with prediction probability 0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.2,0.1, although two observations will be predicted as '8', obviously the result of first observation is better than second one. In other words, the first prediction result is far more confirmed, and the ideal situation is : all the prediction shows similar result to first prediction result. That is why cross-entropy is better in this problem.



Appendix

```
library(ggplot2)
library(kknn)
library(reshape2)
# Assignment 1 KNN-method for hand-writing
################################
# loading data
data <- read.csv('optdigits.csv',header = FALSE)
#############################################################################
# pre-processing (change label into factor and rename columns)
data[,65] <- as.factor(data[,65])
for_rename <- 1:64
for_rename <- as.character(for_rename)
for_rename <- paste0('f',for_rename)
for_rename <- c(for_rename,'target')
colnames(data) <- for_rename
#############################################################################
# Question 1
#############################################################################
set.seed(12345)
n <- dim(data)[1]
id <- sample(1:n, floor(n*0.5))
train <- data[id,]
set.seed(12345)
id1 <- setdiff(1:n, id)
id2 <- sample(id1, floor(n*0.25))
valid <- data[id2,]
set.seed(12345)
id3 <- setdiff(id1,id2)
test <- data[id3,]
```

```
rm(id,id1,id2,id3,n,for_rename)
################################################################################
# Question 2
################################################################################
# fit the model
kn <- train.kknn(target~.,data = train,ks = 30,kernel =  'rectangular')
# evaluate in the train data
pred_train <- predict(kn,train)
t_train <- table(train$target,pred_train)
# evaluate in the test data
pred_test <- predict(kn,test)
t_test <- table(test$target,pred_test)
# calculate the error rate
# for train data
error_rate_train <- 1-sum(diag(t_train))/sum(t_train) # 0.045
# for test data
error_rate_test <- 1-sum(diag(t_test))/sum(t_test) # 0.053

################################################################################
#Question 3
################################################################################
# train the model and fit the model to training data to get the probability.
wri_knn <- kknn(target~.,train = train,test = train,kernel = 'rectangular',k=30)
pre_knn <- predict(wri_knn)
# index_list_of_8 contains all the observations whose prediction and label are both 8.
index_list_of_8 <- which(train[,65]==8)[which(which(train[,65]==8)%in%which(pre_knn==8))]
# these two vector contains the probability of lowest and highest probability
lowest_prob <- sort(wri_knn[['prob']][index_list_of_8,9])[1:3]
highest_prob <- sort(wri_knn[['prob']][index_list_of_8,9],decreasing = TRUE)[1:2]
# find the most 2 difficult observations
lowest_pro_index <- index_list_of_8[which(wri_knn[['prob']][index_list_of_8,9]%in%lowest_prob)][1:3]
# find the 3 easiest observations
highest_pro_index <- index_list_of_8[which(wri_knn[['prob']][index_list_of_8,9]%in%highest_prob)][1:2]
# get the pixel map of good and bad
pixelMapsBad <- train[lowest_pro_index,-65]
pixelMapsGood <- train[highest_pro_index,-65]
# restore the graphic, using heatmap() to view.
figGoodEight1<- matrix(unlist(pixelMapsGood[1,], use.names = FALSE), ncol = 8, nrow = 8, byrow = TRUE)
figGoodEight2<- matrix(unlist(pixelMapsGood[2,], use.names = FALSE), ncol = 8, nrow = 8, byrow = TRUE)
figBadEight1 <- matrix(unlist(pixelMapsBad[1,], use.names = FALSE), ncol = 8, nrow = 8, byrow = TRUE)
figBadEight2 <- matrix(unlist(pixelMapsBad[2,], use.names = FALSE), ncol = 8, nrow = 8, byrow = TRUE)
figBadEight3 <- matrix(unlist(pixelMapsBad[3,], use.names = FALSE), ncol = 8, nrow = 8, byrow = TRUE)
# heatmap(figGoodEight1, Colv = NA, Rowv = NA, main = 'highest probability,row = 129')
# heatmap(figGoodEight2, Colv = NA, Rowv = NA, main = 'highest probability,row = 195')
# heatmap(figBadEight1, Colv = NA, Rowv = NA, main = 'lowest probability,row = 884')
# heatmap(figBadEight2, Colv = NA, Rowv = NA, main = 'lowest probability,row = 1071')
# heatmap(figBadEight3, Colv = NA, Rowv = NA, main = 'lowest probability,row = 1831')
################################################################################
#Question 4
################################################################################
kx <- 1:30
error_train <- c()
error_valid <- c()
```

```
for(i in 1:30)
{
  # model training
  kn <- train.kknn(target~.,data = train,ks = i,kernel = "rectangular")
  # predication
  predic_train <- predict(kn,train)
  predic_valid <- predict(kn,valid)
  # make confusion matrix
  t_train <- table(train$target,predic_train,dnn = c('true','predict'))
  t_valid <- table(valid$target,predic_valid,dnn = c('true','predict'))
  # compute error rate
  error_rate_train <- 1-sum(diag(t_train))/sum(t_train)
  error_rate_valid <- 1-sum(diag(t_valid))/sum(t_valid)
  # add error_rate into a vector for drawing graph
  error_train <- c(error_train,error_rate_train)
  error_valid <- c(error_valid,error_rate_valid)
  # remove redundant data to save memory
  rm(predic_train,predic_valid,error_rate_train,error_rate_valid)
}
# gathering data into a data.frame to apply ggplot2
df <- data.frame(k = 1:30,error_train=error_train,error_valid=error_valid)
# reshape data so that 2 plot lines can be plotted in a single graph
df1 <- melt(df,id.vars='k')
# plot
p1 <-ggplot(df1,aes(x=k,y=value))+
    geom_point(aes(color=variable))+
    ggtitle('error_rate : valid vs train')+
    theme(plot.title = ggplot2::element_text(hjust=0.5))
# from the graph the optimal k is 7
kbest <- train.kknn(target~.,data = train,kernel = 'rectangular',ks=7)
pred_test <- predict(kbest,newdata = test)
t <- table(test$target,pred_test)
error_rate_test <- 1-sum(diag(t))/sum(t)
# remove redundant data
rm(df,df1,pred_test,kbest)
################################################################################
#Question 5
################################################################################
ent_list <- c()
for(i in 1:30)
{
  kn <- kknn(target~.,train = train,test = valid,kernel = 'rectangular',k=i)
  ent <- 0
  for(j in 1:length(valid[,1]))
  {
     col_index <- as.numeric(as.character(valid[j,65]))+1
     ent <- ent - log(as.numeric(as.character((kn[['prob']][j,col_index]))))+1e-15)
  }
  ent_list <- c(ent_list,ent)
}
# make df to draw graph
df <- data.frame(k = 1:30,cross_entropy = ent_list)
p5entro <-ggplot(df,aes(x=k,y=cross_entropy))+
```

```
          geom_point(color=7,size = 3)+
          ggtitle('cross-entropy depends on k')+
          theme(plot.title = ggplot2::element_text(hjust=0.5))
#print(p5entro)
```