Laboration report in Machine Learning

# Computer lab 1 block 1

**732A99**

Sigme Cinar
Duc Tran
William Wiik

Division of Statistics and Machine Learning
Department of Computer Science
Linköping University

10 november 2023

# Contents

# 1 Assignment 1. Handwritten digit recognition with K-nearest neighbors.

The data in this task is from the file optdigits.csv. Data consists of 3822 handwritten digits from 0 to 9 and are stored as images of size 8x8.

## 1.1 1.1

**Question:** Import the data into R and divide it into training, validation and test sets (50%/25%/25%) by using the partitioning principle specified in the lecture slides.

**Answer:** The code used is presented as follows:

```
# Read in data
data <- read.csv("optdigits.csv")

# Renaming the response variable and changing it to a factor variable
data <- rename(data, y=X0.26)
data$y <- as.factor(data$y)

# Partitioning training data (50%)
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
train=data[id,]

# Partitioning validation data (25%)
id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.25))
valid=data[id2,]

# Partitioning test data (25%)
id3=setdiff(id1,id2)
test=data[id3,]
```

## 1.2 1.2

**Question:** Use training data to fit 30-nearest neighbor classifier with function kknn() and kernel="rectangular" from package kknn and estimate

- Confusion matrices for the training and test data (use table())
- Misclassification errors for the training and test data

**Answer:** The confusion matrices for models trained on training data with k=30 and evaluated on training data and test data are presented in table 1 and 2.

```
# kknn on training data and evaluation on training data
model_kknn_train <-
  kknn(formula = y~., train = train, test = train,
     kernel = "rectangular", k=30)
conf_mat_train <- table(train$y, model_kknn_train$fitted.values)
```

```
acc_train <- sum(diag(conf_mat_train)) / sum(conf_mat_train)
miss_train <- 1-acc_train

# kknn on training data and evaluation on test data
model_kknn_test <-
  kknn(formula = y~., train = train, test = test,
       kernel = "rectangular", k=30)
conf_mat_test <- table(test$y, model_kknn_test$fitted.values)
acc_test <- sum(diag(conf_mat_test)) / sum(conf_mat_test)
miss_test <- 1-acc_test

# Rows are true values, columns are model prediction
kable(conf_mat_train, caption = "Confusion matrix for training data.")
```

Table 1: Confusion matrix for 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 |

```
kable(conf_mat_test,  caption = "Confusion matrix for test data.    ")
```

Table 2: Confusion matrix for test data.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 97 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 91 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| 2 | 0 | 0 | 93 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 95 | 0 | 0 | 0 | 2 | 1 | 0 |
| 4 | 1 | 0 | 0 | 0 | 89 | 0 | 1 | 5 | 1 | 3 |
| 5 | 0 | 1 | 0 | 1 | 0 | 79 | 1 | 0 | 0 | 5 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 94 | 0 | 0 | 0 |
| 7 | 0 | 2 | 0 | 0 | 0 | 1 | 0 | 91 | 1 | 0 |
| 8 | 0 | 3 | 0 | 1 | 0 | 0 | 1 | 0 | 86 | 0 |
| 9 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 2 | 1 | 94 |

The missclassification error on training data from table 1 is around 4.24% and on test data from table 2 is around 4.92%.

```
miss_train
```

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

```
miss_test
```

```
## [1] 0.04916318
```

### 1.3 1.3

**Question:** Find any 2 cases of digit "8" in the training data which were easiest to classify and 3 cases that were hardest to classify (i.e. having highest and lowest probabilities of the correct class). Reshape features for each of these cases as matrix 8x8 and visualize the corresponding digits (by using e.g. heatmap() function with parameters Colv=NA and Rowv=NA) and comment on whether these cases seem to be hard or easy to recognize visually.

**Answer:** The code used to find the 2 digits that are hardest to classify were found with the code as follows

```
y <- train$y
fit_y <- model_kknn_train$fitted.values
# probabilities given from number 0 to 9, index 9 = number 8.
prob_8 <- model_kknn_train$prob[, 9]

# Data frame consisting of true value of y, model prediction and the models
# probability that the number is 8.
data_8 <- data.frame(y = y,
                     fit_y = fit_y,
                     prob = prob_8)
data_8$observation_id <- rownames(data_8)

# Only observations with the label 8 is kept.
data_8 <- data_8[data_8$y == "8", ]
head(arrange(data_8, prob), 2)
```

```
##   y fit_y      prob observation_id
## 1 8     6 0.1000000           1624
## 2 8     1 0.1666667           1663
```

From the output, observation 1624 and 1663 were hardest to classify as 8 from the model. The three observations that were easiest to identify as 8 were found with the code as follows

```
tail(arrange(data_8, prob), 3)
```

```
##     y fit_y prob observation_id
## 203 8     8    1           1810
## 204 8     8    1           1811
## 205 8     8    1           1864
```

From the output observation 1810, 1811, and 1864 were three observations that were easiest to identify as 8 with a probability from the model as 100% (in total there were 49 observations that had 100% probability).

A function that reshapes each observation to a 8x8 cases and then visualizing the result in a heatmap was done with the code as follows

```
# Change colour palette to black and white
colfunc <- colorRampPalette(c("white", "black"))

plot_8 <- function(index){
  title <- paste0("Observation: ", index)
  # Reshapes the observations to a 8x8
  plot <- as.matrix(train[index, -65]) # Remove response variable
  plot <- matrix(plot, nrow=8, byrow=TRUE)
  heatmap(plot, col=colfunc(16), Colv=NA, Rowv=NA, main=title)
}
```

The heatmaps for observations 1624 and 1663 are presented in figure 1.

```
plot_8(1624)
plot_8(1663)
```

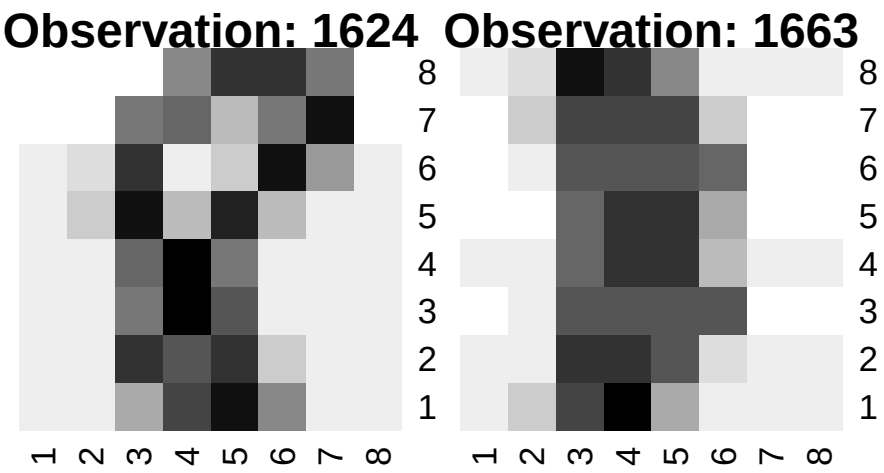


Figure 1: Heatmap for two observations that were hard to classify: 1624 and 1663.

In figure 1, it is hard to visually recognize what number the observations 1624 and 1663 are.

The heatmaps for observations 1810, 1811, and 1864 are presented in figure 2.

```
plot_8(1810)
plot_8(1811)
plot_8(1864)
```

In figure 2, is is easy to visually recognize what number the observations 1810, 1811, and 1864 are.

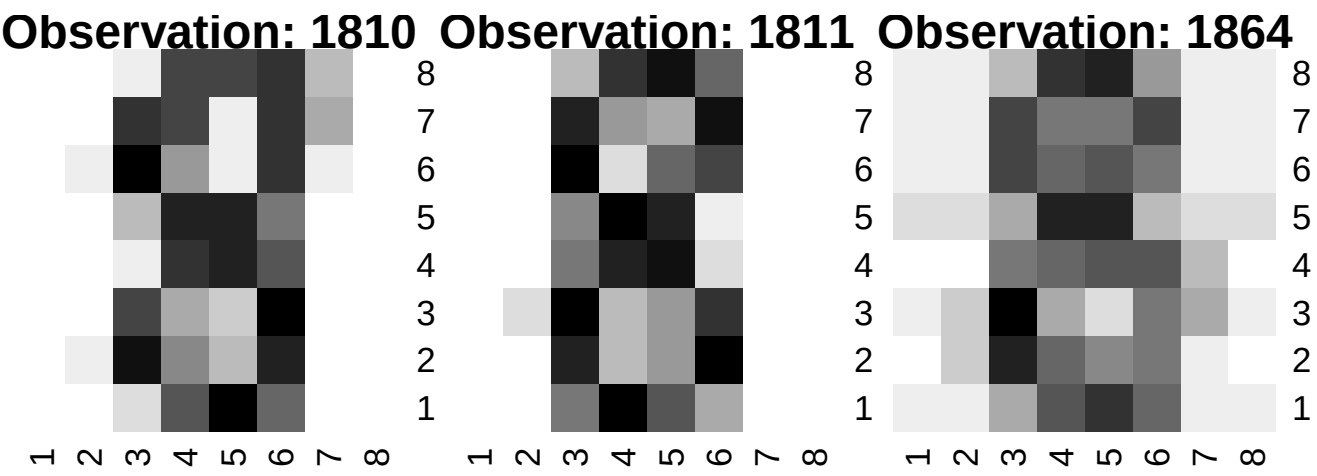


Figure 2: Heatmap for three observations that were easy to classify: 1810, 1811, and 1864.

# 2 Statement of Contribution

We worked on the assignment individually for the computer labs (to be more efficient when asking questions), Duc on task 1, Sigme on task 2, and William on task 3. We later solved all assignment individually and compared and discussed our solutions before dividing the task of writing the laboration report.

## 2.1 Question 1

Text written by Duc.

## 2.2 Question 2

Text written by Sigme.

## 2.3 Question 3

Text written by William.

# 3 Appendix

The code used in this laboration report are summarised in the code as follows:

```
library(ggplot2)
library(kknn)
library(dplyr)
library(knitr)
library(cowplot)
knitr::opts_chunk$set(
  echo = TRUE,
  fig.width = 4.5,
  fig.height = 3)
# Read in data
data <- read.csv("optdigits.csv")

# Renaming the response variable and changing it to a factor variable
data <- rename(data, y=X0.26)
data$y <- as.factor(data$y)

# Partitioning training data (50%)
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
train=data[id,]

# Partitioning validation data (25%)
id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.25))
valid=data[id2,]
```

```r
# Partitioning test data (25%)
id3=setdiff(id1,id2)
test=data[id3,]
# kknn on training data and evaluation on training data
model_kknn_train <-
  kknn(formula = y~., train = train, test = train,
     kernel = "rectangular", k=30)
conf_mat_train <- table(train$y, model_kknn_train$fitted.values)
acc_train <- sum(diag(conf_mat_train)) / sum(conf_mat_train)
miss_train <- 1-acc_train

# kknn on training data and evaluation on test data
model_kknn_test <-
  kknn(formula = y~., train = train, test = test,
       kernel = "rectangular", k=30)
conf_mat_test <- table(test$y, model_kknn_test$fitted.values)
acc_test <- sum(diag(conf_mat_test)) / sum(conf_mat_test)
miss_test <- 1-acc_test

# Rows are true values, columns are model prediction
kable(conf_mat_train, caption = "Confusion matrix for training data.")
kable(conf_mat_test,  caption = "Confusion matrix for test data.    ")
miss_train
miss_test
y <- train$y
fit_y <- model_kknn_train$fitted.values
# probabilities given from number 0 to 9, index 9 = number 8.
prob_8 <- model_kknn_train$prob[, 9]

# Data frame consisting of true value of y, model prediction and the models
# probability that the number is 8.
data_8 <- data.frame(y = y,
                     fit_y = fit_y,
                     prob = prob_8)
data_8$observation_id <- rownames(data_8)

# Only observations with the label 8 is kept.
data_8 <- data_8[data_8$y == "8", ]
head(arrange(data_8, prob), 2)
tail(arrange(data_8, prob), 3)
# Change colour palette to black and white
colfunc <- colorRampPalette(c("white", "black"))

plot_8 <- function(index){
  title <- paste0("Observation: ", index)
  # Reshapes the observations to a 8x8
  plot <- as.matrix(train[index, -65]) # Remove response variable
  plot <- matrix(plot, nrow=8, byrow=TRUE)
  heatmap(plot, col=colfunc(16), Colv=NA, Rowv=NA, main=title)
```

```
}
plot_8(1624)
plot_8(1663)
plot_8(1810)
plot_8(1811)
plot_8(1864)
```