ETC3250/5250 IML Asignment 3 Solution

Code

Thuy Tien Pham (30806933)

2023-05-15

Hide

```
# Load the packages that you will use to complete this assigment.
library(dplyr)
library(tidyverse)
library(rsample)
library(yardstick)
library(stats)
library(ranger)
library(ggdendro)
library(gridExtra)
library(kknn)
library(ipred)
library(rpart)
library(gbm)
library(xgboost)
```

Hide

```
data <- read_csv("data30806933.csv")
```

Preliminary analysis

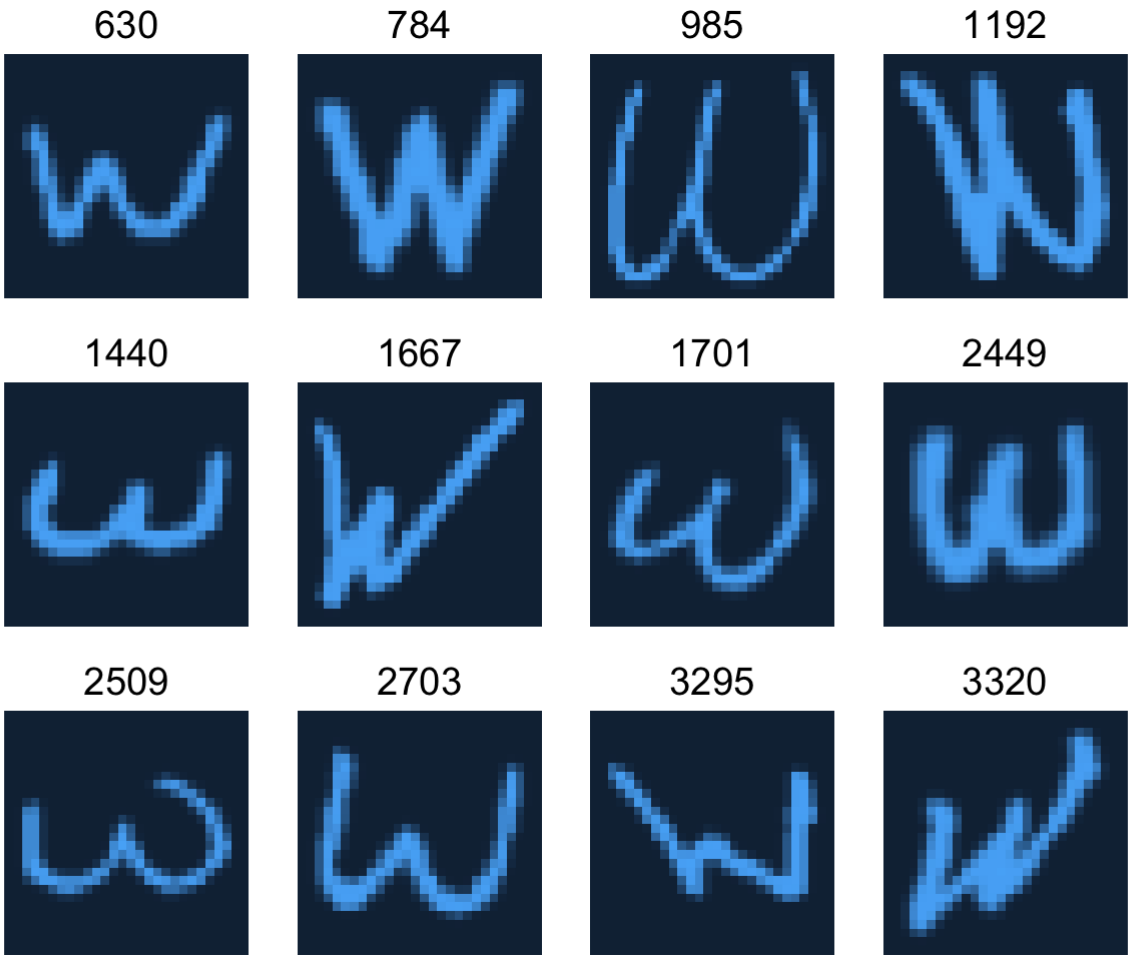
Question 1

The letter in my data is W.

Question 2

Hide

```
set.seed(17072001)
imagedata_to_plotdata <- function(data = data,
                                  w = 28,
                                  h = 28,
                                  which = sample(1:3418, 12)) {
  data %>%
    mutate(id = 1:n()) %>%
    filter(id %in% which) %>%
    pivot_longer(starts_with("V")) %>%
    mutate(col = rep(rep(1:w, each = h), n_distinct(id)),
           row = rep(rep(1:h, times = w), n_distinct(id)))
}

gletters <- imagedata_to_plotdata(data) %>%
    ggplot(aes(col, row)) +
    geom_tile(aes(fill = value)) +
    facet_wrap(~id, nrow = 3) +
    scale_y_reverse() +
    theme_void(base_size = 18) +
    guides(fill = "none") +
    coord_equal()

gletters
```



Question 3

Hide

```
data_pca <- prcomp(data)
```

Hide

```
cumsum(data_pca$sdev^2 / sum(data_pca$sdev^2)) %>% head()
```

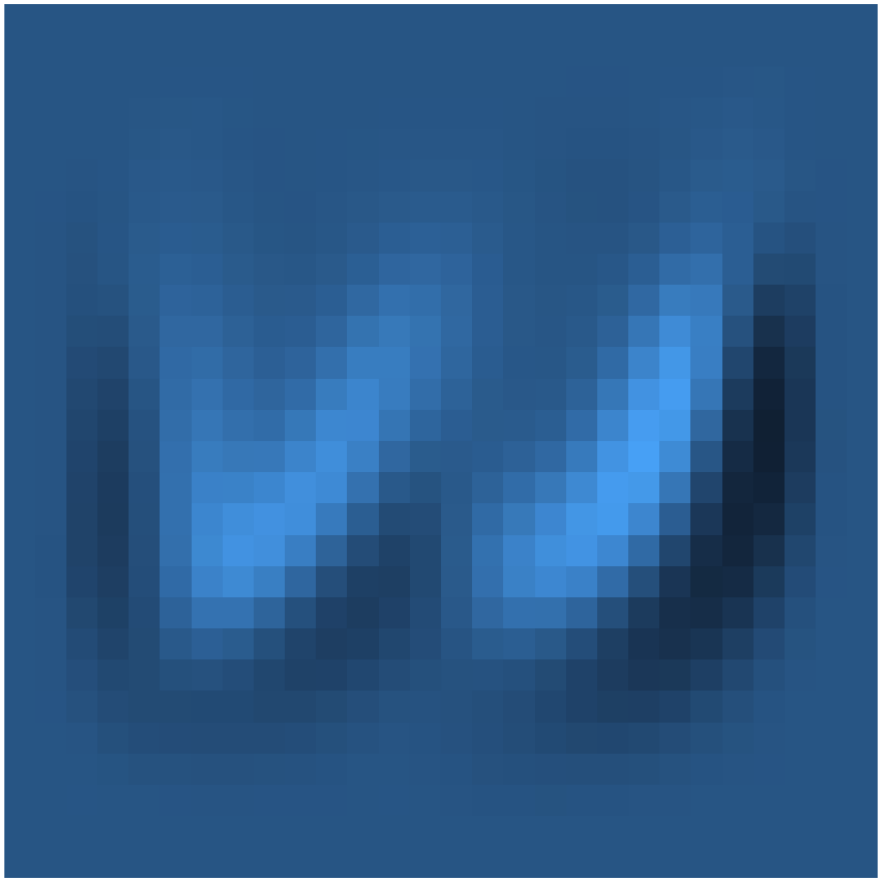
```
## [1] 0.0994894 0.1821534 0.2454012 0.3037705 0.3571161 0.3920728
```

The total variation that the first 5 principal components explain in the data is 0.3571161.

Question 4

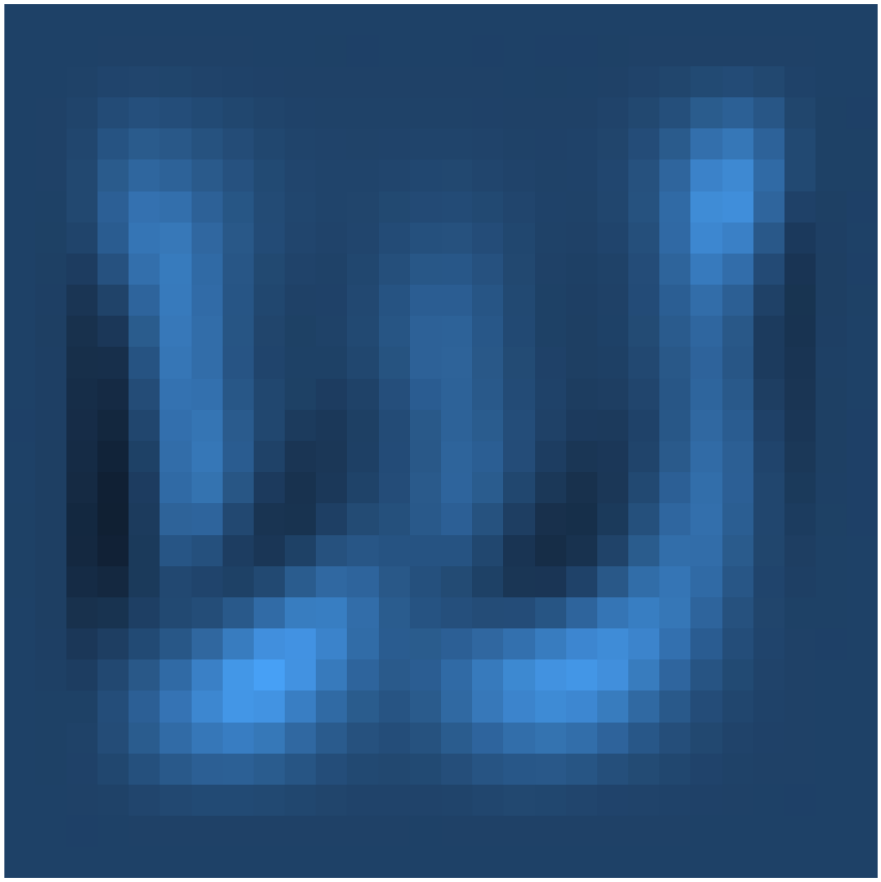
Hide

```
set.seed(1707)
Xnew <- data_pca$x[, 1] %*% t(data_pca$rotation[, 1])

sample_ids <- sample(1:3418, 1)

Pdata <- as.data.frame(Xnew) %>%
  imagedata_to_plotdata(which = sample_ids)
PC1_loading <- Pdata %>%
  ggplot(aes(col, row)) +
    geom_tile(aes(fill = value)) +
    scale_y_reverse() +
    theme_void(base_size = 18) +
    guides(fill = "none") +
    coord_equal()
PC1_loading
```



Hide

```
set.seed(1707)
Xnew2 <- data_pca$x[, 2] %*% t(data_pca$rotation[, 2])

sample_ids <- sample(1:3418, 1)

Pdata <- as.data.frame(Xnew2) %>%
  imagedata_to_plotdata(which = sample_ids)
PC2_loading <- Pdata %>%
  ggplot(aes(col, row)) +
    geom_tile(aes(fill = value)) +
    scale_y_reverse() +
    theme_void(base_size = 18) +
    guides(fill = "none") +
    coord_equal()
PC2_loading
```



Question 5

Hide

```
haverage <- hclust(dist(data_pca$x), method = "average")
haverage
```

```
##
## Call:
## hclust(d = dist(data_pca$x), method = "average")
##
## Cluster method   : average
## Distance         : euclidean
## Number of objects: 3418
```

Question 6

Hide

```
c_average <- cutree(haverage, k =4)
table(c_average)
```

```
## c_average
##    1    2    3    4
## 3406    6    5    1
```

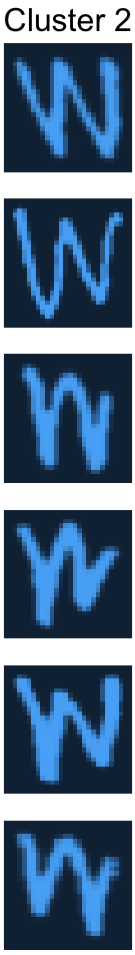
Question 7

Hide

```
set.seed(1707)
data_labels <- data %>%
  mutate(caverage = c_average) %>%
  imagedata_to_plotdata(which = sample(1:3418, 3418))

cluster1 <- data_labels %>%
  filter(caverage == 1) %>%
  filter(id %in% 1:10)

cluster_plot1 <- cluster1 %>%
  ggplot(aes(col, row)) +
    geom_tile(aes(fill = value)) +
    facet_wrap(~id, nrow = 10) +
    scale_y_reverse() +
    theme_void(base_size = 10) +
    guides(fill = "none") +
    coord_equal() +
    ggtitle("Cluster 1") +
    theme(strip.background = element_blank(), strip.text.x = element_blank(),  plot.t
itle = element_text(hjust = 0.5))
cluster_plot1
```

Hide

```
set.seed(1707)
cluster2 <- data_labels %>%
  filter(caverage == 2)

cluster_plot2 <- cluster2 %>%
  ggplot(aes(col, row)) +
    geom_tile(aes(fill = value)) +
    facet_wrap(~id, nrow = 6) +
    scale_y_reverse() +
    theme_void(base_size = 10) +
    guides(fill = "none") +
    coord_equal() +
    ggtitle("Cluster 2") +
    theme(strip.background = element_blank(), strip.text.x = element_blank(),  plot.t
itle = element_text(hjust = 0.5))
cluster_plot2
```



Hide

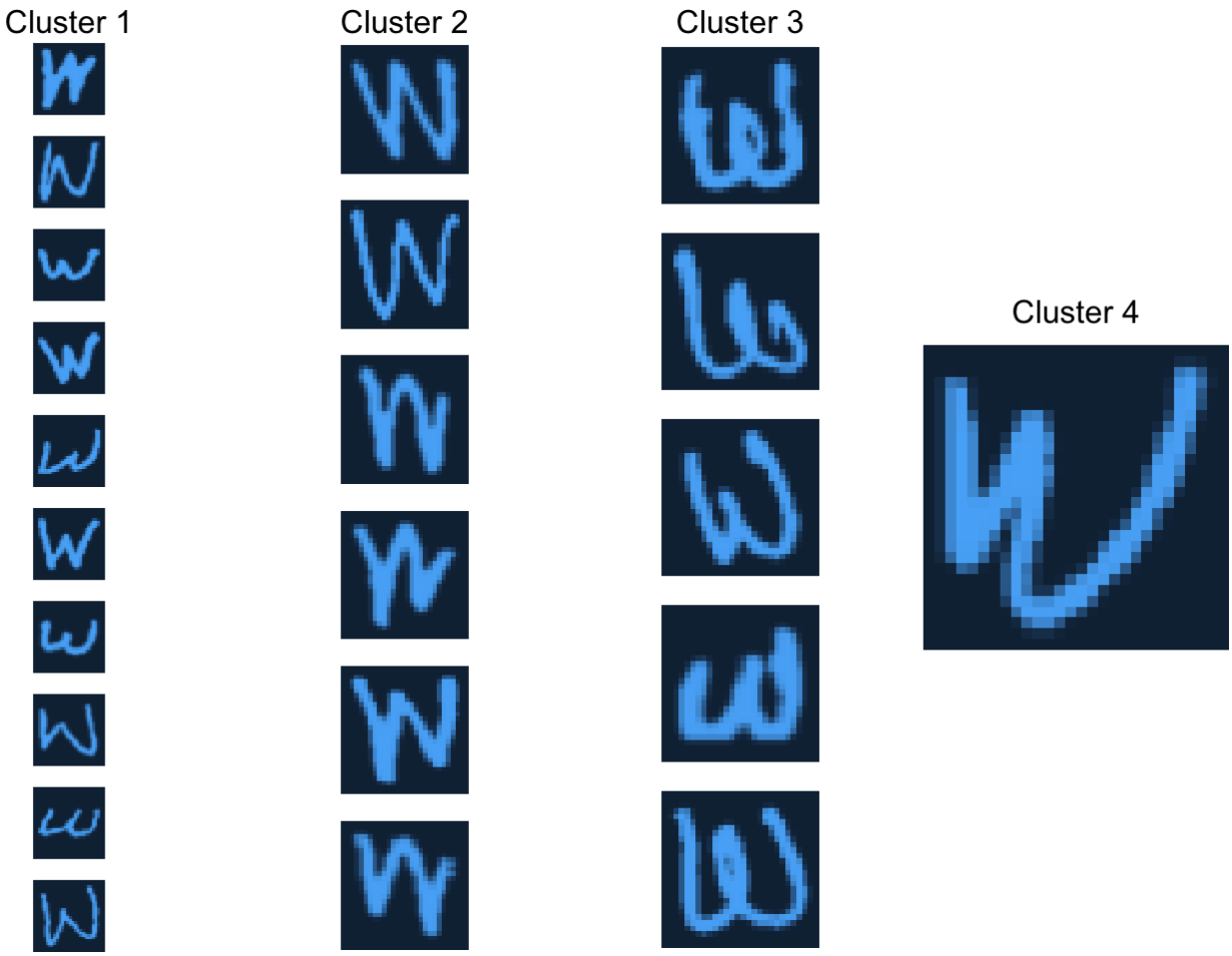
```
set.seed(1707)
cluster3 <- data_labels %>%
  filter(caverage == 3)

cluster_plot3 <- cluster3 %>%
  ggplot(aes(col, row)) +
    geom_tile(aes(fill = value)) +
    facet_wrap(~id, nrow = 5) +
    scale_y_reverse() +
    theme_void(base_size = 10) +
    guides(fill = "none") +
    coord_equal() +
    ggtitle("Cluster 3") +
    theme(strip.background = element_blank(), strip.text.x = element_blank(),  plot.t
itle = element_text(hjust = 0.5))

cluster_plot3
```

Hide

```
set.seed(1707)
cluster4 <- data_labels %>%
  filter(caverage == 4)

cluster_plot4 <- cluster4 %>%
  ggplot(aes(col, row)) +
    geom_tile(aes(fill = value)) +
    facet_wrap(~id, nrow = 1) +
    scale_y_reverse() +
    theme_void(base_size = 10) +
    guides(fill = "none") +
    coord_equal() +
    ggtitle("Cluster 4") +
    theme(strip.background = element_blank(), strip.text.x = element_blank(), plot.ti
tle = element_text(hjust = 0.5))
cluster_plot4
```

Hide

```
gridExtra::grid.arrange(cluster_plot1, cluster_plot2, cluster_plot3, cluster_plot4, n
col = 4)
```



Each cluster group has a similar way of writing letter W. For example, all observations in cluster 2 has a straight letter W, and the way W is written is sharp. Cluster 3 has another way for writing letter W that it looks softer than cluster 2. But cluster 1 appears to be different since not all observations have a similar look, and that could not tell us much about the majority of cluster 1. Cluster 4 only has only one observation and it also doesnt tell us much about the way of writing letter W.

Report

In this report, I have tried 3 different supervised learning method to figure out the best model to classify these 5 observations into clusters. The models that I have builts are: Random forest, k-nearest neighbor and XGB.

Read the newrecord data

Code

Doing PCA on the newrecord data

Code

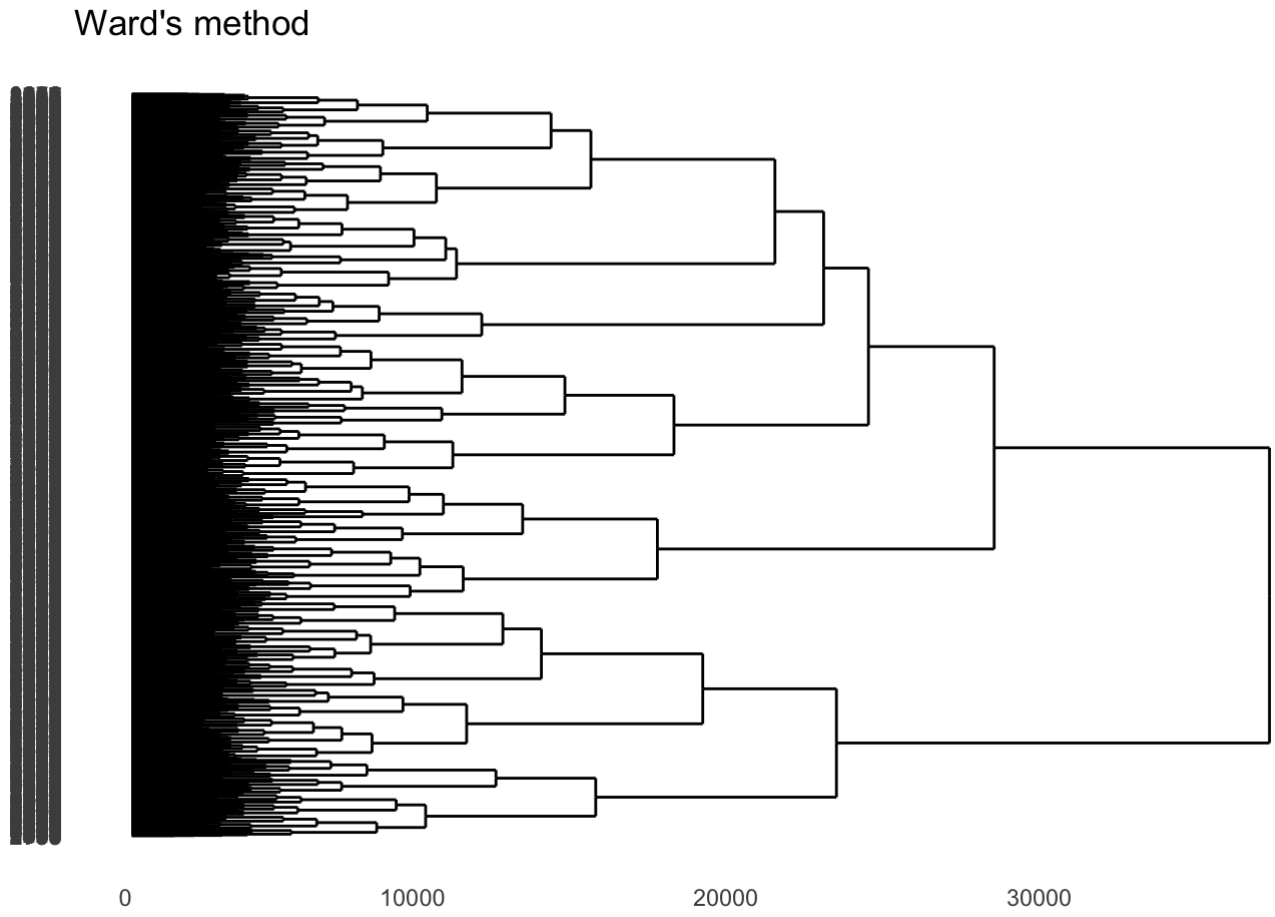
Code

Code

Draw a dendogram to build a tree of the main data

Hide

```
# agglomerative hierarchical clustering with Ward's method to split the main data
hward <- hclust(dist(data_pca$x), method = "ward.D2")
ggdendrogram(hward, rotate = TRUE) + labs(title = "Ward's method")
```



I plot a dendogram to visualize the tree so I could decide how many clusters that I want. It is quite sensible to cut the tree into 7 when we look at the dendogram since some of these 7 trees have quite similar number of smaller tree. If we have more trees, it wont be sensible anymore when looking at the dendogram.

Cut tree to divide clusters of the main data

Hide

```
cward <- cutree(hward, k = 7)
table(cward)
```

```
## cward
##   1   2   3   4   5   6   7
## 359 555 627 350 568 726 233
```

As we look at the table, the number of observations in each group are quite similar. Having too much clusters would be difficult to figure out the way of writing letter W, since maybe a slighly difference in the letters could make them being in a different group, which is not interpretable.

Plots the clusters of the main data

Hide

```
# show the sample images of 10 clusters from the main data
set.seed(1707)
data_labels <- data %>%
  mutate(c_ward= cward) %>%
  imagedata_to_plotdata(which = sample(1:3418, 3418))

plots <- list()

for (i in 1:7) {
  cls <- data_labels %>%
  filter(c_ward == i)

  ids <- head(unique(cls$id), 10)

  cls <- cls %>%
    filter(id %in% ids)

cls_plot <- cls %>%
  ggplot(aes(col, row)) +
    geom_tile(aes(fill = value)) +
    facet_wrap(~id, nrow = 10) +
    scale_y_reverse() +
    theme_void(base_size = 4) +
    guides(fill = "none") +
    coord_equal() +
    ggtitle(paste("Cluster", i)) +
    theme(strip.background = element_blank(), strip.text.x = element_blank(),  plot.t
itle = element_text(hjust = 0.5))
plots[[i]] <- cls_plot
}
grid.arrange(plots[[1]],plots[[2]],plots[[3]],plots[[4]],plots[[5]],plots[[6]],plots
[[7]],
             ncol = 7)
```

If we look at the plots, we could see that most of observations that look similar to each other belong to one cluster. Cluster 1: all the observations look straight and sharp, and look like these letters have been written carefully. Cluster 2: all letters in this cluster look a bit sharp, and they fall a little bit to the right. Cluster 3: these obsevations tend to fall to the right, and they look smaller than other groups. Cluster 4: Letter W in this group have been written like 2 letter U, they look round and soft. Cluster 5: These observations are quite large compared to other groups, and it look sharp but not really straight. Cluster 6: These letters are written carelessly, they fall to the right just a bit. Cluster 7: all letters in this group look soft and straight, and written carefully.

Convert data_pca into a data frame

Hide

```
# convert data_pca$x into a data frame
data_pca1 <- data.frame(data_pca$x)
```

Add column 'cluster' into the data frame

Hide

```
data_pca1 <- data_pca1 %>% # add the cluster column into data frame
  mutate(cluster = as.factor(cward))
```

Create training set and testing set to make and evaluate model

Hide

```
set.seed(1707)
# create the testing and training set from the main data
data_split <- initial_split(data_pca1)
train_data <- training(data_split)
test_data <- testing(data_split)
```

Random forest model

Hide

```
# Random forest model
set.seed(1707)
rf_model <- ranger(cluster ~ .,
                   data = train_data,
                   mtry = floor((ncol(train_data) - 1)/3),
                   importance = "impurity",
                  replace = FALSE,
                   classification = TRUE)
rf_pred <- predict(rf_model, test_data)
```

Hide

```
accuracy_rf <- accuracy_vec(truth = test_data$cluster, estimate = rf_pred$prediction
s) # check the accuracy of the model
accuracy_rf
```

```
## [1] 0.7450292
```

Hide

```
# Check the balance accuracy
bal_accuracy_rf <- bal_accuracy_vec(truth = test_data$cluster, estimate = rf_pred$pre
dictions)
bal_accuracy_rf
```

```
## [1] 0.8497026
```

The accuracy of this model is 74.5% and balance accuracy is approximately 85%.

k-nearest neighbor model

Hide

```
# k-nearest neighbor model
set.seed(1707)
knn_pred <- kknn(cluster ~ .,
                 train = train_data,
                 test = test_data,
                  k = 30,
                 # parameter of Minkowski distance
                 # 2 = Euclidean distance
                 # 1 = Manhattan distance
                 distance = 2)
```

Hide

```
accuracy_knn <- accuracy_vec(truth = test_data$cluster, estimate = as.factor(knn_pred
$fitted.values))
accuracy_knn
```

```
## [1] 0.4736842
```

Hide

```
bal_accuracy_knn <- bal_accuracy_vec(truth = test_data$cluster, estimate = as.factor
(knn_pred$fitted.values))
bal_accuracy_knn
```

```
## [1] 0.6450336
```

The accuracy of this model is 47% and balance accuracy is approximately 64.5%. This is not a good model since the accuracy rate is really low (less than 50%), therefore it is not realiable.

XGB model

Hide

```
# XGB model
set.seed(2000)
xgb_model <- xgboost(data = model.matrix(~ . - cluster, data = train_data)[, -1],
                     label = train_data$cluster,
                     max.depth = 2,
                     eta = 0.3,
                     nrounds = 50,
                     num_class = 8,
                     objective = "multi:softprob",
                     verbose = 0)
xgb_pred <- predict(xgb_model, model.matrix(~. - cluster, data = test_data)[, -1])
xgb_pred <- matrix(xgb_pred, ncol = 8, byrow = TRUE)
xgb_pred <- max.col(xgb_pred) - 1
```

Hide

```
bal_accuracy_xgb <- bal_accuracy_vec(truth = test_data$cluster, estimate = as.factor
(xgb_pred))
bal_accuracy_xgb
```

```
## [1] 0.8497035
```

Hide

```
accuracy_xgb <- accuracy_vec(truth = test_data$cluster, estimate = as.factor(xgb_pre
d))
accuracy_xgb
```

```
## [1] 0.7438596
```

The accuracy of this model is 74% and balance accuracy is approximately 85%.

Evaluate models

As we can see from the accuracy and balance accuracy rate of 3 models, it appears that kNN performs worst. It could not predict well with the data. So we could not use that one for predicting newrecord data.

Random forest model and XGB model have similar performance since the accuracy rate and balance accuracy rate of them are quite similar (about 85% for the balance accuracy). Therefore, we could use these two models to predict the new record data.

Moreover, 85% for a model is a quite sensible rate when it comes to prediction. Not all model could predict 100% correctly, and it also depends on the data and how the clusters are divided.

Fit the model with whole main data

Hide

```
set.seed(1707)
data_pca2 <- data_pca1 %>%
  select(c(PC1:PC5), cluster)
xgb_model1 <- xgboost(data = model.matrix(~ . - cluster, data = data_pca2)[, -1],
                      label = data_pca2$cluster,
                      max.depth = 2,
                      eta = 0.3,
                      nrounds = 50,
                      num_class = 8,
                      objective = "multi:softprob",
                      verbose = 0)
```

Predict the model using newrecord data as the testing set.

Hide

```
set.seed(1707)
newrecord_pca2 <- newrecord_pca1
xgb_pred1 <- predict(xgb_model1, model.matrix(~. - cluster, data = newrecord_pca1)[,
-1])
xgb_pred1 <- matrix(xgb_pred1, ncol = 8, byrow=TRUE)
xgb_pred1 <- max.col(xgb_pred1) -1
```