Elif Küçük
0040851
COMP 421
HW 06 Report.

COMP/INDR 421/521 HW 06: LINEAR DISCRIMINANT ANALYSIS– Report

This file briefly explains each function and variable defined in the project. The defined function names and variables are written in bold. The code also has the comment lines to make understating easier.

Question 1

Using the read.csv function we read the given training and testing digits and labels into **training_digits, testing_digits, training_labels** and **test_labels**.

In order to normalize training_digits and testing_digits, in **X_train** and **X_test** we are taking the matrix form of training_digits and testing_digits divided by 255.

y_train and **y_labels** are holding the corresponding labels for each sample data.

Question 2

To implement linear discriminant analysis I defined a function called **linear_disc_analysis(X,y,R)** which takes input as normalized training_digits X, corresponding training_labels y and dimension to return W that will be used to reduce dimensions of X.

Linear discriminant analysis converts N*D matrice to N*K matrice using weight variable W which is of dimension D*K.

$$Z = W^T X$$

In linear discriminant analysis because we want to be able to distinguish the classes from each other; we want to take the maximum value of between-class scatter matrix after projection. And because we want to define area that is little for each class; we want to take minimum value of within class scatter matrix after projection.

Therefore our objective function is :

$$J(W) = \frac{abs(W^T S_B W)}{abs(W^T S_W W)}$$

$$taking\ the\ derivative\ of\ J(W) wrt\ W, we\ find\ that\ W = the\ largest\ eigenvector\ of\ S_W^{-1} S_B$$

In order to find S_w and S_B we are defining and calculating the following variables.

N: # of training data.
D: Dimension of each training data. (284 for our X_training)
K: Number of classes.

Elif Küçük
0040851
COMP 421
HW 06 Report.
N_i: K*1 column vector specifying the number of sample points belonging to class i.

m_before: means of sample belonging to each class. It's dimension is K*D.

$$m_{before[i,]} = \frac{\sum_t x^t r_i^t}{\sum_t r^t}, r^t = 1 \; if \; x^t \in class \; i$$

add: Helper method I used to add the list elements one by one in R.
S: A list that contains within-class scatter matrix of each class. The length of list is K (class number) and the dimension of each member is (D*D).

$$S\_i = \sum_t r_i^t (x^t - m_i)(x^t - m_i)^T \; where \; m_i is \; m\_before[i,]$$

S_w: The total within class scatter. The dimension is D*D. Using the add method, I add the members S.

$$S\_w = \sum_{i=1}^{K} S_i$$

overall_mean: Mean of each dimension in overall training_digits.

S_b: Between class scatter matrix. It's dimension is also D*D.

$$S\_b = \sum_i N_i (m_i - m)(m_i - m)^T \; where \; m_i is \; m_{before[i,]} and \; m \; is \; overall\_mean$$

S_w_in: In order to make S_w matrice invertible I add 1e-10 to diagonal of S_w matrix and using the chol and chol2inv functions I take the inverse of S_w.

```
A<-diag(rep(1e-10,784))
S_w<-S_w+A
S_w_in<-chol2inv(chol(S_w))
```

val: S_w_in * S_B.

decomposition: Using the eigen built-in function I take the eigen values and eigen vectors.

W: The largest R eigen vectors of decomposition. This is returned by the linear_disc_analysis method.

Question 3:

Elif Küçük
0040851
COMP 421
HW 06 Report.
Using the linear_disc_analysis function defined in question 2 and giving the X_train, y_train and 2 (R value) we learn W matrix.

Using this W matrix we find Z_train.

$$Z = W^T X$$

```
W<-linear_disc_analysis(X_train,y_train, 2)
Z_train <- (X_train) %*% W
point_colors <- c("#1f78b4", "#33a02c", "#e31a1c", "#ff7f00", "#6a3d9a", "#a6cee3",
"#b2df8a", "#fb9a99", "#fdbf6f", "#cab2d6")
plot(Z_train[,1], Z_train[,2], type = "p", pch = 19, col = point_colors[y_train], cex = 0,
    xlab = "PC1", ylab = "PC2", las = 1)
text(Z_train[,1], Z_train[,2], labels = y_train %% 10, col = point_colors[y_train])
```

Plotting two dimensional projections of the training data set gives the Figure 1.

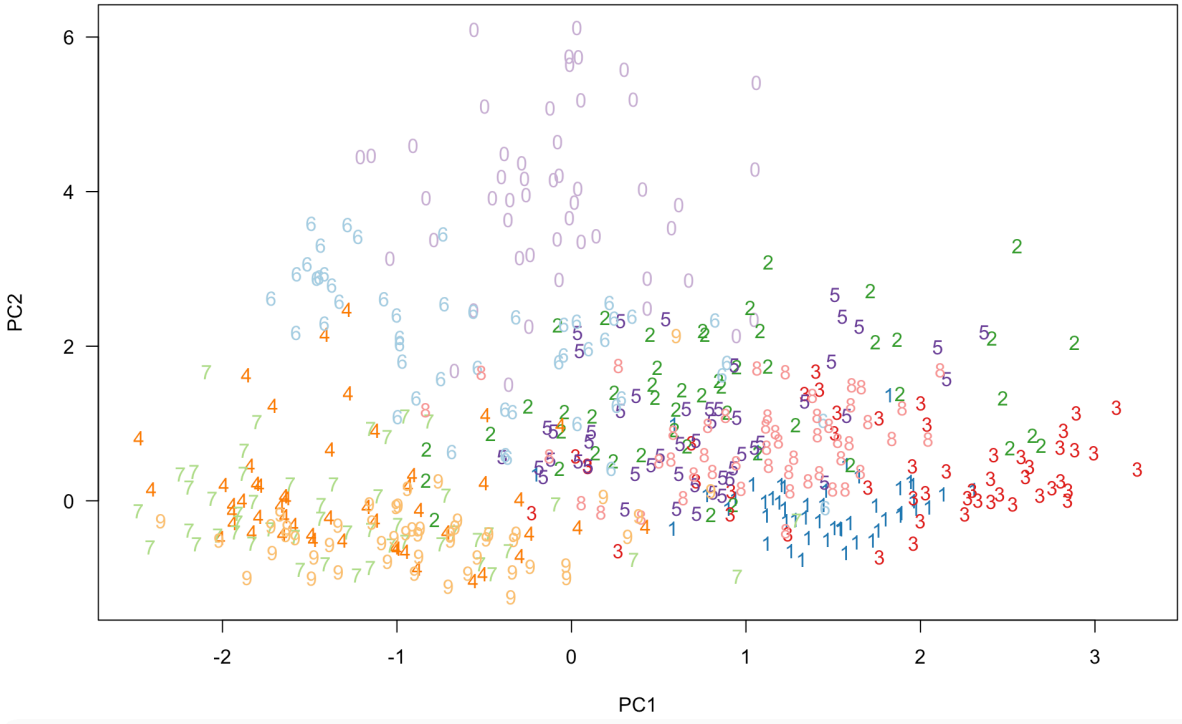

Figure 1

Using W matrix learned from training data set on X_test data points and plotting two dimensional projection of test data we get the Figure 2.

Elif Küçük
0040851
COMP 421
HW 06 Report.

```
Z_test <- (X_test) %*% W
# plot two-dimensional projections for test set
plot(Z_test[,1], Z_test[,2], type = "p", pch = 19, col = point_colors[y_test], cex = 0,
    xlab = "PC1", ylab = "PC2", las = 1)
text(Z_test[,1], Z_test[,2], labels = y_test %% 10, col = point_colors[y_test])
```

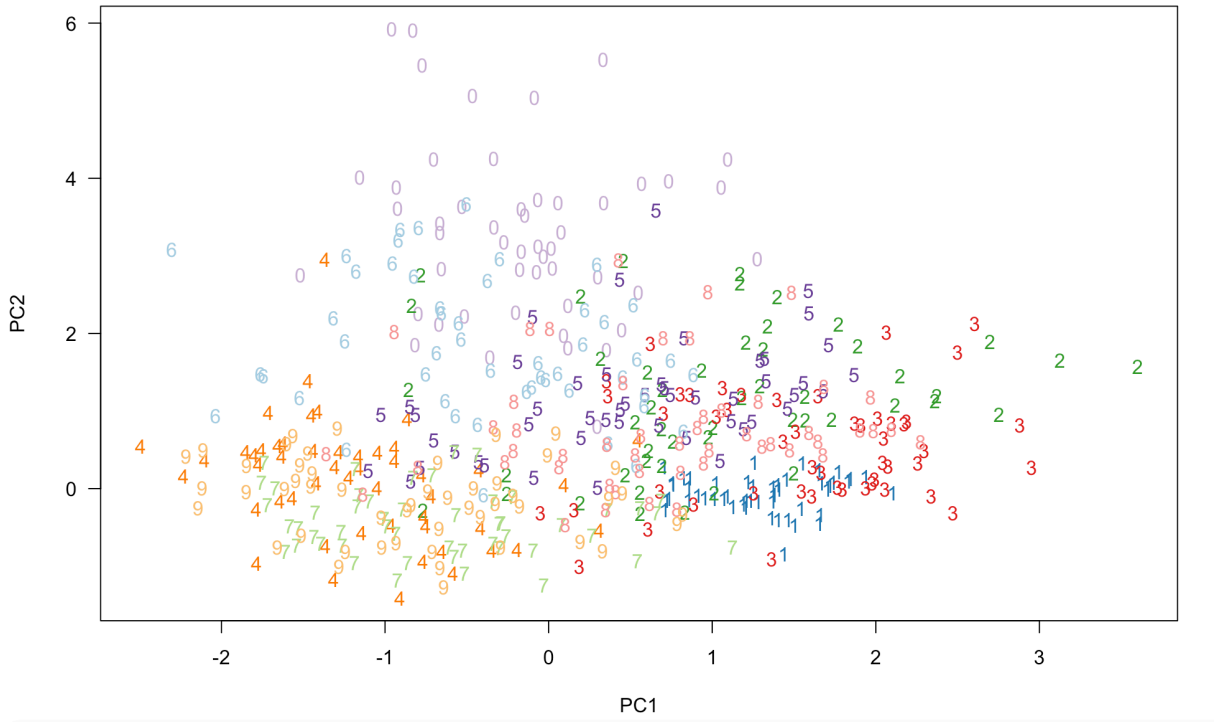

Figure 2

Question #4:

To learn 5-nearest neighbour classifier of low dimensional projections of training and test data points I defined following functions.

euclidian: This function takes an input that is a data point (x) and a vector input that is all data points together (y) all of which we will find the distance from x. For example if y has 4 data points euclidian method returns a vector that has 4 members which are distance from x to each data point.

$$\sum_{i=1}^{D}\big(x(i)-y(,i)\big)^2, where\ i = 1,2,\dots D\ and\ D\ is\ the\ dimension\ of\ x.$$

Elif Küçük
0040851
COMP 421
HW 06 Report.
find_KNN(x,X,Y): This method finds the nearest 5 neighbors of x in data set X; and returns the value of most occurred label value in its closest 5 neighbors. Y stores the label set for data set X.

In order draw classification accuracy on the projections of test points I perform the following code.

```
R<-seq(1:9)
accuracy<-c()
for(r in R ){
  W<-linear_disc_analysis(X_train,y_train, r)
  Z_test <- (X_test) %*% W
  Z_train<-(X_train) %*% W
  p_head <- apply(Z_test, 1, find_KNN, Z_train, y_train)
  accuracy<-cbind(accuracy, sum(p_head==y_test)/N)
}
plot(R, accuracy*100, type="o", pch=19, ylab="Classification Accuracy %", xlab="R")
```

The resulting graph is in the Figure 3.

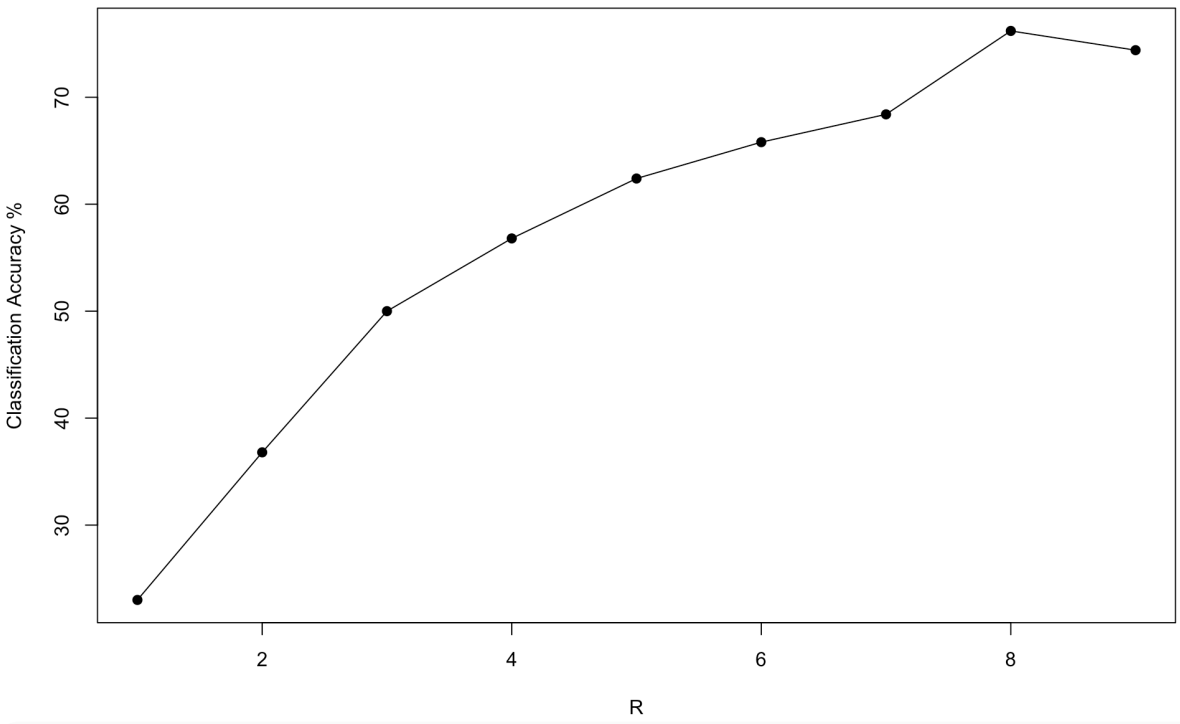

Figure 3