154HW5

Jiyoon Clover Jeong

10/26/2017

```r
wine <- read.table("/Users/cloverjiyoon/2017Fall/Stat 154/data/wine.data.txt", sep = ",", header =  T)
wine$class <- as.factor(wine$class)

names(wine)
```

```
##  [1] "class"           "alcohol"         "malic"
##  [4] "ash"             "alcalinity"      "magnesium"
##  [7] "phenols"         "flavanoids"      "nonflavanoids"
## [10] "proanthocyanins" "color"           "hue"
## [13] "dilution"        "proline"
```

1) Sum-of-Squares Dispersion Functions (10 pts)

```r
tss <- function(x) {

  sum((x - mean(x))^2)
}

cat("TSS : ", tss(iris$Sepal.Length), "\n")
```

```
## TSS :  102.1683
```

```r
bss <- function(x, y){


  y <- as.factor(as.character(y))

  if(length(x) != length(y))
    stop("x and y have different lengths")


  sum <- 0
  x_bar <-  mean(x)       #tss(x)
  splited <- split(x, y)

  X_k <- sapply(splited, mean)

  for(i in 1: length(splited)){
    sum <- sum + length(splited[[i]]) * (X_k[i] - x_bar)^2
  }

  return(sum)

}

cat("BSS : ", bss(iris$Sepal.Length, iris$Species), "\n")
```

```
## BSS :  63.21213
```

```r
wss <- function(x,y){

  y <- as.factor(as.character(y))

  if(length(x) != length(y))
    stop("x and y have different lengths")

  sum <- 0
  x_bar <-  mean(x)      #tss(x)
  splited <- split(x, y)
  X_k <- sapply(splited, mean)

  for(i in 1: length(splited)){
    for(j in 1: length(splited[[i]])){
       sum <- sum +  (splited[[i]][j] - X_k[i])^2
    }

  }

  return(sum)

}


cat("WSS : ", wss(iris$Sepal.Length, iris$Species), "\n")
```

```
## WSS :  38.9562
```

2) Sum-of-Squares Ratio Functions (10 pts)

```r
cor_ratio <- function(x, y){
  BSS <- bss(x,y)
  TSS <- tss(x)

  cor <- BSS / TSS
  return (as.numeric(cor))
}

cor_ratio(iris$Sepal.Length, iris$Species)
```

```
## [1] 0.6187057
```

```r
F_ratio <- function(x,y){

  y <- as.factor(as.character(y))
  BSS <- bss(x,y)
  WSS <- wss(x,y)

  n <- length(x)
  k <- nlevels(y)
```

```
  Fratio <- (BSS / (k - 1)) / (WSS / (n - k))
  return(as.numeric(Fratio))
}



F_ratio(iris$Sepal.Length, iris$Species)
```

```
## [1] 119.2645
```

3) Discriminant Power of Predictors (30 pts)

```
# The first approach consists of running simple logistic regressions
# Q : For  number 3, consider only wine classes 1 and 2 (ignore class 3)

predictors <- names(wine[1:130,-1])
formula <- c("class ~")
AIC <- data.frame(name = predictors, AIC = rep(0,13))

for(i in 1: length(predictors)){

  formula <- paste("class ~", predictors[i])
  #fit <- multinom(formula, data = wine[1:130,])
  fit <- glm(formula, data = wine[1:130,] , family = binomial)
  AIC[i,2] <- fit$aic


}

AIC
```

```
##               name       AIC
## 1          alcohol  56.30075
## 2            malic 182.85454
## 3              ash 165.30370
## 4       alcalinity 148.51462
## 5        magnesium 162.10222
## 6          phenols 139.62520
## 7       flavanoids 121.51589
## 8    nonflavanoids 166.94370
## 9  proanthocyanins 174.71983
## 10           color  81.96971
## 11             hue 183.07125
## 12        dilution 161.00793
## 13         proline  45.21948
```

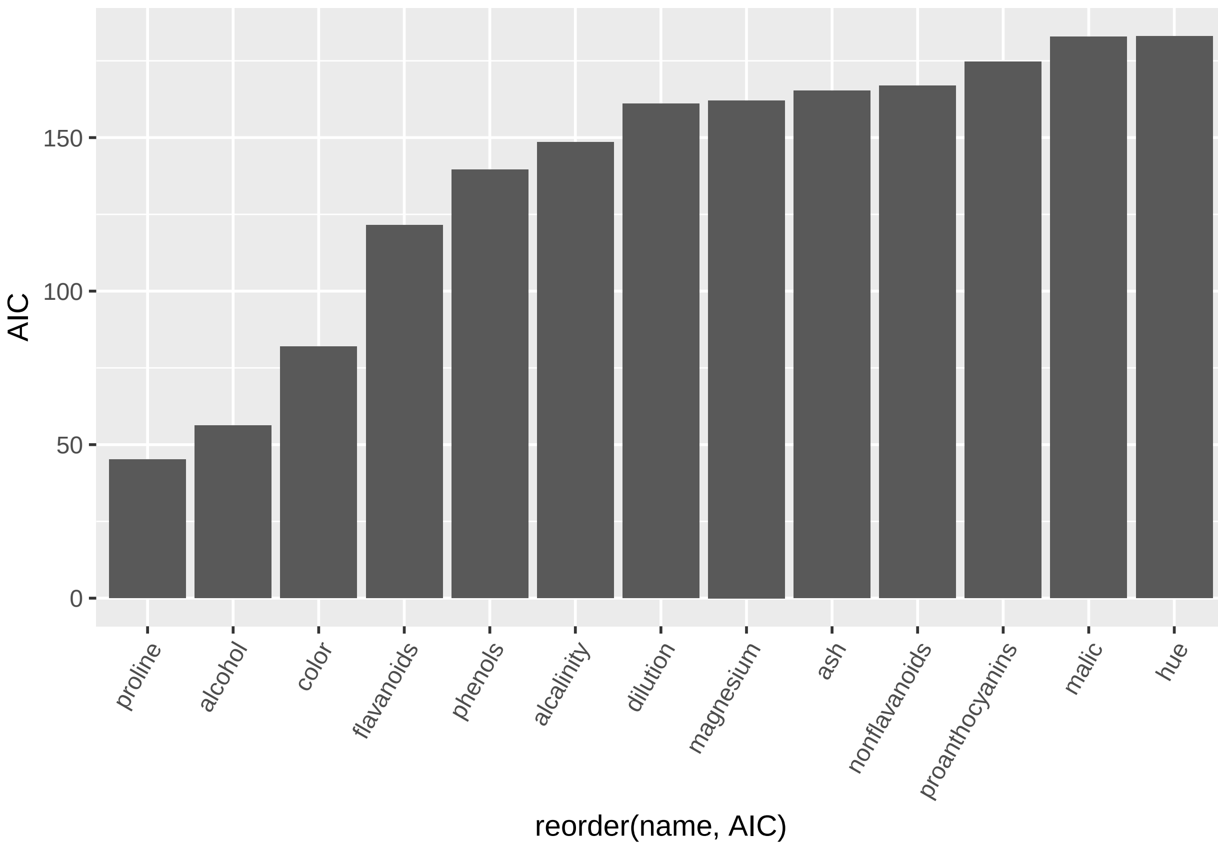
```
AIC <- AIC[ order(AIC[,2]), ]

AIC
```

```
##               name       AIC
## 13         proline  45.21948
## 1          alcohol  56.30075
```

```
## 10            color  81.96971
## 7        flavanoids 121.51589
## 6           phenols 139.62520
## 4        alcalinity 148.51462
## 12         dilution 161.00793
## 5         magnesium 162.10222
## 3               ash 165.30370
## 8    nonflavanoids 166.94370
## 9  proanthocyanins 174.71983
## 2             malic 182.85454
## 11              hue 183.07125
```

```
ggplot(AIC, aes( x = reorder(name, AIC), y =  AIC)) + geom_col() + theme(axis.text.x = element_text(angl
```




```
# The second approach consists of computing correlation ratios

corratio <- data.frame(name = predictors, cor_ratio = rep(0,13))

for(i in 1:length(predictors)){
  corratio[i,2] <- cor_ratio(wine[1:130,i+1], wine[1:130,1])
}

corratio
```

```
##              name    cor_ratio
## 1         alcohol 0.6796337087
## 2           malic 0.0019626987
## 3             ash 0.1257044543
```

```
## 4       alcalinity 0.2213110717
## 5        magnesium 0.1467544634
## 6          phenols 0.2837609465
## 7       flavanoids 0.3729916215
## 8    nonflavanoids 0.1138987546
## 9  proanthocyanins 0.0621031600
## 10           color 0.5634196215
## 11             hue 0.0002904483
## 12        dilution 0.1534649761
## 13         proline 0.7145258216
```

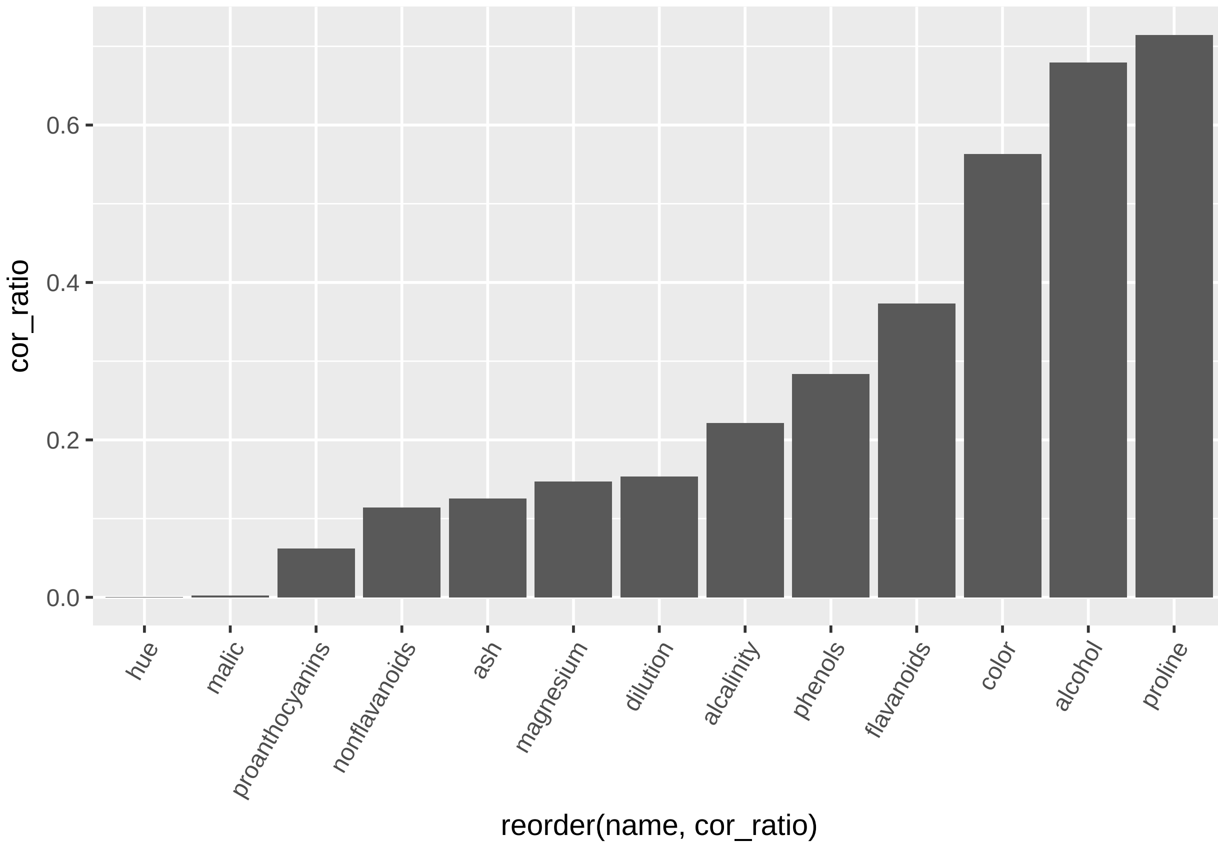
```r
corratio <- corratio[ order( corratio[,2]), ]

corratio
```

```
##               name    cor_ratio
## 11             hue 0.0002904483
## 2            malic 0.0019626987
## 9  proanthocyanins 0.0621031600
## 8    nonflavanoids 0.1138987546
## 3              ash 0.1257044543
## 5        magnesium 0.1467544634
## 12        dilution 0.1534649761
## 4       alcalinity 0.2213110717
## 6          phenols 0.2837609465
## 7       flavanoids 0.3729916215
## 10           color 0.5634196215
## 1          alcohol 0.6796337087
## 13         proline 0.7145258216
```

```r
# Q : automatic ordering??
#ggplot(corratio, aes(x = name, y = cor_ratio)) + geom_col()

ggplot(corratio, aes(x = reorder(name, cor_ratio), y = cor_ratio)) + geom_bar(stat = "identity") + theme
```




```r
# Q : automatic ordering??
#ggplot(corratio, aes(x = name, y = cor_ratio)) + geom_bar(stat = "identity")




# The third approach consists of computing F-ratios

Fratio <- data.frame(name = predictors, Fratio = rep(0,13))

for(i in 1:length(predictors)){
  Fratio[i,2] <- F_ratio(wine[1:130,i+1], wine[1:130,1])
}

Fratio
```

```
##               name       Fratio
## 1          alcohol 271.54265938
## 2            malic   0.25171949
## 3              ash  18.40358243
## 4       alcalinity  36.37886215
## 5        magnesium  22.01543461
## 6          phenols  50.71128273
## 7       flavanoids  76.14400251
## 8    nonflavanoids  16.45301895
## 9  proanthocyanins   8.47556377
## 10           color 165.18770681
```

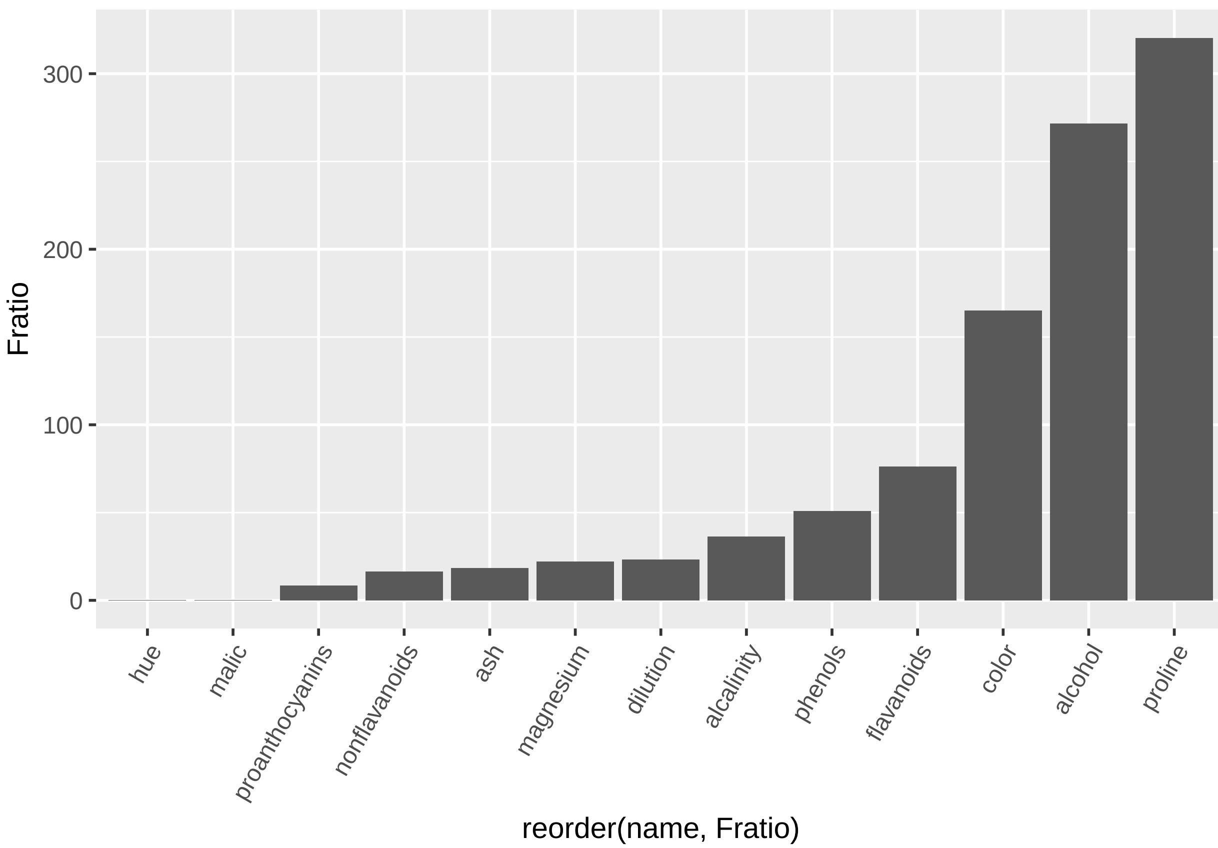
```
## 11             hue   0.03718818
## 12        dilution  23.20461220
## 13         proline 320.37680494
```

```
Fratio <- Fratio[ order( Fratio[,2]), ]

Fratio
```

```
##               name       Fratio
## 11             hue   0.03718818
## 2            malic   0.25171949
## 9  proanthocyanins   8.47556377
## 8    nonflavanoids  16.45301895
## 3              ash  18.40358243
## 5        magnesium  22.01543461
## 12        dilution  23.20461220
## 4       alcalinity  36.37886215
## 6          phenols  50.71128273
## 7       flavanoids  76.14400251
## 10           color 165.18770681
## 1          alcohol 271.54265938
## 13         proline 320.37680494
```

```
ggplot(Fratio, aes(x = reorder(name, Fratio), y = Fratio)) + geom_bar(stat = "identity") + theme(axis.te
```



4) Variance functions

```
total_variance <- function(X){
  X <- as.matrix(X)
  X <- scale(X, scale = F)
  n = dim(X)[1]
  V <- (1/ (n-1)) * t(X) %*% X
  return(V)

}

# test total_variance()
total_variance(iris[ ,1:4])
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    0.6856935  -0.0424340    1.2743154   0.5162707
## Sepal.Width    -0.0424340   0.1899794   -0.3296564  -0.1216394
## Petal.Length    1.2743154  -0.3296564    3.1162779   1.2956094
## Petal.Width     0.5162707  -0.1216394    1.2956094   0.5810063
```

```
# compare with var()
var(iris[ ,1:4])
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    0.6856935  -0.0424340    1.2743154   0.5162707
## Sepal.Width    -0.0424340   0.1899794   -0.3296564  -0.1216394
## Petal.Length    1.2743154  -0.3296564    3.1162779   1.2956094
## Petal.Width     0.5162707  -0.1216394    1.2956094   0.5810063
```

```
# PPT 25- p64

between_variance <- function(X, y){

  X <- scale(X, scale = F)
  n <- length(y)
  y <- as.data.frame( y)
  Y <- model.matrix(~ y -1, data =  y)
  BSS <- t(X) %*% Y %*% solve(t(Y) %*% Y)  %*% t(Y)  %*% X
  B <- BSS / (n - 1)

  return(B)

}


# test between_variance()
between_variance(iris[ ,1:4], iris$Species)
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    0.4242425 -0.13391051    1.1090497   0.4783848
## Sepal.Width    -0.1339105  0.07614049   -0.3841584  -0.1539105
## Petal.Length    1.1090497 -0.38415839    2.9335758   1.2535168
## Petal.Width     0.4783848 -0.15391051    1.2535168   0.5396868
```

```r
# PPT 25- p64

within_variance <- function(X, y){

  # X <- as.data.frame(scale(X, scale =F))
  # #X <- scale(X, scale = F)
  # n <- dim(X)[1]
  # p <- dim(X)[2]
  # XX <- data.frame(X,y)
  # X_k <-  split(X, y)
  # GSS_k <- list()
  # W_k <- list()
  # W <- matrix(0, p, p)
  # for(i in 1:nlevels(y)){
  #   X_k[[i]] <- as.matrix(X_k[[i]])
  #   GSS_k[[i]] <- t(X_k[[i]]) %*% X_k[[i]]
  #   n_k <- dim(X_k[[i]])[1]
  #   W_k <- (1 / (n_k -1)) *  GSS_k[[i]]
  #   W <- W + ( (n_k - 1) / (n - 1)) * W_k
  # }

  X <- scale(X, scale = F)
  n <- length(y)
  y <- as.data.frame(y)
  Y <- model.matrix(~ y -1, data =  y)
  WSS <- t(X) %*% (diag(n) - Y %*% solve(t(Y) %*% Y) %*% t(Y)) %*% X
  W <- WSS / (n - 1)


  return(W)
}


# test within_variance()
within_variance(iris[ ,1:4], iris$Species)
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length   0.26145101  0.09147651   0.16526577  0.03788591
## Sepal.Width    0.09147651  0.11383893   0.05450201  0.03227114
## Petal.Length   0.16526577  0.05450201   0.18270201  0.04209262
## Petal.Width    0.03788591  0.03227114   0.04209262  0.04131946
```

```r
# Confirm that V = B +W

# confirm V = B + W
Viris <- total_variance(iris[ ,1:4])
Viris
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    0.6856935  -0.0424340    1.2743154   0.5162707
## Sepal.Width    -0.0424340   0.1899794   -0.3296564  -0.1216394
## Petal.Length    1.2743154  -0.3296564    3.1162779   1.2956094
## Petal.Width     0.5162707  -0.1216394    1.2956094   0.5810063
```

```
# B + W
Biris <- between_variance(iris[ ,1:4], iris$Species)
Wiris <- within_variance(iris[ ,1:4], iris$Species)
Biris + Wiris
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    0.6856935  -0.0424340    1.2743154   0.5162707
## Sepal.Width    -0.0424340   0.1899794   -0.3296564  -0.1216394
## Petal.Length    1.2743154  -0.3296564    3.1162779   1.2956094
## Petal.Width     0.5162707  -0.1216394    1.2956094   0.5810063
```

Challenge

find the eigenvectors uk

```
# find the eigenvectors uk

X <- wine[,2:14]
y <- as.factor(wine[,1])

W <- within_variance(X, y)    # X should be dataframe

K <- nlevels(y)
J <- dim(X)[2]
n <- dim(X)[1]

splited <- split(X, y)

C <- matrix(0, J, K)

for(j in 1: J){
  for(k in 1: K){

    n_k <- dim(splited[[k]])[1]
    n_k
    Xbar_jk <- mean(splited[[k]][, j])
    Xbar_jk
    Xbar_j <- mean(X[, j])
    Xbar_j
    C[j,k] <- sqrt(n_k / (n-1)) *  (Xbar_jk - Xbar_j)
    C[j,k]

  }
}

C
```

```
##              [,1]          [,2]        [,3]
##  [1,]  0.42962238 -4.572049e-01  0.07974436
##  [2,] -0.18802586 -2.556651e-01  0.51940256
##  [3,]  0.05142826 -7.709629e-02  0.03674789
##  [4,] -1.41892817  4.706311e-01  1.00074802
```

```
##  [5,]   3.80901645 -3.288519e+00  -0.22344220
##  [6,]   0.31468888 -2.295198e-02  -0.32097418
##  [7,]   0.55027440  3.266519e-02  -0.64980479
##  [8,]  -0.04148489  1.145118e-03   0.04460067
##  [9,]   0.17806819  2.494303e-02  -0.22775624
## [10,]   0.27147887 -1.248627e+00   1.21761007
## [11,]   0.06038187  6.259523e-02  -0.14307298
## [12,]   0.31529746  1.099915e-01  -0.48333608
## [13,] 212.93752144 -1.440147e+02 -60.92706944
```

```r
# check

B <- between_variance(X,y)


head(B,2)
```

```
##            alcohol      malic        ash alcalinity   magnesium    phenols
## alcohol 0.39997090 0.07753065 0.06027397 -0.7449742 3.122147772  0.1200953
## malic   0.07753065 0.37049738 0.02912794  0.6662623 0.008509593 -0.2200164
##         flavanoids nonflavanoids proanthocyanins     color         hue
## alcohol  0.1696572   -0.01478974      0.04693573 0.7846094 -0.01408671
## malic   -0.4493274    0.03067317     -0.15815566 0.9006151 -0.10166924
##            dilution   proline
## alcohol  0.04662686 152.46834
## malic   -0.33845106 -34.86392
```

```r
head(C %*% t(C), 2)
```

```
##            [,1]       [,2]       [,3]       [,4]        [,5]       [,6]
## [1,] 0.39997090 0.07753065 0.06027397 -0.7449742 3.122147772  0.1200953
## [2,] 0.07753065 0.37049738 0.02912794  0.6662623 0.008509593 -0.2200164
##            [,7]        [,8]        [,9]     [,10]       [,11]       [,12]
## [1,]  0.1696572 -0.01478974  0.04693573 0.7846094 -0.01408671  0.04662686
## [2,] -0.4493274  0.03067317 -0.15815566 0.9006151 -0.10166924 -0.33845106
##          [,13]
## [1,] 152.46834
## [2,] -34.86392
```

```r
dim(B)
```

```
## [1] 13 13
```

```r
dim(C %*% t(C))
```

```
## [1] 13 13
```

```r
eigen_w <- eigen( t(C) %*% solve(W) %*% C )$vectors

eigen_values <- eigen( t(C) %*% solve(W) %*% C )$values

eigen_values
```

```
## [1] 9.081739e+00 4.128469e+00 1.776357e-15
```

```r
eigen_u <- solve(W) %*% C %*% eigen_w

eigen_u
```

```
##                          [,1]          [,2]          [,3]
## alcohol           1.222609554  1.7814577940  1.665335e-15
## malic            -0.500847689  0.6240254979 -1.165734e-15
## ash               1.118580020  4.7936058021 -1.032507e-14
## alcalinity       -0.469155877 -0.2991204731  6.383782e-16
## magnesium         0.006557047 -0.0009456156  9.107298e-18
## phenols          -1.873169990 -0.0658249947 -1.110223e-16
## flavanoids        5.034678679 -1.0053690476  2.664535e-15
## nonflavanoids     4.533472756 -3.3327580255  9.325873e-15
## proanthocyanins -0.406403118 -0.6275153789  1.464107e-15
## color            -1.076090082  0.5174619938 -1.665335e-15
## hue               2.479274325 -3.0971098347 -1.998401e-15
## dilution          3.508289345  0.1045914210 -1.998401e-15
## proline           0.008156412  0.0058299061 -3.686287e-18
```

Obtain the linear combinations zk and make a scatterplot of the wines

```r
X <- as.matrix(X)

Z <- X %*% eigen_u
head(Z)
```

```
##          [,1]     [,2]          [,3]
## [1,] 42.22167 33.96474 -1.694088e-15
## [2,] 41.01456 32.31215 -2.826645e-15
## [3,] 38.34373 32.84077 -2.596109e-15
## [4,] 40.72298 38.10012 -5.157801e-15
## [5,] 32.55273 30.84253 -8.013989e-16
## [6,] 41.67142 36.48634 -2.632079e-15
```

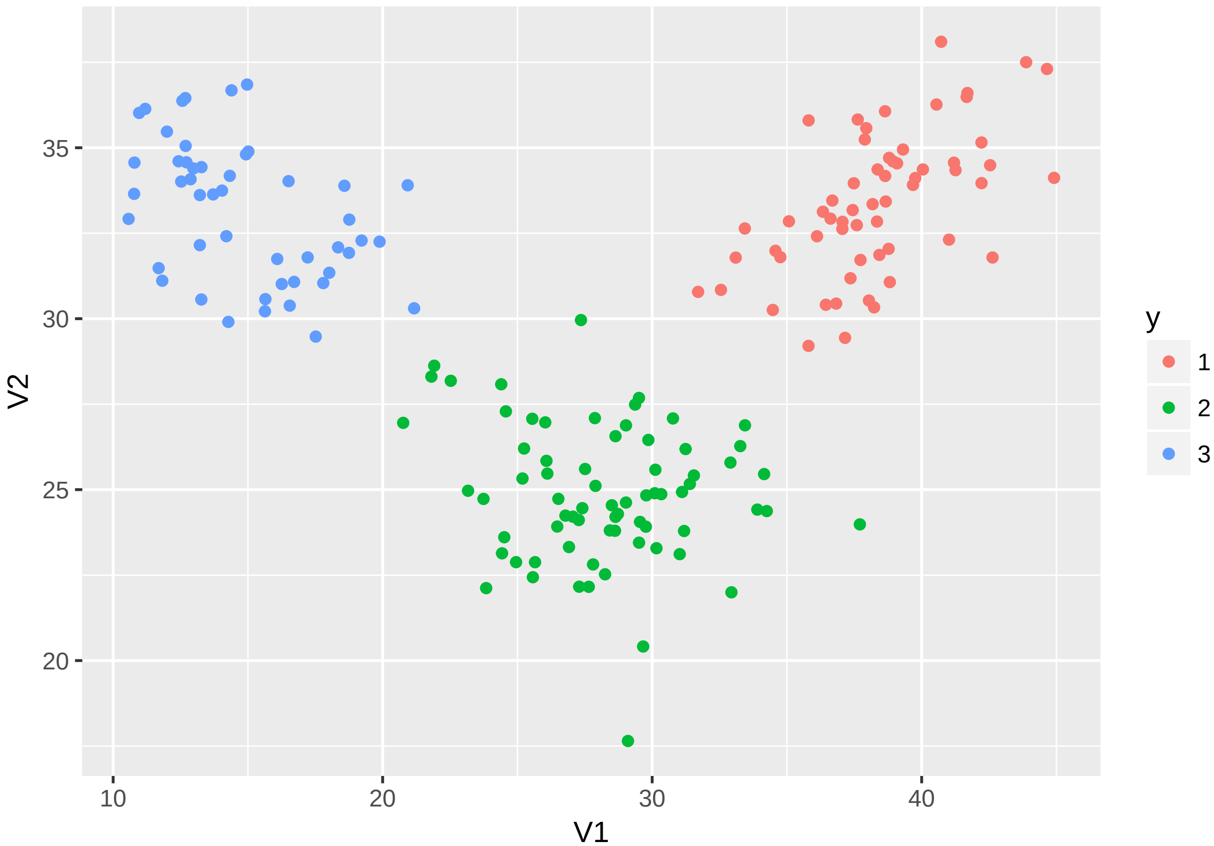
```r
Z_splited <- split(as.data.frame(Z), y)


# for(i in 1:nlevels(y)){
#   z_k <- c(Z_splited[[1]][, i], Z_splited[[2]][ ,i],  Z_splited[[3]][ ,i])
#   Z_k <- data.frame(z_k, y)
#   print(ggplot(Z_k, aes(x = z_k, y = y, color = y)) + geom_point() +
#           labs(title = paste0("linear combinations Z_", i), y = "class", color = "class"))
#
# }




  z_k <- rbind(Z_splited[[1]][, 1:2], Z_splited[[2]][ ,1:2],
          Z_splited[[3]][ ,1:2])

  Z_k <- data.frame(z_k, y)
  print(ggplot(Z_k, aes(x = V1, y = V2, color = y)) + geom_point())
```



scatterplot of the wines but this time using the first two principal components on the standardized predictors.

```
X_scaled <- scale(X)

score <-princomp(X_scaled, cor = TRUE)$score[,1:2]

score[,1:2]
```

```
##            Comp.1      Comp.2
##   [1,] -3.31675081 -1.44346263
##   [2,] -2.20946492  0.33339289
##   [3,] -2.51674015 -1.03115130
##   [4,] -3.75706561 -2.75637191
##   [5,] -1.00890849 -0.86983082
##   [6,] -3.05025392 -2.12240111
##   [7,] -2.44908967 -1.17485013
##   [8,] -2.05943687 -1.60896307
##   [9,] -2.51087430 -0.91807096
##  [10,] -2.75362819 -0.78943767
##  [11,] -3.47973668 -1.30233324
##  [12,] -1.75475290 -0.61197723
##  [13,] -2.11346234 -0.67570634
##  [14,] -3.45815682 -1.13062988
##  [15,] -4.31278391 -2.09597558
```

```
##  [16,] -2.30518820 -1.66255173
##  [17,] -2.17195527 -2.32730534
##  [18,] -1.89897118 -1.63136888
##  [19,] -3.54198508 -2.51834367
##  [20,] -2.08452220 -1.06113799
##  [21,] -3.12440254 -0.78689711
##  [22,] -1.08657007 -0.24174355
##  [23,] -2.53522408  0.09184062
##  [24,] -1.64498834  0.51627893
##  [25,] -1.76157587  0.31714893
##  [26,] -0.99007910 -0.94066734
##  [27,] -1.77527763 -0.68617513
##  [28,] -1.23542396  0.08980704
##  [29,] -2.18840633 -0.68956962
##  [30,] -2.25610898 -0.19146194
##  [31,] -2.50022003 -1.24083383
##  [32,] -2.67741105 -1.47187365
##  [33,] -1.62857912 -0.05270445
##  [34,] -1.90269086 -1.63306043
##  [35,] -1.41038853 -0.69793432
##  [36,] -1.90382623 -0.17671095
##  [37,] -1.38486223 -0.65863985
##  [38,] -1.12220741 -0.11410976
##  [39,] -1.50219450  0.76943201
##  [40,] -2.52980109 -1.80300198
##  [41,] -2.58809543 -0.77961630
##  [42,] -0.66848199 -0.16996094
##  [43,] -3.07080699 -1.15591896
##  [44,] -0.46220914 -0.33074213
##  [45,] -2.10135193  0.07100892
##  [46,] -1.13616618 -1.77710739
##  [47,] -2.72660096 -1.19133469
##  [48,] -2.82133927 -0.64625860
##  [49,] -2.00985085 -1.24702946
##  [50,] -2.70749130 -1.75196741
##  [51,] -3.21491747 -0.16699199
##  [52,] -2.85895983 -0.74527880
##  [53,] -3.50560436 -1.61273386
##  [54,] -2.22479138 -1.87516800
##  [55,] -2.14698782 -1.01675154
##  [56,] -2.46932948 -1.32900831
##  [57,] -2.74151791 -1.43654878
##  [58,] -2.17374092 -1.21219984
##  [59,] -3.13938015 -1.73157912
##  [60,]  0.92858197  3.07348616
##  [61,]  1.54248014  1.38144351
##  [62,]  1.83624976  0.82998412
##  [63,] -0.03060683  1.26278614
##  [64,] -2.05026161  1.92503260
##  [65,]  0.60968083  1.90805881
##  [66,] -0.90022784  0.76391147
##  [67,] -2.24850719  1.88459248
##  [68,] -0.18338403  2.42714611
##  [69,]  0.81280503  0.22051399
```

```
##  [70,] -1.97562050  1.40328323
##  [71,]  1.57221622  0.88498314
##  [72,] -1.65768181  0.95671220
##  [73,]  0.72537239  1.06364540
##  [74,] -2.56222717 -0.26019855
##  [75,] -1.83256757  1.28787820
##  [76,]  0.86799290  2.44410119
##  [77,] -0.37001440  2.15390698
##  [78,]  1.45737704  1.38335177
##  [79,] -1.26293085  0.77084953
##  [80,] -0.37615037  1.02704340
##  [81,] -0.76206390  3.37505381
##  [82,] -1.03457797  1.45070974
##  [83,]  0.49487676  2.38124353
##  [84,]  2.53897708  0.08744336
##  [85,] -0.83532015  1.47367055
##  [86,] -0.78790461  2.02662652
##  [87,]  0.80683216  2.23383039
##  [88,]  0.55804262  2.37298543
##  [89,]  1.11511104  1.80224719
##  [90,]  0.55572283  2.65754004
##  [91,]  1.34928528  2.11800147
##  [92,]  1.56448261  1.85221452
##  [93,]  1.93255561  1.55949546
##  [94,] -0.74666594  2.31293171
##  [95,] -0.95745536  2.22352843
##  [96,] -2.54386518 -0.16927402
##  [97,]  0.54395259  0.36892655
##  [98,] -1.03104975  2.56556935
##  [99,] -2.25190942  1.43274138
## [100,] -1.41021602  2.16619177
## [101,] -0.79771979  2.37694880
## [102,]  0.54953173  2.29312864
## [103,]  0.16117374  1.16448332
## [104,]  0.65979494  2.67996119
## [105,] -0.39235441  2.09873171
## [106,]  1.77249908  1.71728847
## [107,]  0.36626736  2.16935330
## [108,]  1.62067257  1.35558339
## [109,] -0.08253578  2.30623459
## [110,] -1.57827507  1.46203429
## [111,] -1.42056925  1.41820664
## [112,]  0.27870275  1.93056809
## [113,]  1.30314497  0.76317231
## [114,]  0.45707187  2.26941561
## [115,]  0.49418585  1.93904505
## [116,] -0.48207441  3.87178385
## [117,]  0.25288888  2.82149237
## [118,]  0.10722764  1.92892204
## [119,]  2.43301260  1.25714104
## [120,]  0.55108954  2.22216155
## [121,] -0.73962193  1.40895667
## [122,] -1.33632173 -0.25333693
## [123,]  1.17708700  0.66396684
```

```
## [124,]  0.46233501  0.61828818
## [125,] -0.97847408  1.44557050
## [126,]  0.09680973  2.10999799
## [127,] -0.03848715  1.26676211
## [128,]  1.59715850  1.20814357
## [129,]  0.47956492  1.93884066
## [130,]  1.79283347  1.15028810
## [131,]  1.32710166 -0.17038923
## [132,]  2.38450083 -0.37458261
## [133,]  2.93694010 -0.26386183
## [134,]  2.14681113 -0.36825495
## [135,]  2.36986949  0.45963481
## [136,]  3.06384157 -0.35341284
## [137,]  3.91575378 -0.15458252
## [138,]  3.93646339 -0.65968723
## [139,]  3.09427612 -0.34884276
## [140,]  2.37447163 -0.29198035
## [141,]  2.77881295 -0.28680487
## [142,]  2.28656128 -0.37250784
## [143,]  2.98563349 -0.48921791
## [144,]  2.37519470 -0.48233372
## [145,]  2.20986553 -1.16005250
## [146,]  2.62562100 -0.56316076
## [147,]  4.28063878 -0.64967096
## [148,]  3.58264137 -1.27270275
## [149,]  2.80706372 -1.57053379
## [150,]  2.89965933 -2.04105701
## [151,]  2.32073698 -2.35636608
## [152,]  2.54983095 -2.04528309
## [153,]  1.81254128 -1.52764595
## [154,]  2.76014464 -2.13893235
## [155,]  2.73715050 -0.40988627
## [156,]  3.60486887 -1.80238422
## [157,]  2.88982600 -1.92521861
## [158,]  3.39215608 -1.31187639
## [159,]  1.04818190 -3.51508969
## [160,]  1.60991228 -2.40663816
## [161,]  3.14313097 -0.73816104
## [162,]  2.24015690 -1.17546529
## [163,]  2.84767378 -0.55604397
## [164,]  2.59749706 -0.69796554
## [165,]  2.94929937 -1.55530896
## [166,]  3.53003227 -0.88252680
## [167,]  2.40611054 -2.59235618
## [168,]  2.92908473 -1.27444695
## [169,]  2.18141278 -2.07753731
## [170,]  2.38092779 -2.58866743
## [171,]  3.21161722  0.25124910
## [172,]  3.67791872 -0.84774784
## [173,]  2.46555580 -2.19379830
## [174,]  3.37052415 -2.21628914
## [175,]  2.60195585 -1.75722935
## [176,]  2.67783946 -2.76089913
## [177,]  2.38701709 -2.29734668
```

```
## [178,]  3.20875816 -2.76891957
```

```r
n1 <- dim(Z_splited[[1]])[1]
n2 <- dim(Z_splited[[2]])[1]


score <- cbind(score, y = y)
score
```

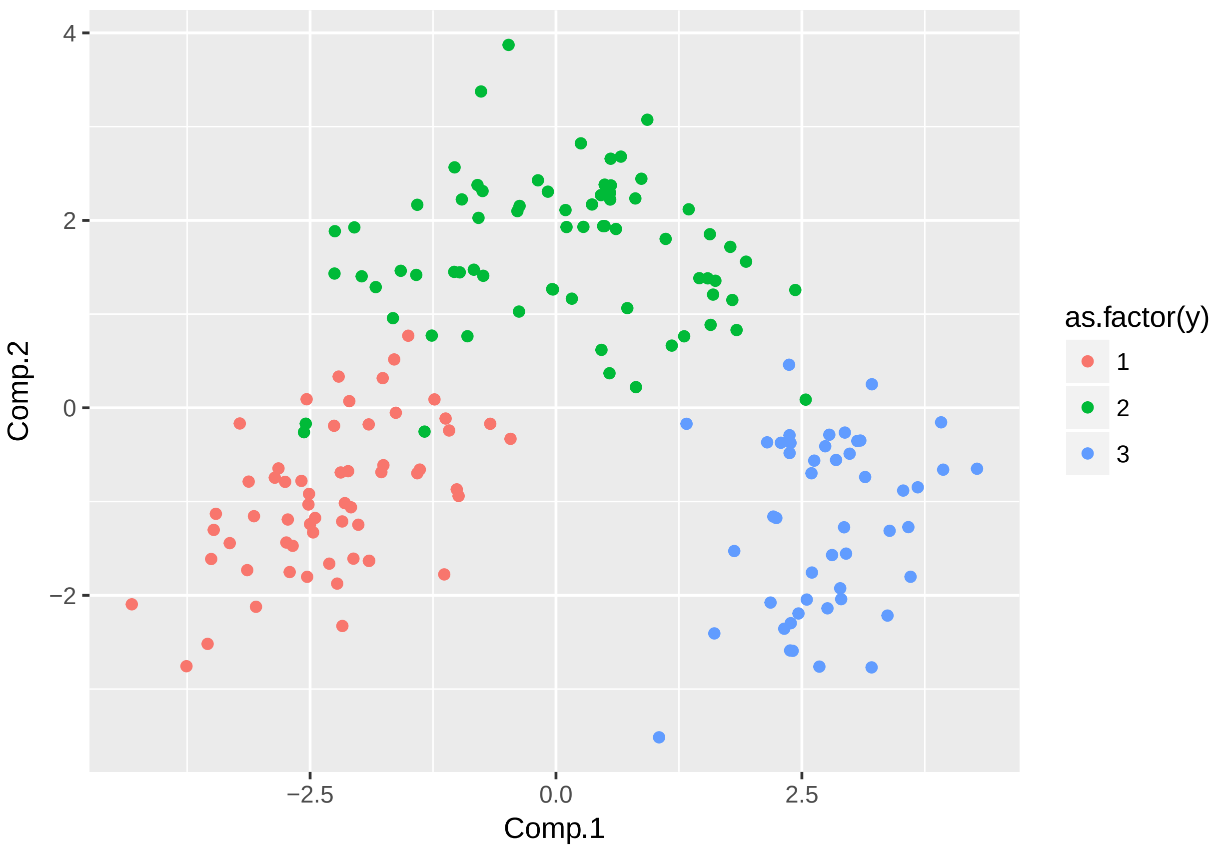
```
##              Comp.1      Comp.2 y
##   [1,] -3.31675081 -1.44346263 1
##   [2,] -2.20946492  0.33339289 1
##   [3,] -2.51674015 -1.03115130 1
##   [4,] -3.75706561 -2.75637191 1
##   [5,] -1.00890849 -0.86983082 1
##   [6,] -3.05025392 -2.12240111 1
##   [7,] -2.44908967 -1.17485013 1
##   [8,] -2.05943687 -1.60896307 1
##   [9,] -2.51087430 -0.91807096 1
##  [10,] -2.75362819 -0.78943767 1
##  [11,] -3.47973668 -1.30233324 1
##  [12,] -1.75475290 -0.61197723 1
##  [13,] -2.11346234 -0.67570634 1
##  [14,] -3.45815682 -1.13062988 1
##  [15,] -4.31278391 -2.09597558 1
##  [16,] -2.30518820 -1.66255173 1
##  [17,] -2.17195527 -2.32730534 1
##  [18,] -1.89897118 -1.63136888 1
##  [19,] -3.54198508 -2.51834367 1
##  [20,] -2.08452220 -1.06113799 1
##  [21,] -3.12440254 -0.78689711 1
##  [22,] -1.08657007 -0.24174355 1
##  [23,] -2.53522408  0.09184062 1
##  [24,] -1.64498834  0.51627893 1
##  [25,] -1.76157587  0.31714893 1
##  [26,] -0.99007910 -0.94066734 1
##  [27,] -1.77527763 -0.68617513 1
##  [28,] -1.23542396  0.08980704 1
##  [29,] -2.18840633 -0.68956962 1
##  [30,] -2.25610898 -0.19146194 1
##  [31,] -2.50022003 -1.24083383 1
##  [32,] -2.67741105 -1.47187365 1
##  [33,] -1.62857912 -0.05270445 1
##  [34,] -1.90269086 -1.63306043 1
##  [35,] -1.41038853 -0.69793432 1
##  [36,] -1.90382623 -0.17671095 1
##  [37,] -1.38486223 -0.65863985 1
##  [38,] -1.12220741 -0.11410976 1
##  [39,] -1.50219450  0.76943201 1
##  [40,] -2.52980109 -1.80300198 1
##  [41,] -2.58809543 -0.77961630 1
##  [42,] -0.66848199 -0.16996094 1
##  [43,] -3.07080699 -1.15591896 1
##  [44,] -0.46220914 -0.33074213 1
##  [45,] -2.10135193  0.07100892 1
```

```
##  [46,] -1.13616618 -1.77710739 1
##  [47,] -2.72660096 -1.19133469 1
##  [48,] -2.82133927 -0.64625860 1
##  [49,] -2.00985085 -1.24702946 1
##  [50,] -2.70749130 -1.75196741 1
##  [51,] -3.21491747 -0.16699199 1
##  [52,] -2.85895983 -0.74527880 1
##  [53,] -3.50560436 -1.61273386 1
##  [54,] -2.22479138 -1.87516800 1
##  [55,] -2.14698782 -1.01675154 1
##  [56,] -2.46932948 -1.32900831 1
##  [57,] -2.74151791 -1.43654878 1
##  [58,] -2.17374092 -1.21219984 1
##  [59,] -3.13938015 -1.73157912 1
##  [60,]  0.92858197  3.07348616 2
##  [61,]  1.54248014  1.38144351 2
##  [62,]  1.83624976  0.82998412 2
##  [63,] -0.03060683  1.26278614 2
##  [64,] -2.05026161  1.92503260 2
##  [65,]  0.60968083  1.90805881 2
##  [66,] -0.90022784  0.76391147 2
##  [67,] -2.24850719  1.88459248 2
##  [68,] -0.18338403  2.42714611 2
##  [69,]  0.81280503  0.22051399 2
##  [70,] -1.97562050  1.40328323 2
##  [71,]  1.57221622  0.88498314 2
##  [72,] -1.65768181  0.95671220 2
##  [73,]  0.72537239  1.06364540 2
##  [74,] -2.56222717 -0.26019855 2
##  [75,] -1.83256757  1.28787820 2
##  [76,]  0.86799290  2.44410119 2
##  [77,] -0.37001440  2.15390698 2
##  [78,]  1.45737704  1.38335177 2
##  [79,] -1.26293085  0.77084953 2
##  [80,] -0.37615037  1.02704340 2
##  [81,] -0.76206390  3.37505381 2
##  [82,] -1.03457797  1.45070974 2
##  [83,]  0.49487676  2.38124353 2
##  [84,]  2.53897708  0.08744336 2
##  [85,] -0.83532015  1.47367055 2
##  [86,] -0.78790461  2.02662652 2
##  [87,]  0.80683216  2.23383039 2
##  [88,]  0.55804262  2.37298543 2
##  [89,]  1.11511104  1.80224719 2
##  [90,]  0.55572283  2.65754004 2
##  [91,]  1.34928528  2.11800147 2
##  [92,]  1.56448261  1.85221452 2
##  [93,]  1.93255561  1.55949546 2
##  [94,] -0.74666594  2.31293171 2
##  [95,] -0.95745536  2.22352843 2
##  [96,] -2.54386518 -0.16927402 2
##  [97,]  0.54395259  0.36892655 2
##  [98,] -1.03104975  2.56556935 2
##  [99,] -2.25190942  1.43274138 2
```

```
## [100,] -1.41021602  2.16619177 2
## [101,] -0.79771979  2.37694880 2
## [102,]  0.54953173  2.29312864 2
## [103,]  0.16117374  1.16448332 2
## [104,]  0.65979494  2.67996119 2
## [105,] -0.39235441  2.09873171 2
## [106,]  1.77249908  1.71728847 2
## [107,]  0.36626736  2.16935330 2
## [108,]  1.62067257  1.35558339 2
## [109,] -0.08253578  2.30623459 2
## [110,] -1.57827507  1.46203429 2
## [111,] -1.42056925  1.41820664 2
## [112,]  0.27870275  1.93056809 2
## [113,]  1.30314497  0.76317231 2
## [114,]  0.45707187  2.26941561 2
## [115,]  0.49418585  1.93904505 2
## [116,] -0.48207441  3.87178385 2
## [117,]  0.25288888  2.82149237 2
## [118,]  0.10722764  1.92892204 2
## [119,]  2.43301260  1.25714104 2
## [120,]  0.55108954  2.22216155 2
## [121,] -0.73962193  1.40895667 2
## [122,] -1.33632173 -0.25333693 2
## [123,]  1.17708700  0.66396684 2
## [124,]  0.46233501  0.61828818 2
## [125,] -0.97847408  1.44557050 2
## [126,]  0.09680973  2.10999799 2
## [127,] -0.03848715  1.26676211 2
## [128,]  1.59715850  1.20814357 2
## [129,]  0.47956492  1.93884066 2
## [130,]  1.79283347  1.15028810 2
## [131,]  1.32710166 -0.17038923 3
## [132,]  2.38450083 -0.37458261 3
## [133,]  2.93694010 -0.26386183 3
## [134,]  2.14681113 -0.36825495 3
## [135,]  2.36986949  0.45963481 3
## [136,]  3.06384157 -0.35341284 3
## [137,]  3.91575378 -0.15458252 3
## [138,]  3.93646339 -0.65968723 3
## [139,]  3.09427612 -0.34884276 3
## [140,]  2.37447163 -0.29198035 3
## [141,]  2.77881295 -0.28680487 3
## [142,]  2.28656128 -0.37250784 3
## [143,]  2.98563349 -0.48921791 3
## [144,]  2.37519470 -0.48233372 3
## [145,]  2.20986553 -1.16005250 3
## [146,]  2.62562100 -0.56316076 3
## [147,]  4.28063878 -0.64967096 3
## [148,]  3.58264137 -1.27270275 3
## [149,]  2.80706372 -1.57053379 3
## [150,]  2.89965933 -2.04105701 3
## [151,]  2.32073698 -2.35636608 3
## [152,]  2.54983095 -2.04528309 3
## [153,]  1.81254128 -1.52764595 3
```

```
## [154,]  2.76014464 -2.13893235 3
## [155,]  2.73715050 -0.40988627 3
## [156,]  3.60486887 -1.80238422 3
## [157,]  2.88982600 -1.92521861 3
## [158,]  3.39215608 -1.31187639 3
## [159,]  1.04818190 -3.51508969 3
## [160,]  1.60991228 -2.40663816 3
## [161,]  3.14313097 -0.73816104 3
## [162,]  2.24015690 -1.17546529 3
## [163,]  2.84767378 -0.55604397 3
## [164,]  2.59749706 -0.69796554 3
## [165,]  2.94929937 -1.55530896 3
## [166,]  3.53003227 -0.88252680 3
## [167,]  2.40611054 -2.59235618 3
## [168,]  2.92908473 -1.27444695 3
## [169,]  2.18141278 -2.07753731 3
## [170,]  2.38092779 -2.58866743 3
## [171,]  3.21161722  0.25124910 3
## [172,]  3.67791872 -0.84774784 3
## [173,]  2.46555580 -2.19379830 3
## [174,]  3.37052415 -2.21628914 3
## [175,]  2.60195585 -1.75722935 3
## [176,]  2.67783946 -2.76089913 3
## [177,]  2.38701709 -2.29734668 3
## [178,]  3.20875816 -2.76891957 3
```

```
ggplot(as.data.frame(score), aes(x = Comp.1, y = Comp.2, color = as.factor(y))) + geom_point()
```



```
# for(i in 1:2){
#   z_k <- c(Z_splited[[1]][, i], Z_splited[[2]][ ,i],  Z_splited[[3]][ ,i])
#   Z_k <- data.frame(z_k, y)
#   print(ggplot(Z_k, aes(x = z_k, y = y, color = y)) + geom_point() +
#           labs(title = paste0("linear combinations Z_", i), y = "class", color = "class"))
#
# }
```

Calculate the correlations between zk and the predictors

```
# Q : Z with standardzied predictors or not??
# Q : interpret score?

cor(Z[,-3], X)
```

```
##        alcohol      malic        ash alcalinity magnesium    phenols
## [1,] 0.2798969 -0.4891760 0.01918243 -0.5299978 0.1935927 0.75482118
## [2,] 0.8162180  0.3178155 0.40451247 -0.2148215 0.3355196 0.07008972
##       flavanoids nonflavanoids proanthocyanins      color        hue
## [1,]  0.89849357   -0.51522117      0.53203867 -0.3441133  0.6840759
## [2,] -0.02635971   -0.02507846     -0.05042644  0.7665231 -0.3780354
##        dilution   proline
## [1,]  0.8503779 0.6148947
## [2,] -0.2031988 0.6717132
```

we can think

Create a matrix of size n $\times$ K, with the squared Mahalanobis distances

```
mahal <- matrix(0, n, K)

X_splited <- split(as.data.frame(X), y)

g_k <- lapply(X_splited, colMeans)


W_inverse <- solve(W)

for(i in 1:n){
  for(j in 1:K){
    mahal[i,j] <- (X[i, , drop = F] - g_k[[j]])  %*% W_inverse %*%
      t(X[i, , drop = F] - g_k[[j]])
  }
}

head(mahal)
```

```
##           [,1]     [,2]     [,3]
## [1,] 11.471872 51.37512 92.28077
## [2,]  8.738074 39.13556 83.11946
## [3,]  7.884262 34.50203 68.51471
## [4,] 13.484011 67.09116 87.00835
```

```
## [5,] 11.668097 17.12809 42.12974
## [6,]  6.913637 55.98424 85.16075
```

1. assign each observation to the class Gk for which the Mahalanobis distance d^2(xi, gk) is the smallest

```r
assigned <- apply(mahal, 1, which.min)

assigned
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2
##  [71] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [106] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3
## [141] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [176] 3 3 3
```

2. create a confussion matrix comparing the actual class versus the predicted class

```r
#data.frame(assigned, y)


table(assigned, y)
```

```
##         y
## assigned  1  2  3
##        1 59  0  0
##        2  0 71  0
##        3  0  0 48
```

```r
confusionMatrix(assigned, y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2  3
##          1 59  0  0
##          2  0 71  0
##          3  0  0 48
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9795, 1)
##     No Information Rate : 0.3989
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3
```

```
## Sensitivity            1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000
## Prevalence             0.3315   0.3989   0.2697
## Detection Rate         0.3315   0.3989   0.2697
## Detection Prevalence   0.3315   0.3989   0.2697
## Balanced Accuracy      1.0000   1.0000   1.0000
```