Rmarkdown take home exam

TAEK YUN LEE

2018/9/09

# 1-1

Minimize : In this case, and . So, the optimization looks like Minimize :

# 1-2

Answer : Argue that in this setting, the ridge coefficient estimates satisfy . Given the situations that , take the derivatives of the expression in (a) with respect to both and and setting them equal zero, then we get

The symmetry form in the above formula suggests that

# 1-3

Answer: The optimization looks like Minimize :

# 1-4

Argue that in this setting, the lasso coefficients and are not unique-in other wores, there are man possible solutions to the optimization problem in 3. Describe these solutions.

$ Answer : $ The Lasso contraint takes the form which plotted takes the shape of a diamond centered at origin (0,0). Next consider the sdquared optimization constrain . We use the facts , , , , and to simlify is to minimize: .

This optimization problem has a simple solution: . this is a line parallel to the edge of Lasso-diamond . Now the soluitions to the original Lasso optimization problem are contours of the function that touch the Lasso-diamond . Finally, as and vary along the line , these coniours touch the Lasso-diamond edge at different points. As a result, the enrire edge isd a potential solution to the Lasso optimization problem!

Similar argument can be made for the opposite Lasso-diamond edge: .

Thus, the Lasso problem does not have a unique solution. The general form of solution is

and .

# 2-1

The least square line is given by [ = 50 + 20GPA + 0.07IQ + 35Gender + 0.01GPAIQ - 10GPAGender] which becomes for the males [ = 50 + 20GPA + 0.07IQ + 0.01GPAIQ,] and for the females [ = 85 + 10GPA + 0.07IQ + 0.01GPAIQ.] So the starting salary for males is higher than for females on average iff which is equivalent to . Therefore iii. is the right answer.

# 2-2

It suffices to plug in the given values in the least square line for females given above and we obtain [ = 85 + 40 + 7.7 + 4.4 = 137.1,] which gives us a starting salary of $.

# 2-3

False. To verify if the GPA/IQ has an impact on the quality of the model we need to test the hypothesis and look at the p-value associated with the or the statistic to draw a conclusion.

# 3번

library(randomForest)  
dat3 = read.csv('C:/Users/renz/Desktop/data3.csv')  
dat3$k3 <- ifelse(dat3$k3 == 2 , 0 , dat3$k3)   
  
  
cols<-c('k2','k3','k4','k6','k7','k8','k10','k12','k13','k14')  
dat3[,cols]<-lapply(dat3[,cols], factor)   
  
dat3$k14 <- as.factor(dat3$k14)  
dat3$age <- as.factor(dat3$age)  
dat3$area <- as.factor(dat3$area)  
dat3$edu <- as.factor(dat3$edu)  
dat3$sex <- as.factor(dat3$sex)  
dat3$income <- as.factor(dat3$income)  
dat3$ideo\_self <- as.factor(dat3$ideo\_self)  
  
### k14 예측  
k = dat3[,c('sex','age1','age','area','edu','income','k14') ]  
  
  
train <- na.omit(k)  
rf.fit <- randomForest(k14 ~ sex + age+ age1 + area + edu + income , data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k14) , ][,c('sex','age1','age','area','edu','income','k14')]  
  
j=1   
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k14'] )) {  
   
 pred = predict(rf.fit, newdata = test[ j, ] , type ='response')  
 dat3[ i , 'k14'] = pred   
 j=j+1  
 }   
}   
### k13 예측  
  
k = dat3[,c('sex','age1','age','area','edu','income','k14','k13') ]  
  
  
train <- na.omit(k)  
rf.fit <- randomForest(k13 ~ sex + age+ age1 + area + edu + income + k14 , data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k13) , ][,c('sex','age1','age','area','edu','income','k14','k13')]  
  
  
j=1  
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k13'] )) {  
 pred = predict(rf.fit, newdata = test[ j , ] , type ='response')  
 dat3[ i , 'k13'] = pred   
 j=j+1}   
}   
### k6 예측  
  
k = dat3[,c('sex','age1','age','area','edu','income','k14','k13','k6') ]  
  
train <- na.omit(k)  
rf.fit <- randomForest(k6 ~ sex + age+ age1 + area + edu + income + k14 +k13, data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k6) , ][,c('sex','age1','age','area','edu','income','k14','k13','k6')]  
  
  
j=1  
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k6'] )) {  
 pred = predict(rf.fit, newdata = test[ j , ] , type ='response')  
 dat3[ i , 'k6'] = pred   
 j=j+1}   
}   
  
### k7 예측  
  
k = dat3[,c('sex','age1','age','area','edu','income','k14','k13','k6','k7') ]  
  
train <- na.omit(k)  
rf.fit <- randomForest(k7 ~ sex + age+ age1 + area + edu + income + k14 +k13 +k6, data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k7) , ][,c('sex','age1','age','area','edu','income','k14','k13','k6','k7')]  
  
  
j=1  
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k7'] )) {  
 pred = predict(rf.fit, newdata = test[ j , ] , type ='response')  
 dat3[ i , 'k7'] = pred   
 j=j+1}   
}   
  
### k8 예측  
  
k = dat3[,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8') ]  
  
train <- na.omit(k)  
rf.fit <- randomForest(k8 ~ sex + age+ age1 + area + edu + income + k14 +k13 +k6 +k7, data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k8) , ][,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8')]  
  
  
j=1  
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k8'] )) {  
 pred = predict(rf.fit, newdata = test[ j , ] , type ='response')  
 dat3[ i , 'k8'] = pred   
 j=j+1}   
}   
  
### k2 예측  
  
k = dat3[,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2') ]  
  
train <- na.omit(k)  
rf.fit <- randomForest(k2 ~ sex + age+ age1 + area + edu + income + k14 +k13 +k6 +k7 +k8, data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k2) , ][,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2')]  
  
  
j=1  
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k2'] )) {  
 pred = predict(rf.fit, newdata = test[ j , ] , type ='response')  
 dat3[ i , 'k2'] = pred   
 j=j+1}   
}   
  
### k10 예측  
  
k = dat3[,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2','k10') ]  
  
train <- na.omit(k)  
rf.fit <- randomForest(k10 ~ sex + age+ age1 + area + edu + income + k14 +k13 +k6 +k7 +k8 +k2, data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k10) , ][,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2','k10')]  
  
  
j=1  
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k10'] )) {  
 pred = predict(rf.fit, newdata = test[ j , ] , type ='response')  
 dat3[ i , 'k10'] = pred   
 j=j+1}   
}   
  
### k12 예측  
  
k = dat3[,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2','k10','k12') ]  
  
train <- na.omit(k)  
rf.fit <- randomForest(k12 ~ sex + age+ age1 + area + edu + income + k14 +k13 +k6 +k7 +k8 +k2 +k10, data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k12) , ][,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2','k10','k12')]  
  
  
j=1  
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k12'] )) {  
 pred = predict(rf.fit, newdata = test[ j , ] , type ='response')  
 dat3[ i , 'k12'] = pred   
 j=j+1}   
}   
  
### k4 예측  
  
k = dat3[,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2','k10','k12','k4') ]  
  
train <- na.omit(k)  
rf.fit <- randomForest(k4 ~ sex + age+ age1 + area + edu + income + k14 +k13 +k6 +k7 +k8 +k2 +k10 +k12, data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k4) , ][,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2','k10','k12','k4')]  
  
  
j=1  
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k4'] )) {  
 pred = predict(rf.fit, newdata = test[ j , ] , type ='response')  
 dat3[ i , 'k4'] = pred   
 j=j+1}   
}   
  
### k3 예측  
  
k = dat3[,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2','k10','k12','k4','k3') ]  
  
train <- na.omit(k)  
rf.fit <- randomForest(k3 ~ sex + age+ age1 + area + edu + income + k14 +k13 +k6 +k7 +k8 +k2 +k10 +k12 +k4, data = train, n\_tree =100)  
test <- dat3[is.na(dat3$k3) , ][,c('sex','age1','age','area','edu','income','k14','k13','k6','k7','k8','k2','k10','k12','k4','k3')]  
  
  
j=1  
for(i in 1:nrow(dat3)){  
 if (is.na(dat3[ i , 'k3'] )) {  
 pred = predict(rf.fit, newdata = test[ j , ] , type ='response')  
 dat3[ i , 'k3'] = pred   
 j=j+1}   
}   
  
###   
  
#install.packages('caret')  
library(caret)  
  
  
result = list()  
result\_acc = 0  
Fold\_index <- createFolds(1:nrow(dat3), k = 10)  
  
Out\_of\_Sample <- c()  
for(k in 1:10){  
 Train <- dat3[-Fold\_index[[k]],]  
 Test <- dat3[Fold\_index[[k]],]  
   
 out <- randomForest(ideo\_self ~ ., data = Train)  
 test\_pred <- predict(out, Test)  
 Out\_of\_Sample[[k]] <- test\_pred  
   
 result[[k]] = table(Test$ideo\_self, test\_pred)  
 result\_acc[k] = sum(diag(result[[k]]))/sum(result[[k]])  
 print(confusionMatrix(Test$ideo\_self , test\_pred))  
}

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 0 0 0 6 0 1 0 0 0  
## 1 0 0 0 0 0 1 0 0 0 0 0  
## 2 0 0 0 1 0 0 0 0 0 0 0  
## 3 0 0 0 3 1 12 0 1 0 0 0  
## 4 0 0 0 2 1 7 0 0 0 0 0  
## 5 0 0 1 4 6 22 2 0 0 0 0  
## 6 0 0 0 1 2 5 1 0 0 0 1  
## 7 0 0 0 0 1 9 0 0 0 0 3  
## 8 0 0 0 0 0 2 0 1 1 0 2  
## 9 0 0 0 0 0 1 0 0 0 0 0  
## 10 0 0 0 0 0 4 0 1 0 0 0  
##   
## Overall Statistics  
##   
## Accuracy : 0.2642   
## 95% CI : (0.1832, 0.3587)  
## No Information Rate : 0.6509   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.0163   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity NA NA 0.000000 0.2727 0.090909 0.3188  
## Specificity 0.93396 0.990566 0.990476 0.8526 0.905263 0.6486  
## Pos Pred Value NA NA 0.000000 0.1765 0.100000 0.6286  
## Neg Pred Value NA NA 0.990476 0.9101 0.895833 0.3380  
## Prevalence 0.00000 0.000000 0.009434 0.1038 0.103774 0.6509  
## Detection Rate 0.00000 0.000000 0.000000 0.0283 0.009434 0.2075  
## Detection Prevalence 0.06604 0.009434 0.009434 0.1604 0.094340 0.3302  
## Balanced Accuracy NA NA 0.495238 0.5627 0.498086 0.4837  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.333333 0.00000 1.000000 NA 0.00000  
## Specificity 0.912621 0.87255 0.952381 0.990566 0.95000  
## Pos Pred Value 0.100000 0.00000 0.166667 NA 0.00000  
## Neg Pred Value 0.979167 0.95699 1.000000 NA 0.94059  
## Prevalence 0.028302 0.03774 0.009434 0.000000 0.05660  
## Detection Rate 0.009434 0.00000 0.009434 0.000000 0.00000  
## Detection Prevalence 0.094340 0.12264 0.056604 0.009434 0.04717  
## Balanced Accuracy 0.622977 0.43627 0.976190 NA 0.47500  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 1 0 0 3 0 0 0 0 1  
## 1 0 0 0 0 1 2 0 0 0 0 0  
## 2 0 0 0 1 0 6 0 0 0 0 0  
## 3 0 0 0 1 1 8 1 0 0 0 1  
## 4 0 0 0 3 1 8 0 1 0 0 0  
## 5 0 0 0 1 1 21 0 1 0 0 0  
## 6 0 0 0 0 0 8 3 0 2 0 1  
## 7 0 0 0 0 2 7 2 1 0 0 0  
## 8 0 0 0 0 0 3 2 1 1 1 3  
## 9 0 0 0 0 0 0 1 0 0 1 0  
## 10 0 0 0 0 0 0 1 0 0 0 1  
##   
## Overall Statistics  
##   
## Accuracy : 0.2857   
## 95% CI : (0.2018, 0.3821)  
## No Information Rate : 0.6286   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.1294   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity NA NA 0.000000 0.166667 0.166667 0.3182  
## Specificity 0.95238 0.97143 0.932692 0.888889 0.878788 0.9231  
## Pos Pred Value NA NA 0.000000 0.083333 0.076923 0.8750  
## Neg Pred Value NA NA 0.989796 0.946237 0.945652 0.4444  
## Prevalence 0.00000 0.00000 0.009524 0.057143 0.057143 0.6286  
## Detection Rate 0.00000 0.00000 0.000000 0.009524 0.009524 0.2000  
## Detection Prevalence 0.04762 0.02857 0.066667 0.114286 0.123810 0.2286  
## Balanced Accuracy NA NA 0.466346 0.527778 0.522727 0.6206  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.30000 0.250000 0.333333 0.500000 0.142857  
## Specificity 0.88421 0.891089 0.901961 0.990291 0.989796  
## Pos Pred Value 0.21429 0.083333 0.090909 0.500000 0.500000  
## Neg Pred Value 0.92308 0.967742 0.978723 0.990291 0.941748  
## Prevalence 0.09524 0.038095 0.028571 0.019048 0.066667  
## Detection Rate 0.02857 0.009524 0.009524 0.009524 0.009524  
## Detection Prevalence 0.13333 0.114286 0.104762 0.019048 0.019048  
## Balanced Accuracy 0.59211 0.570545 0.617647 0.745146 0.566327  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 0 0 0 5 0 0 0 0 0  
## 1 0 0 0 0 0 3 0 0 0 0 0  
## 2 0 0 0 3 1 1 0 0 0 0 0  
## 3 0 0 0 3 1 7 0 0 0 0 0  
## 4 0 0 0 3 1 5 1 0 0 0 0  
## 5 0 0 0 3 4 19 1 1 2 1 3  
## 6 0 0 0 1 0 6 2 4 0 0 0  
## 7 0 0 0 0 0 3 1 4 1 0 1  
## 8 0 0 0 0 2 2 0 0 2 0 0  
## 9 0 0 0 0 0 1 1 2 0 0 0  
## 10 0 0 0 0 0 3 1 0 2 0 0  
##   
## Overall Statistics  
##   
## Accuracy : 0.2897   
## 95% CI : (0.2061, 0.3854)  
## No Information Rate : 0.514   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.1039   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity NA NA NA 0.23077 0.111111 0.3455  
## Specificity 0.95327 0.97196 0.95327 0.91489 0.908163 0.7115  
## Pos Pred Value NA NA NA 0.27273 0.100000 0.5588  
## Neg Pred Value NA NA NA 0.89583 0.917526 0.5068  
## Prevalence 0.00000 0.00000 0.00000 0.12150 0.084112 0.5140  
## Detection Rate 0.00000 0.00000 0.00000 0.02804 0.009346 0.1776  
## Detection Prevalence 0.04673 0.02804 0.04673 0.10280 0.093458 0.3178  
## Balanced Accuracy NA NA NA 0.57283 0.509637 0.5285  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.28571 0.36364 0.28571 0.000000 0.00000  
## Specificity 0.89000 0.93750 0.96000 0.962264 0.94175  
## Pos Pred Value 0.15385 0.40000 0.33333 0.000000 0.00000  
## Neg Pred Value 0.94681 0.92784 0.95050 0.990291 0.96040  
## Prevalence 0.06542 0.10280 0.06542 0.009346 0.03738  
## Detection Rate 0.01869 0.03738 0.01869 0.000000 0.00000  
## Detection Prevalence 0.12150 0.09346 0.05607 0.037383 0.05607  
## Balanced Accuracy 0.58786 0.65057 0.62286 0.481132 0.47087  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 0 1 0 2 0 0 0 0 0  
## 1 0 0 0 1 1 1 0 0 0 0 0  
## 2 1 0 0 1 1 1 0 0 0 0 0  
## 3 0 0 0 2 2 6 1 0 0 0 0  
## 4 0 0 0 1 1 8 0 0 0 0 0  
## 5 0 0 0 3 1 22 1 2 1 0 2  
## 6 0 0 0 0 1 11 1 2 0 0 0  
## 7 0 0 0 0 0 7 1 2 3 0 0  
## 8 0 0 0 0 0 1 0 2 2 0 0  
## 9 0 0 0 0 0 0 0 1 0 0 0  
## 10 0 0 0 0 0 3 0 1 1 0 3  
##   
## Overall Statistics  
##   
## Accuracy : 0.3143   
## 95% CI : (0.2272, 0.4122)  
## No Information Rate : 0.5905   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.1213   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity 0.000000 NA NA 0.22222 0.142857 0.3548  
## Specificity 0.971154 0.97143 0.9619 0.90625 0.908163 0.7674  
## Pos Pred Value 0.000000 NA NA 0.18182 0.100000 0.6875  
## Neg Pred Value 0.990196 NA NA 0.92553 0.936842 0.4521  
## Prevalence 0.009524 0.00000 0.0000 0.08571 0.066667 0.5905  
## Detection Rate 0.000000 0.00000 0.0000 0.01905 0.009524 0.2095  
## Detection Prevalence 0.028571 0.02857 0.0381 0.10476 0.095238 0.3048  
## Balanced Accuracy 0.485577 NA NA 0.56424 0.525510 0.5611  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.250000 0.20000 0.28571 NA 0.60000  
## Specificity 0.861386 0.88421 0.96939 0.990476 0.95000  
## Pos Pred Value 0.066667 0.15385 0.40000 NA 0.37500  
## Neg Pred Value 0.966667 0.91304 0.95000 NA 0.97938  
## Prevalence 0.038095 0.09524 0.06667 0.000000 0.04762  
## Detection Rate 0.009524 0.01905 0.01905 0.000000 0.02857  
## Detection Prevalence 0.142857 0.12381 0.04762 0.009524 0.07619  
## Balanced Accuracy 0.555693 0.54211 0.62755 NA 0.77500  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 0 0 1 2 0 0 0 0 0  
## 1 0 0 1 0 0 3 0 0 0 0 0  
## 2 0 0 0 1 0 3 0 0 0 0 0  
## 3 0 0 1 1 1 6 1 0 0 0 0  
## 4 0 0 1 2 1 7 0 0 0 0 1  
## 5 0 0 0 4 5 24 1 1 1 0 1  
## 6 0 0 0 1 1 5 1 0 0 0 0  
## 7 0 0 0 1 0 3 1 2 2 0 0  
## 8 0 0 0 0 0 0 0 1 2 0 0  
## 9 0 0 0 0 0 1 1 0 1 1 0  
## 10 0 0 0 0 0 3 0 2 5 0 1  
##   
## Overall Statistics  
##   
## Accuracy : 0.3143   
## 95% CI : (0.2272, 0.4122)  
## No Information Rate : 0.5429   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.1139   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity NA NA 0.00000 0.100000 0.111111 0.4211  
## Specificity 0.97143 0.9619 0.96078 0.905263 0.885417 0.7292  
## Pos Pred Value NA NA 0.00000 0.100000 0.083333 0.6486  
## Neg Pred Value NA NA 0.97030 0.905263 0.913978 0.5147  
## Prevalence 0.00000 0.0000 0.02857 0.095238 0.085714 0.5429  
## Detection Rate 0.00000 0.0000 0.00000 0.009524 0.009524 0.2286  
## Detection Prevalence 0.02857 0.0381 0.03810 0.095238 0.114286 0.3524  
## Balanced Accuracy NA NA 0.48039 0.502632 0.498264 0.5751  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.200000 0.33333 0.18182 1.000000 0.333333  
## Specificity 0.930000 0.92929 0.98936 0.971154 0.901961  
## Pos Pred Value 0.125000 0.22222 0.66667 0.250000 0.090909  
## Neg Pred Value 0.958763 0.95833 0.91176 1.000000 0.978723  
## Prevalence 0.047619 0.05714 0.10476 0.009524 0.028571  
## Detection Rate 0.009524 0.01905 0.01905 0.009524 0.009524  
## Detection Prevalence 0.076190 0.08571 0.02857 0.038095 0.104762  
## Balanced Accuracy 0.565000 0.63131 0.58559 0.985577 0.617647  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 0 0 0 2 0 0 0 0 0  
## 1 0 0 1 0 0 2 0 0 0 0 0  
## 2 0 0 0 0 0 4 0 0 0 0 0  
## 3 0 0 0 4 1 5 2 0 0 0 0  
## 4 1 0 0 1 2 11 1 1 0 0 0  
## 5 1 0 0 3 0 22 0 4 1 0 0  
## 6 0 0 0 0 0 8 0 3 0 0 2  
## 7 0 0 0 1 1 3 1 2 1 1 0  
## 8 0 0 0 0 0 5 1 0 0 2 0  
## 9 0 0 0 0 0 0 0 0 0 0 1  
## 10 0 0 0 0 0 2 0 1 1 0 2  
##   
## Overall Statistics  
##   
## Accuracy : 0.2991   
## 95% CI : (0.2144, 0.3952)  
## No Information Rate : 0.5981   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.1132   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity 0.00000 NA 0.000000 0.44444 0.50000 0.3438  
## Specificity 0.98095 0.97196 0.962264 0.91837 0.85437 0.7907  
## Pos Pred Value 0.00000 NA 0.000000 0.33333 0.11765 0.7097  
## Neg Pred Value 0.98095 NA 0.990291 0.94737 0.97778 0.4474  
## Prevalence 0.01869 0.00000 0.009346 0.08411 0.03738 0.5981  
## Detection Rate 0.00000 0.00000 0.000000 0.03738 0.01869 0.2056  
## Detection Prevalence 0.01869 0.02804 0.037383 0.11215 0.15888 0.2897  
## Balanced Accuracy 0.49048 NA 0.481132 0.68141 0.67718 0.5672  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.00000 0.18182 0.00000 0.000000 0.40000  
## Specificity 0.87255 0.91667 0.92308 0.990385 0.96078  
## Pos Pred Value 0.00000 0.20000 0.00000 0.000000 0.33333  
## Neg Pred Value 0.94681 0.90722 0.96970 0.971698 0.97030  
## Prevalence 0.04673 0.10280 0.02804 0.028037 0.04673  
## Detection Rate 0.00000 0.01869 0.00000 0.000000 0.01869  
## Detection Prevalence 0.12150 0.09346 0.07477 0.009346 0.05607  
## Balanced Accuracy 0.43627 0.54924 0.46154 0.495192 0.68039  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 0 1 0 4 0 0 0 0 0  
## 1 0 0 0 0 0 3 0 0 0 0 0  
## 2 0 0 0 0 2 3 2 0 0 0 0  
## 3 0 0 1 2 5 5 1 1 0 0 0  
## 4 0 0 1 0 0 6 0 0 1 0 0  
## 5 0 0 0 2 2 25 3 0 2 0 0  
## 6 0 0 1 1 0 5 0 1 1 0 1  
## 7 0 0 0 1 0 2 0 3 0 0 0  
## 8 0 0 0 0 0 3 0 2 2 0 0  
## 9 0 0 0 0 0 1 0 0 0 1 0  
## 10 0 0 0 0 0 4 0 1 1 2 0  
##   
## Overall Statistics  
##   
## Accuracy : 0.3143   
## 95% CI : (0.2272, 0.4122)  
## No Information Rate : 0.581   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.1191   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity NA NA 0.00000 0.28571 0.00000 0.4098  
## Specificity 0.95238 0.97143 0.93137 0.86735 0.91667 0.7955  
## Pos Pred Value NA NA 0.00000 0.13333 0.00000 0.7353  
## Neg Pred Value NA NA 0.96939 0.94444 0.90722 0.4930  
## Prevalence 0.00000 0.00000 0.02857 0.06667 0.08571 0.5810  
## Detection Rate 0.00000 0.00000 0.00000 0.01905 0.00000 0.2381  
## Detection Prevalence 0.04762 0.02857 0.06667 0.14286 0.07619 0.3238  
## Balanced Accuracy NA NA 0.46569 0.57653 0.45833 0.6026  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.00000 0.37500 0.28571 0.333333 0.000000  
## Specificity 0.89899 0.96907 0.94898 0.990196 0.923077  
## Pos Pred Value 0.00000 0.50000 0.28571 0.500000 0.000000  
## Neg Pred Value 0.93684 0.94949 0.94898 0.980583 0.989691  
## Prevalence 0.05714 0.07619 0.06667 0.028571 0.009524  
## Detection Rate 0.00000 0.02857 0.01905 0.009524 0.000000  
## Detection Prevalence 0.09524 0.05714 0.06667 0.019048 0.076190  
## Balanced Accuracy 0.44949 0.67204 0.61735 0.661765 0.461538  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 0 1 0 0 0 0 0 0 0  
## 1 0 0 0 1 1 1 0 0 0 0 0  
## 2 0 0 0 1 0 4 0 0 0 0 0  
## 3 0 0 1 2 0 5 1 0 0 0 0  
## 4 0 0 0 2 0 9 0 0 0 0 0  
## 5 0 0 0 3 0 26 1 0 0 0 1  
## 6 0 0 0 0 0 11 2 3 0 0 0  
## 7 0 0 0 0 0 5 3 2 2 0 0  
## 8 0 0 0 0 1 4 1 4 2 0 1  
## 9 0 0 0 0 0 1 0 0 2 0 0  
## 10 0 0 0 0 0 1 0 0 0 0 0  
##   
## Overall Statistics  
##   
## Accuracy : 0.3238   
## 95% CI : (0.2357, 0.4221)  
## No Information Rate : 0.6381   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.1245   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity NA NA 0.000000 0.20000 0.00000 0.3881  
## Specificity 0.990476 0.97143 0.951923 0.92632 0.89320 0.8684  
## Pos Pred Value NA NA 0.000000 0.22222 0.00000 0.8387  
## Neg Pred Value NA NA 0.990000 0.91667 0.97872 0.4459  
## Prevalence 0.000000 0.00000 0.009524 0.09524 0.01905 0.6381  
## Detection Rate 0.000000 0.00000 0.000000 0.01905 0.00000 0.2476  
## Detection Prevalence 0.009524 0.02857 0.047619 0.08571 0.10476 0.2952  
## Balanced Accuracy NA NA 0.475962 0.56316 0.44660 0.6282  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.25000 0.22222 0.33333 NA 0.000000  
## Specificity 0.85567 0.89583 0.88889 0.97143 0.990291  
## Pos Pred Value 0.12500 0.16667 0.15385 NA 0.000000  
## Neg Pred Value 0.93258 0.92473 0.95652 NA 0.980769  
## Prevalence 0.07619 0.08571 0.05714 0.00000 0.019048  
## Detection Rate 0.01905 0.01905 0.01905 0.00000 0.000000  
## Detection Prevalence 0.15238 0.11429 0.12381 0.02857 0.009524  
## Balanced Accuracy 0.55284 0.55903 0.61111 NA 0.495146  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 0 0 1 2 0 0 0 0 0  
## 1 0 0 0 0 0 2 0 0 0 0 0  
## 2 0 0 0 2 0 2 0 0 0 0 0  
## 3 0 0 1 0 1 9 1 0 0 0 0  
## 4 0 0 0 0 1 10 0 0 1 0 0  
## 5 0 0 0 3 1 27 2 0 1 0 2  
## 6 0 0 0 0 1 4 3 1 0 0 0  
## 7 0 0 0 1 1 7 1 1 0 0 0  
## 8 0 0 0 0 0 4 0 0 1 0 0  
## 9 0 0 0 0 0 0 1 0 1 0 1  
## 10 0 0 0 0 1 3 0 1 1 0 2  
##   
## Overall Statistics  
##   
## Accuracy : 0.3333   
## 95% CI : (0.2443, 0.432)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.1009   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity NA NA 0.000000 0.00000 0.142857 0.3857  
## Specificity 0.97143 0.98095 0.961538 0.87879 0.887755 0.7429  
## Pos Pred Value NA NA 0.000000 0.00000 0.083333 0.7500  
## Neg Pred Value NA NA 0.990099 0.93548 0.935484 0.3768  
## Prevalence 0.00000 0.00000 0.009524 0.05714 0.066667 0.6667  
## Detection Rate 0.00000 0.00000 0.000000 0.00000 0.009524 0.2571  
## Detection Prevalence 0.02857 0.01905 0.038095 0.11429 0.114286 0.3429  
## Balanced Accuracy NA NA 0.480769 0.43939 0.515306 0.5643  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.37500 0.333333 0.200000 NA 0.40000  
## Specificity 0.93814 0.901961 0.960000 0.97143 0.94000  
## Pos Pred Value 0.33333 0.090909 0.200000 NA 0.25000  
## Neg Pred Value 0.94792 0.978723 0.960000 NA 0.96907  
## Prevalence 0.07619 0.028571 0.047619 0.00000 0.04762  
## Detection Rate 0.02857 0.009524 0.009524 0.00000 0.01905  
## Detection Prevalence 0.08571 0.104762 0.047619 0.02857 0.07619  
## Balanced Accuracy 0.65657 0.617647 0.580000 NA 0.67000  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1 2 3 4 5 6 7 8 9 10  
## 0 0 0 0 1 1 1 0 0 0 0 0  
## 1 0 0 0 0 0 2 0 0 0 0 0  
## 2 0 0 0 2 1 2 0 0 0 0 0  
## 3 0 0 0 3 1 6 0 0 0 0 0  
## 4 0 0 0 1 2 9 0 0 0 0 1  
## 5 0 0 0 2 2 22 1 0 2 0 0  
## 6 0 0 0 0 0 8 1 2 1 1 0  
## 7 0 0 0 2 0 5 1 2 0 0 0  
## 8 0 0 0 0 0 7 0 1 0 0 1  
## 9 0 0 0 0 0 3 0 0 0 0 1  
## 10 0 0 0 1 0 3 1 0 0 0 1  
##   
## Overall Statistics  
##   
## Accuracy : 0.2981   
## 95% CI : (0.2123, 0.3957)  
## No Information Rate : 0.6538   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.1043   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity NA NA NA 0.25000 0.28571 0.3235  
## Specificity 0.97115 0.98077 0.95192 0.92391 0.88660 0.8056  
## Pos Pred Value NA NA NA 0.30000 0.15385 0.7586  
## Neg Pred Value NA NA NA 0.90426 0.94505 0.3867  
## Prevalence 0.00000 0.00000 0.00000 0.11538 0.06731 0.6538  
## Detection Rate 0.00000 0.00000 0.00000 0.02885 0.01923 0.2115  
## Detection Prevalence 0.02885 0.01923 0.04808 0.09615 0.12500 0.2788  
## Balanced Accuracy NA NA NA 0.58696 0.58616 0.5645  
## Class: 6 Class: 7 Class: 8 Class: 9 Class: 10  
## Sensitivity 0.250000 0.40000 0.00000 0.000000 0.250000  
## Specificity 0.880000 0.91919 0.91089 0.961165 0.950000  
## Pos Pred Value 0.076923 0.20000 0.00000 0.000000 0.166667  
## Neg Pred Value 0.967033 0.96809 0.96842 0.990000 0.969388  
## Prevalence 0.038462 0.04808 0.02885 0.009615 0.038462  
## Detection Rate 0.009615 0.01923 0.00000 0.000000 0.009615  
## Detection Prevalence 0.125000 0.09615 0.08654 0.038462 0.057692  
## Balanced Accuracy 0.565000 0.65960 0.45545 0.480583 0.600000

result\_acc

## [1] 0.2641509 0.2857143 0.2897196 0.3142857 0.3142857 0.2990654 0.3142857  
## [8] 0.3238095 0.3333333 0.2980769

mean(result\_acc)

## [1] 0.3036727

ideo\_self를 예측하기전에 우선 k2 ~ k14 사이의 NA 값들을 랜덤포레스트를 이용하여 값을 채워 넣습니다. K14 -> K13 -> K6 -> K7 -> K8 -> K2 -> K10 -> K12 -> K4 -> K3 이와 같은 순서대로 na값을 채워넣었고  
처음 k14를 채워넣을때 ‘sex’,‘age1’,‘age’,‘area’,‘edu’,’income’들을 이용하여 학습시켰고 그후에 k14가 모두 채워지면 k14또한 변수로서 추가하는 방식으로 수행하였습니다. k2 ~ k14사이의 NA값을 모두채워 넣은뒤에는 모든 변수를 이용하여 랜덤포레스트로 ideo\_self를 예측하였습니다.

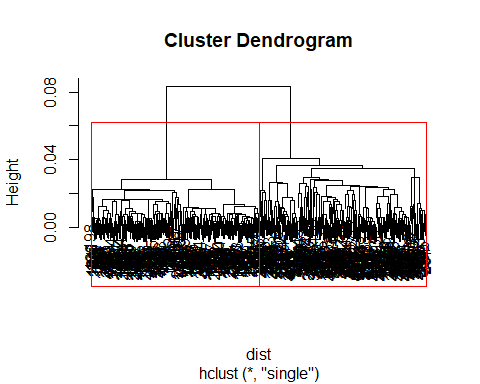
Acurracy는 0.3036727정도로 나왔습니다.

# 4번

## 군집  
library(cluster)  
dat4 = read.csv('C:/Users/renz/Desktop/datt4.csv')   
  
dat4$V1<- as.factor(dat4$V1)  
  
test.y = dat4$V1  
  
dist = dat4[,c(2,3)]  
dist <- daisy(dat4[,c(2,3)], metric = "gower")  
fit = hclust(dist, method = 'single')  
  
sum(cutree(fit, 2) - 1 == test.y)

## [1] 500

plot(fit)  
rect.hclust(fit, k=2, border="red")



## 예측  
  
library(randomForest)  
library(caret)  
  
dat4$V1 <- as.factor(dat4$V1 )  
  
result = list()  
result\_acc = 0  
Fold\_index <- createFolds(1:nrow(dat4), k = 10)  
  
Out\_of\_Sample <- c()  
for(k in 1:10){  
 Train <- dat4[-Fold\_index[[k]],]  
 Test <- dat4[Fold\_index[[k]],]  
   
 out <- randomForest(V1 ~ V2 + V3, data = Train)  
 test\_pred <- predict(out, Test)  
 Out\_of\_Sample[[k]] <- test\_pred  
   
 result[[k]] = table(Test$V1, test\_pred)  
 result\_acc[k] = sum(diag(result[[k]]))/sum(result[[k]])  
 print(confusionMatrix(Test$V1 , test\_pred))  
}

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 25 0  
## 1 0 26  
##   
## Accuracy : 1   
## 95% CI : (0.9302, 1)  
## No Information Rate : 0.5098   
## P-Value [Acc > NIR] : 1.196e-15   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0000   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 1.0000   
## Prevalence : 0.4902   
## Detection Rate : 0.4902   
## Detection Prevalence : 0.4902   
## Balanced Accuracy : 1.0000   
##   
## 'Positive' Class : 0   
##   
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 24 0  
## 1 0 25  
##   
## Accuracy : 1   
## 95% CI : (0.9275, 1)  
## No Information Rate : 0.5102   
## P-Value [Acc > NIR] : 4.78e-15   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0000   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 1.0000   
## Prevalence : 0.4898   
## Detection Rate : 0.4898   
## Detection Prevalence : 0.4898   
## Balanced Accuracy : 1.0000   
##   
## 'Positive' Class : 0   
##   
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 26 0  
## 1 0 25  
##   
## Accuracy : 1   
## 95% CI : (0.9302, 1)  
## No Information Rate : 0.5098   
## P-Value [Acc > NIR] : 1.196e-15   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0000   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 1.0000   
## Prevalence : 0.5098   
## Detection Rate : 0.5098   
## Detection Prevalence : 0.5098   
## Balanced Accuracy : 1.0000   
##   
## 'Positive' Class : 0   
##   
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 25 0  
## 1 0 24  
##   
## Accuracy : 1   
## 95% CI : (0.9275, 1)  
## No Information Rate : 0.5102   
## P-Value [Acc > NIR] : 4.78e-15   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0000   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 1.0000   
## Prevalence : 0.5102   
## Detection Rate : 0.5102   
## Detection Prevalence : 0.5102   
## Balanced Accuracy : 1.0000   
##   
## 'Positive' Class : 0   
##   
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 25 0  
## 1 0 25  
##   
## Accuracy : 1   
## 95% CI : (0.9289, 1)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : 8.882e-16   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0   
## Specificity : 1.0   
## Pos Pred Value : 1.0   
## Neg Pred Value : 1.0   
## Prevalence : 0.5   
## Detection Rate : 0.5   
## Detection Prevalence : 0.5   
## Balanced Accuracy : 1.0   
##   
## 'Positive' Class : 0   
##   
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 25 0  
## 1 0 25  
##   
## Accuracy : 1   
## 95% CI : (0.9289, 1)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : 8.882e-16   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0   
## Specificity : 1.0   
## Pos Pred Value : 1.0   
## Neg Pred Value : 1.0   
## Prevalence : 0.5   
## Detection Rate : 0.5   
## Detection Prevalence : 0.5   
## Balanced Accuracy : 1.0   
##   
## 'Positive' Class : 0   
##   
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 24 0  
## 1 0 25  
##   
## Accuracy : 1   
## 95% CI : (0.9275, 1)  
## No Information Rate : 0.5102   
## P-Value [Acc > NIR] : 4.78e-15   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0000   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 1.0000   
## Prevalence : 0.4898   
## Detection Rate : 0.4898   
## Detection Prevalence : 0.4898   
## Balanced Accuracy : 1.0000   
##   
## 'Positive' Class : 0   
##   
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 26 0  
## 1 0 25  
##   
## Accuracy : 1   
## 95% CI : (0.9302, 1)  
## No Information Rate : 0.5098   
## P-Value [Acc > NIR] : 1.196e-15   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0000   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 1.0000   
## Prevalence : 0.5098   
## Detection Rate : 0.5098   
## Detection Prevalence : 0.5098   
## Balanced Accuracy : 1.0000   
##   
## 'Positive' Class : 0   
##   
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 25 0  
## 1 0 25  
##   
## Accuracy : 1   
## 95% CI : (0.9289, 1)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : 8.882e-16   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0   
## Specificity : 1.0   
## Pos Pred Value : 1.0   
## Neg Pred Value : 1.0   
## Prevalence : 0.5   
## Detection Rate : 0.5   
## Detection Prevalence : 0.5   
## Balanced Accuracy : 1.0   
##   
## 'Positive' Class : 0   
##   
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 25 0  
## 1 0 25  
##   
## Accuracy : 1   
## 95% CI : (0.9289, 1)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : 8.882e-16   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0   
## Specificity : 1.0   
## Pos Pred Value : 1.0   
## Neg Pred Value : 1.0   
## Prevalence : 0.5   
## Detection Rate : 0.5   
## Detection Prevalence : 0.5   
## Balanced Accuracy : 1.0   
##   
## 'Positive' Class : 0   
##

result\_acc

## [1] 1 1 1 1 1 1 1 1 1 1

mean(result\_acc)

## [1] 1

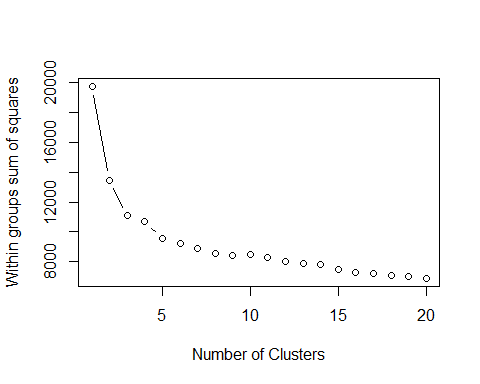
변수 v2과 v3를 이용하여 계층적 군집 분석을 시행하였으며 단일 연결법을 이용하여 계산하였습니다. 또한 v1에 대한 예측시 사용된 변수는 역시 v2와 v3를 이용하였고 랜덤포레스트를 적용한 결과 분류정확도는 100%를 보였습니다.

# 5번

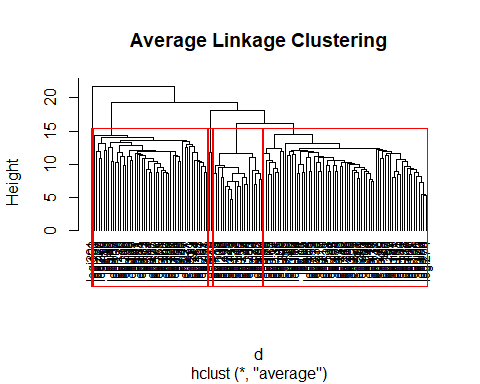
dat5 = read.csv('C:/Users/renz/Desktop/dat5.csv')   
party = read.csv('C:/Users/renz/Desktop/party.csv')   
party = party[,-3]  
  
party <- party$party   
party2 <- factor(party, labels=c("국민의당","더불어민주당","무소속", "바른정당","자유한국당","정의당"))  
  
dat5 <- dat5[,-1]  
  
rownames(dat5) = colnames(dat5)   
  
dat5 <- max(dat5) - dat5

국회의원들끼리 공동발의한 법안의수가 클수록 거리가 가깝다고 할 수 있으므로 거리상의 가까움으로 정보를 다시 표시하기 위해서 공동발의 법안수가 가장큰값으로 각각의 법안 발의수를 빼 주었다. max(dat5) - dat5

dat5\_z <- as.data.frame(lapply(dat5, scale))   
rownames(dat5\_z) = colnames(dat5\_z)  
  
d <- dist(dat5\_z)   
  
par(mfrow = c(1,1))  
  
wssplot <- function(data, nc=20, seed=1){   
 wss <- (nrow(data)-1)\*sum(apply(data,2,var))   
 for (i in 2:nc){  
 set.seed(seed)  
 wss[i] <- sum(kmeans(data, centers=i)$withinss)}  
 plot(1:nc, wss, type="b", xlab="Number of Clusters",  
 ylab="Within groups sum of squares")}  
  
wssplot(dat5\_z)

 군집수에 따른 집단 내 제곱합을 계산해본 결과 군집수를 5개를 기준으로 늘리거나 줄이는것이 좋아보인다.

# 계층적 군집분석 (평균연결법)  
set.seed(2)  
  
d=dist(dat5\_z)  
fit.average=hclust(d, method="average")  
plot(fit.average,hang=-1,cex=.8,main="Average Linkage Clustering")  
  
rect.hclust(fit.average,k=5)



cutree(fit.average, k=5)

## legi1 legi2 legi5 legi8 legi9 legi11 legi12 legi13 legi15   
## 1 2 2 1 2 1 2 1 3   
## legi16 legi17 legi18 legi19 legi20 legi23 legi24 legi26 legi27   
## 2 2 2 2 3 1 3 1 1   
## legi30 legi31 legi33 legi35 legi36 legi37 legi38 legi40 legi41   
## 1 1 2 2 2 3 1 1 1   
## legi42 legi43 legi46 legi50 legi51 legi57 legi60 legi62 legi64   
## 2 3 1 1 1 1 2 1 2   
## legi67 legi71 legi72 legi73 legi74 legi79 legi80 legi81 legi83   
## 1 1 2 2 2 4 2 2 1   
## legi84 legi85 legi87 legi89 legi92 legi94 legi95 legi101 legi106   
## 2 2 2 1 2 2 2 2 2   
## legi107 legi111 legi112 legi113 legi114 legi116 legi118 legi119 legi120   
## 1 2 2 2 3 3 2 1 2   
## legi126 legi128 legi131 legi132 legi133 legi134 legi135 legi137 legi139   
## 2 2 2 3 2 3 2 2 1   
## legi141 legi144 legi145 legi147 legi149 legi150 legi151 legi152 legi153   
## 2 3 2 2 1 2 2 1 3   
## legi155 legi156 legi158 legi159 legi160 legi161 legi168 legi171 legi175   
## 2 2 1 1 3 1 2 1 2   
## legi177 legi179 legi180 legi183 legi186 legi188 legi189 legi190 legi192   
## 1 1 1 3 1 2 2 2 3   
## legi195 legi196 legi199 legi200 legi203 legi206 legi207 legi209 legi210   
## 3 2 2 1 3 1 1 1 2   
## legi213 legi214 legi219 legi220 legi223 legi225 legi227 legi228 legi229   
## 2 1 4 1 2 3 1 2 1   
## legi233 legi235 legi236 legi239 legi241 legi242 legi244 legi246 legi248   
## 2 2 3 1 2 2 1 3 2   
## legi256 legi259 legi260 legi261 legi266 legi267 legi269 legi271 legi274   
## 2 2 1 2 2 2 1 1 2   
## legi278 legi280 legi284 legi285 legi288 legi301   
## 3 3 2 1 2 5

table(cutree(fit.average, k=5))

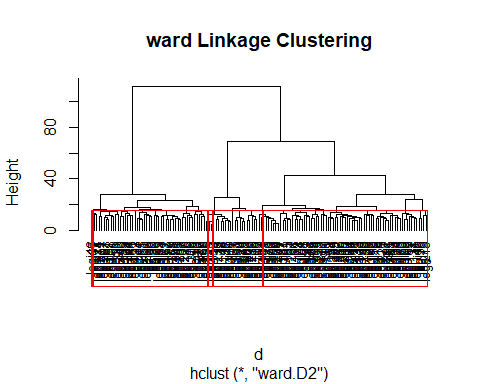
##   
## 1 2 3 4 5   
## 48 69 21 2 1

table(party2)

## party2  
## 국민의당 더불어민주당 무소속 바른정당 자유한국당   
## 24 67 1 7 41   
## 정의당   
## 1

계층적 군집분석 적용시 평균 연결법을 이용해보았습니다.

d=dist(dat5\_z)  
fit.ward=hclust(d, method="ward.D2")  
plot(fit.ward,hang=-1,cex=.8,main="ward Linkage Clustering")  
  
rect.hclust(fit.average,k=5)



cutree(fit.ward, k=5)

## legi1 legi2 legi5 legi8 legi9 legi11 legi12 legi13 legi15   
## 1 2 3 1 3 1 2 1 4   
## legi16 legi17 legi18 legi19 legi20 legi23 legi24 legi26 legi27   
## 5 3 2 2 4 1 4 1 1   
## legi30 legi31 legi33 legi35 legi36 legi37 legi38 legi40 legi41   
## 1 1 2 2 3 4 1 1 1   
## legi42 legi43 legi46 legi50 legi51 legi57 legi60 legi62 legi64   
## 3 4 1 1 1 1 5 1 2   
## legi67 legi71 legi72 legi73 legi74 legi79 legi80 legi81 legi83   
## 1 1 3 5 3 5 2 3 1   
## legi84 legi85 legi87 legi89 legi92 legi94 legi95 legi101 legi106   
## 3 2 3 1 2 2 5 3 3   
## legi107 legi111 legi112 legi113 legi114 legi116 legi118 legi119 legi120   
## 1 2 5 2 4 4 2 1 2   
## legi126 legi128 legi131 legi132 legi133 legi134 legi135 legi137 legi139   
## 2 2 5 4 3 4 2 3 1   
## legi141 legi144 legi145 legi147 legi149 legi150 legi151 legi152 legi153   
## 3 4 5 3 1 2 3 1 4   
## legi155 legi156 legi158 legi159 legi160 legi161 legi168 legi171 legi175   
## 3 2 1 1 4 1 5 1 2   
## legi177 legi179 legi180 legi183 legi186 legi188 legi189 legi190 legi192   
## 1 1 1 4 1 3 5 3 4   
## legi195 legi196 legi199 legi200 legi203 legi206 legi207 legi209 legi210   
## 4 3 3 1 4 1 1 1 2   
## legi213 legi214 legi219 legi220 legi223 legi225 legi227 legi228 legi229   
## 3 1 5 1 2 4 1 3 1   
## legi233 legi235 legi236 legi239 legi241 legi242 legi244 legi246 legi248   
## 3 2 4 1 2 2 1 4 2   
## legi256 legi259 legi260 legi261 legi266 legi267 legi269 legi271 legi274   
## 2 3 1 2 3 3 1 1 3   
## legi278 legi280 legi284 legi285 legi288 legi301   
## 4 4 2 1 2 4

table(cutree(fit.ward, k=5))

##   
## 1 2 3 4 5   
## 48 31 29 22 11

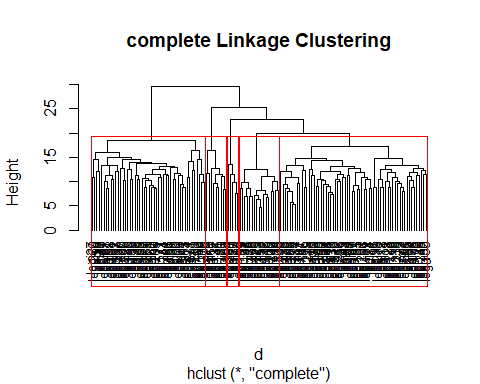
table(party2)

## party2  
## 국민의당 더불어민주당 무소속 바른정당 자유한국당   
## 24 67 1 7 41   
## 정의당   
## 1

clus <- cutree(fit.ward, k=5)  
ctable <- data.frame(colnames(dat5\_z), clus)

계층적 군집분석 적용시 와드 연결법을 이용해 보았습니다.

# 계층적 군집분석 (최장연결법)  
d=dist(dat5\_z)  
fit.complete=hclust(d, method="complete")  
plot(fit.complete,hang=-1,cex=.8,main="complete Linkage Clustering")  
  
rect.hclust(fit.complete,k=5)



cutree(fit.complete, k=5)

## legi1 legi2 legi5 legi8 legi9 legi11 legi12 legi13 legi15   
## 1 2 2 1 2 1 2 1 3   
## legi16 legi17 legi18 legi19 legi20 legi23 legi24 legi26 legi27   
## 2 2 2 2 4 1 3 1 1   
## legi30 legi31 legi33 legi35 legi36 legi37 legi38 legi40 legi41   
## 1 1 2 2 2 4 1 1 1   
## legi42 legi43 legi46 legi50 legi51 legi57 legi60 legi62 legi64   
## 2 3 1 1 1 1 5 1 2   
## legi67 legi71 legi72 legi73 legi74 legi79 legi80 legi81 legi83   
## 1 1 2 5 2 5 2 2 1   
## legi84 legi85 legi87 legi89 legi92 legi94 legi95 legi101 legi106   
## 2 2 2 1 2 2 2 2 2   
## legi107 legi111 legi112 legi113 legi114 legi116 legi118 legi119 legi120   
## 1 2 5 2 3 3 2 1 2   
## legi126 legi128 legi131 legi132 legi133 legi134 legi135 legi137 legi139   
## 2 2 5 3 2 3 2 2 1   
## legi141 legi144 legi145 legi147 legi149 legi150 legi151 legi152 legi153   
## 2 3 5 2 1 2 2 1 3   
## legi155 legi156 legi158 legi159 legi160 legi161 legi168 legi171 legi175   
## 2 2 1 1 3 1 5 1 2   
## legi177 legi179 legi180 legi183 legi186 legi188 legi189 legi190 legi192   
## 1 1 1 3 1 2 5 2 4   
## legi195 legi196 legi199 legi200 legi203 legi206 legi207 legi209 legi210   
## 3 2 2 1 3 1 1 1 2   
## legi213 legi214 legi219 legi220 legi223 legi225 legi227 legi228 legi229   
## 2 1 5 1 2 3 1 2 1   
## legi233 legi235 legi236 legi239 legi241 legi242 legi244 legi246 legi248   
## 2 2 3 1 2 2 1 3 2   
## legi256 legi259 legi260 legi261 legi266 legi267 legi269 legi271 legi274   
## 2 2 1 2 2 2 1 1 2   
## legi278 legi280 legi284 legi285 legi288 legi301   
## 3 4 2 1 2 4

table(cutree(fit.complete, k=5))

##   
## 1 2 3 4 5   
## 48 62 17 5 9

계층적 군집분석 적용시 최장 연결법을 이용해 보았습니다.