Lab 3 - Part 3

Group 1: Laxman Panthi & Zubin Shah

Overview:

This part is a continuation of Part 2 where we are going to use the clusters of the data to train a classifier to classify wines. Firstly, we will label the data frame that we have been using with the cluster labels and randomize the dataset. Later, train the classifier, plot the results and use the model to predict cluster labels for test dataset.

Part 3 Code:

Step 1:

Create the data frame that will include the cluster labels.

```
#Set-up to train a model for classification of wines

df <- data.frame(k=fit.km$cluster, df)
print(str(df))

## 'data.frame':    178 obs. of  14 variables:
##  $ k                   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Alcohol             : num  1.514 0.246 0.196 1.687 0.295 ...
##  $ Malic.acid          : num  -0.5607 -0.498 0.0212 -0.3458 0.2271 ...
##  $ Ash                 : num  0.231 -0.826 1.106 0.487 1.835 ...
```

```
##  $ Acl                 : num  -1.166 -2.484 -0.268 -0.807 0.451 ...
##  $ Mg                  : num  1.9085 0.0181 0.0881 0.9283 1.2784 ...
##  $ Phenols             : num  0.807 0.567 0.807 2.484 0.807 ...
##  $ Flavanoids          : num  1.032 0.732 1.212 1.462 0.661 ...
##  $ Nonflavanoid.phenols: num  -0.658 -0.818 -0.497 -0.979 0.226 ...
##  $ Proanth             : num  1.221 -0.543 2.13 1.029 0.4 ...
##  $ Color.int           : num  0.251 -0.292 0.268 1.183 -0.318 ...
##  $ Hue                 : num  0.361 0.405 0.317 -0.426 0.361 ...
##  $ OD                  : num  1.843 1.11 0.786 1.181 0.448 ...
##  $ Proline             : num  1.0102 0.9625 1.3912 2.328 -0.0378 ...
## NULL
```

Step 2:

From the above, it is clear we have a new column that indicates cluster (k). We will now randomize the dataset and review the upper head of the data frame.

```
#Randomize the dataset
rdf <- df[sample(1:nrow(df)), ]
print(head(rdf))
```

```
##     k    Alcohol Malic.acid         Ash        Acl         Mg      Phenols
## 93  2 -0.3826162 -0.7217931 -0.38826018  0.3608424 -1.3822227 -1.462188745
## 69  2  0.4180475 -1.2499245 -0.02375431 -0.7470867  0.7182523  0.375309174
## 13  1  0.9230815 -0.5427655  0.15849862 -1.0465271 -0.7520802  0.487156874
## 57  1  1.5020229 -0.5696196 -0.24245783 -0.9566950  1.2783790  1.445851440
## 117 2 -1.4542737 -0.7755014 -1.37242601  0.3907864 -0.9621277 -0.503494178
```

```
## 161 3 -0.7891070  1.3370245  0.04914686  0.4506745 -0.8220960  0.007809591
##     Flavanoids Nonflavanoid.phenols     Proanth  Color.int         Hue
## 93  -0.5699201            1.7528342  0.05084419 -0.8661960  0.01115870
## 69  -0.7301029            1.5117800 -2.04574255 -0.8144336  0.27365854
## 13   0.7315653           -0.5773564  0.38280376  0.2337547  0.84240820
## 57   0.9718395           -0.8184106  0.76717799  0.5702101 -0.07634125
## 117 -0.4297602           -0.4970050 -0.10639981 -1.3406845 -0.03259127
## 161 -1.1105371            1.1100230 -0.96250606  1.1180287 -1.73884025
##             OD      Proline
## 93  -0.7770322 -0.799896093
## 69  -0.9601332  0.009865569
## 13   0.4060824  1.819921051
## 57   0.9835550  0.708483475
## 117  1.0117244 -0.799896093
## 161 -1.4530976 -0.720507695
```

Step 3:

Create the train and test dataset by splitting the above randomized data. (80% - Train, remaining test). Review that in train and test the clusters are equally distributed.

```
train <- rdf[1:(as.integer(.8*nrow(rdf))-1), ]
test <- rdf[(as.integer(.8*nrow(rdf))):nrow(rdf), ]


table(train$k)
```

```
##

##  1  2  3

## 51 51 39

table(test$k)

##

##  1  2  3

## 11 14 12
```

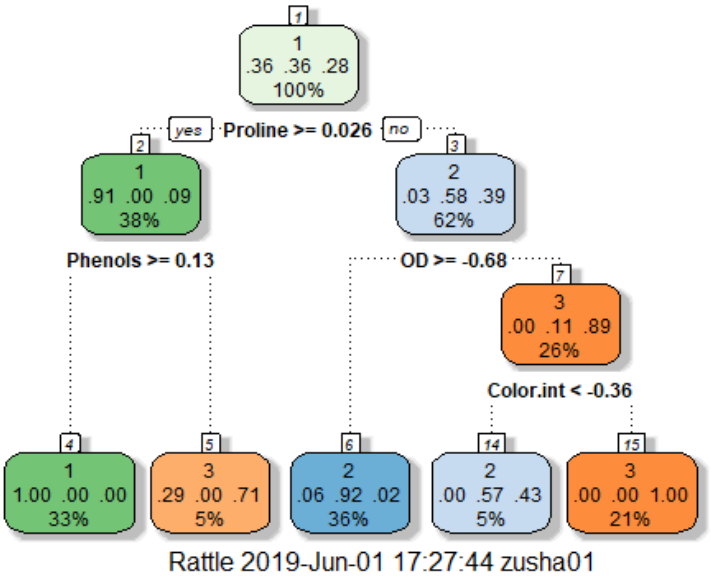
Step 4:

Since the data is evenly distributed, we will train the classifier and plot the results.

```
#Train the classifier and plot the results

fit <- rpart(k ~ ., data=train, method="class")

fancyRpartPlot(fit)
```



Step 5:

We will use the trained model to predict the test values and create confusion matrix to evaluate the accuracy.

```
#Now use the predict() function to see how well the model works
pred <- predict(fit, test, type="class")
print(table(pred, test$k))

##
## pred  1  2  3
##    1 11  1  0
##    2  0 12  1
##    3  0  1 11
```

Conclusion & Summary:

From the above confusion matrix, it is seen that we have got 91.89% prediction accuracy. Thus, we can conclude our model is a good predictor.