[Title Here, up to 12 Words, on One to Two Lines]

ALY 6015: INTERMEDIATE ANALYTICS

Dr. STEWARD HUNG

MAY 2, 2020

Abstract

In this assignment I have used sample (), ctree() and predict () functions for building and visualization of a decision tree. Findings on a k-means cluster and density-based cluster are explained in it.

Keywords: [data, mining, classification, clustering, iris, k-means, cluster]

[Title Here, up to 12 Words, on One to Two Lines]

Initially the required packages are installed using the install.packages () function and activated using library() function. I have performed the following on iris dataset (a) Decision tree model (b) Visualization of decision trees (c) Prediction and misclassification errors (d) K-means clustering (e) Density based clustering.

***Code:  
 install.packages('factoextra')***

***install.packages('NbClust')***

***library(factoextra)***

***library(NbClust)***

***library(MASS)***

# DECISION TREE MODEL

Iris dataset is already available in R so first step is to invoke the dataset. Let us check the structure of dataset to understand it.

***Code:***

***data('iris')***

***str(iris)***

Output:

'data.frame': 150 obs. of 5 variables:

$ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...

$ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...

$ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...

$ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...

$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...

So, iris dataset has 150 variables and 5 variables in which first 4 variables i.e. Sepal. Length,

Sepal. Width, Petal. Width and Petal. Length are numeric variables and the last variable (Species) is a categorical variable with 3 levels. Next step is to check the summary of iris dataset.

***Code: summary(iris)***

Output:

Sepal.Length Sepal.Width Petal.Length Petal.Width

Min. :4.300 Min. :2.000 Min. :1.000 Min. :0.100

1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600 1st Qu.:0.300

Median :5.800 Median :3.000 Median :4.350 Median :1.300

Mean :5.843 Mean :3.057 Mean :3.758 Mean :1.199

3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100 3rd Qu.:1.800

Max. :7.900 Max. :4.400 Max. :6.900 Max. :2.500

Species

setosa :50

versicolor:50

virginica :50

So, summary (iris) command minimum, Quartile 1, Median, Mean, Quartile 3 and maximum value for all the numerical variables and it calculates the number of observations for categorical or factor variables. Here we have 3 categorical variables i.e. setosa, verscicolor, virginica. 150 observations are equally divided into these 3 categorical variables. Each variables has 50 observations.

**Data Partition:**

***Code:***

***set.seed(555)***

***ind <- sample(2,***

***nrow(iris),***

***replace = TRUE,***

***prob = c(0.8,0.2))***

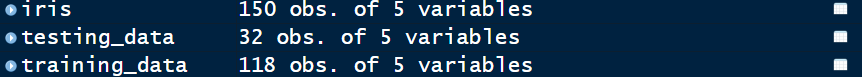
set.seed () is used to get the same samples in training and testing dataset. The independent sample is stored in ind. Size of sample is 2 and this sampling is with replacement. Then the probability is specified, and we have chosen 80% for training data and 20% for testing data.

***Code:***

***training\_data <- iris [ind == 1, ]***

***testing\_data <- iris [ind == 2, ]***

Within the independent samples where the values are 1 is used for the training data and for the testing data the (ind) is 2. So, training data has 118 observations and testing data has 32 observations. These values are not exactly divided into 80:20 ratio but they are very near to 80% and 20%.

******

For the decision tree model a package called (party) is installed which stands for partition as the while building the decision tree model, we constantly divide the datasets into 2 parts.

***Code:***

***tree <- ctree (Species~., training\_data)***

***tree***

Here ctree function is used where (c) stands for classification. Species here is taken as the dependent variable as we want to predict it. Data taken here is (training data) and the output is stores in (tree).

Output:  
 Condi inference tree with 4 terminal nodes

Conditional inference tree with 4 terminal nodes

Response: Species

Inputs: Sepal.Length, Sepal.Width, Petal.Length, Petal.Width

Number of observations: 118

1) Petal.Length <= 1.9; criterion = 1, statistic = 110.067

2)\* weights = 42

1) Petal.Length > 1.9

3) Petal.Width <= 1.7; criterion = 1, statistic = 50.039

4) Petal.Length <= 4.8; criterion = 0.998, statistic = 12.36

5)\* weights = 35

4) Petal.Length > 4.8

6)\* weights = 7

3) Petal.Width > 1.7

7)\* weights = 34

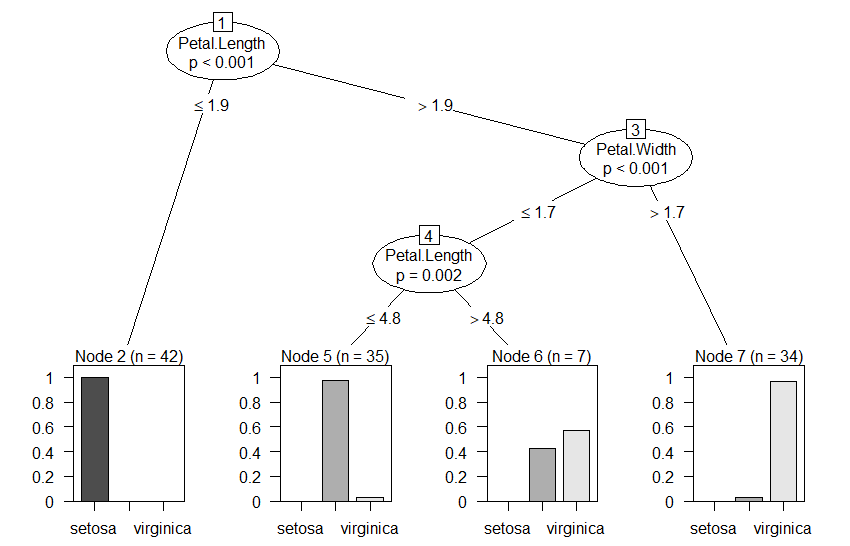
So, this is a Conditional Inference tree with 4 terminal nodes. Here, the response variable is species which is also a dependent variable and there are 4 inputs i.e. Sepal.Length. Sepal.Width, Petal.Length, Petal.Width .118 variables are taken has training data has 118 variables and we have 7 nodes here. The tree is not huge as it has only 4 input variables.

1. **VISUALIZATION**

The entire output and tree is explained below using a plot.

***Code:  
plot (tree)***

The model is called tree and so above code plots it.



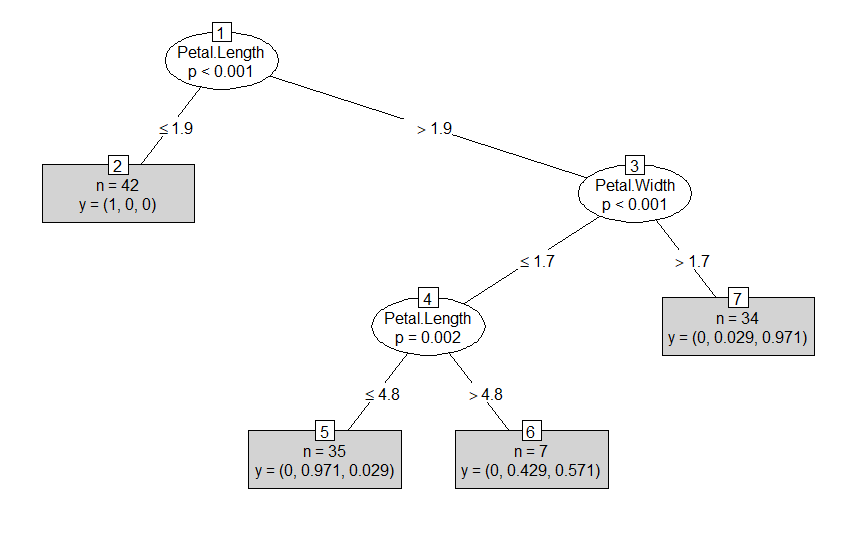
The most important variable which is (Petal. Length) is always at the top which helps to classify other 3 species. It is an inverted and upside tree as root node is at the top. There 4 nodes at the bottom which are called terminal nodes and decision can be made at that point. So that helps to decide and based on the data and the model whether we have first, second or third species. The probability of all three species is displayed in the plot above but only name of two species are available. Name of the species (versicolor) is missing but its probability is shown which is the middle bin in the plot. Here I will explain the decision of tree.

1. According to the data of *petal length* if it is less than *(1.9)* than we can tell that it is most likely to be the first species as it has very high probability which is almost *1*.
2. If the *petal length* is greater than *1.9*, then we need to look at the *petal width* also and if it is *higher than 1.7* than we can classify the observation to be belonging to the third species i.e. *virginica with very high probability.*
3. According to the data if the *petal length* is higher than 1.9 then we check the *petal width* and if it is *less than 1.7*, next step is to check *petal length* and if it is *greater than 4.8* than we see the third terminal node where outcome is not as clear as other 3 terminal nodes. It shows that probability of second species *versicolor is 40%* and probability of third species *(virginica) is 60%.* We end up classifying that observation has virginica but versicolor has probability 40% and there is not much difference. Such situation leads to a situation where we can have higher mis-classification error.
4. At the 4th node where *if petal length is less than or equal to 4.8* we can see that probability of second species *(versicolor) is higher* which is almost 1.

The above plot shows the terminal nodes and the probability of the species in a bar plot format. If we directly want to look at the exact number of probabilities of the species we can simplify the plot using the following code.

***Code:***

***plot (tree, type = 'simple')***

******

As we can see the exact probabilities of all three species are displayed here. So, using these trees we can visually see what is happening and interpretation becomes easy.

***Code:***

***head(training\_data)***

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

1 5.1 3.5 1.4 0.2 setosa

3 4.7 3.2 1.3 0.2 setosa

4 4.6 3.1 1.5 0.2 setosa

5 5.0 3.6 1.4 0.2 setosa

6 5.4 3.9 1.7 0.4 setosa

8 5.0 3.4 1.5 0.2 setosa

For the first observation we can see that petal length is 1.4 which is less than 1.9 so we move towards the second node where probability of setosa is highest and it is a correct classification.

***tail (training\_data)***

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

143 5.8 2.7 5.1 1.9 virginica

144 6.8 3.2 5.9 2.3 virginica

145 6.7 3.3 5.7 2.5 virginica

146 6.7 3.0 5.2 2.3 virginica

147 6.3 2.5 5.0 1.9 virginica

149 6.2 3.4 5.4 2.3 virginica

For the 143rd observation, the petal length is 5.1 which is greater than 1.9 so we move towards node 3 and check the petal width which is 1.9. It is greater than 1.7 thus we go to the 7 terminal node as shown in the plot. Here the probability of 3rd species is highest which is 0.971.

Here Species is the categorical variable but to check the result if one of the numerical or quantitative variable is categorical I have made another model.

***Code:  
tree1 <- ctree(Sepal.Length~.,training\_data)***

***tree1***

Output:

Conditional inference tree with 7 terminal nodes

Response: Sepal.Length

Inputs: Sepal.Width, Petal.Length, Petal.Width, Species

Number of observations: 118

1) Petal.Length <= 4.2; criterion = 1, statistic = 91.667

2) Petal.Length <= 3.3; criterion = 1, statistic = 26.444

3) Sepal.Width <= 3.2; criterion = 1, statistic = 18.176

4)\* weights = 17

3) Sepal.Width > 3.2

5) Sepal.Width <= 3.7; criterion = 0.981, statistic = 7.98

6)\* weights = 20

5) Sepal.Width > 3.7

7)\* weights = 7

2) Petal.Length > 3.3

8)\* weights = 15

1) Petal.Length > 4.2

9) Petal.Length <= 5.6; criterion = 1, statistic = 34.579

10) Sepal.Width <= 3; criterion = 0.967, statistic = 6.981

11)\* weights = 31

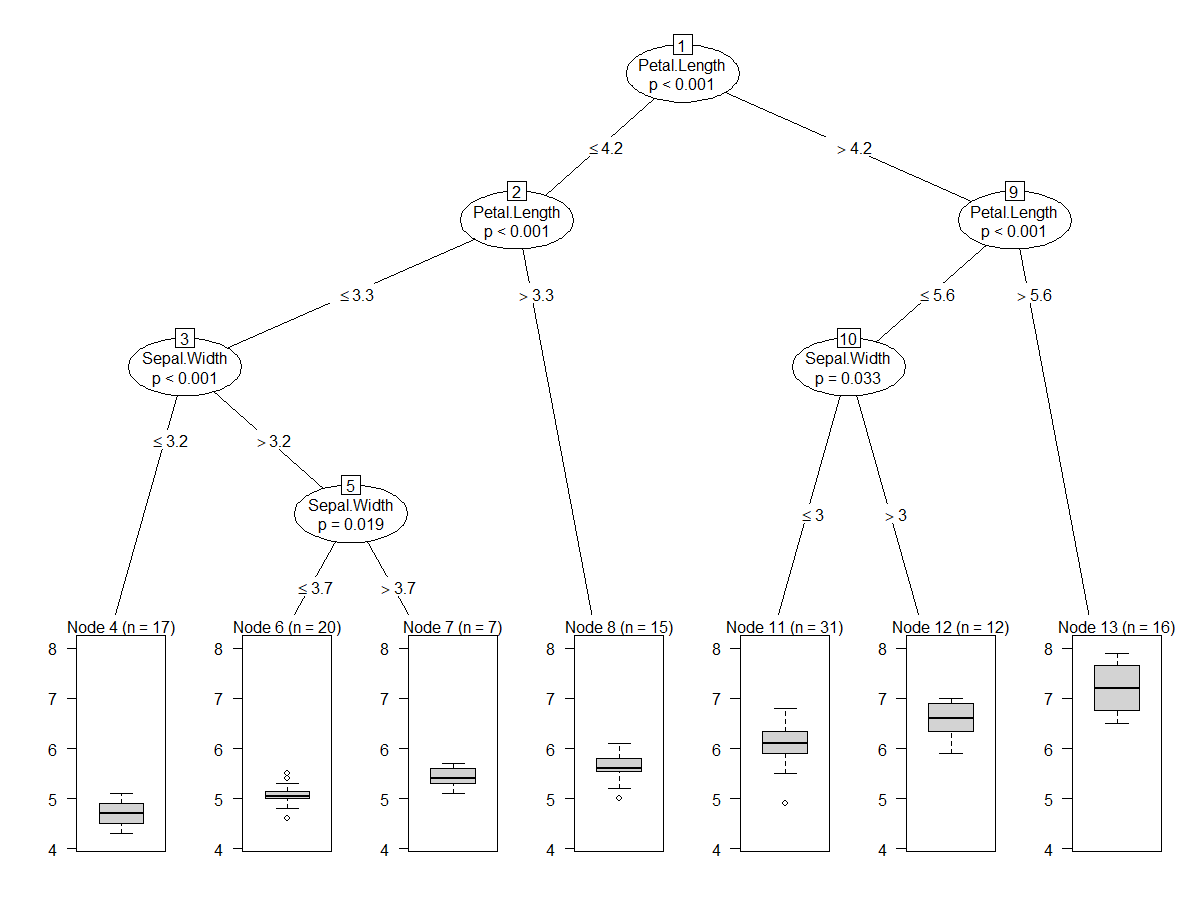
10) Sepal.Width > 3

12)\* weights = 12

9) Petal.Length > 5.6

13)\* weights = 16

So this is a conditional inference tree with 7 terminal nodes where response variable is (Sepal.Length) and input variables are Sepal. Width, Petal. Length, Petal. Width, Species.

Here we get a huge decision tree plot and as our dependent variable is quantitative variable we also get the boxplots in terminal nodes.

**[C – MISCLASSIFICATION ERROR]**

***Code:***

***predict(tree, training\_data)***

Here we have used predict () function for the prediction, model is tree and data is training data.

[1] setosa setosa setosa setosa setosa setosa

[7] setosa setosa setosa setosa setosa setosa

[13] setosa setosa setosa setosa setosa setosa

[19] setosa setosa setosa setosa setosa setosa

[25] setosa setosa setosa setosa setosa setosa

[31] setosa setosa setosa setosa setosa setosa

[37] setosa setosa setosa setosa setosa setosa

[43] versicolor versicolor virginica versicolor versicolor versicolor

[49] versicolor versicolor versicolor versicolor versicolor versicolor

[55] versicolor versicolor versicolor versicolor versicolor versicolor

[61] virginica versicolor virginica versicolor versicolor versicolor

[67] virginica versicolor versicolor versicolor versicolor versicolor

[73] versicolor versicolor versicolor versicolor versicolor versicolor

[79] versicolor versicolor virginica virginica virginica versicolor

[85] virginica virginica virginica virginica virginica virginica

[91] virginica virginica virginica virginica virginica virginica

[97] virginica virginica virginica virginica virginica virginica

[103] virginica virginica virginica virginica virginica virginica

[109] virginica virginica virginica virginica virginica virginica

[115] virginica virginica virginica virginica

Levels: setosa versicolor virginica

Here we can see that all the 118 and observations have predicted species. For example. 1 to 42 observation are setosa species, 43 to 81 are versicolor species and 82 to 118 are virginica species.

But we can also directly get the probabilities which is used for prediction using following code.

***predict(tree, training\_data, type = ‘prob’)***

The probability in last 5 observations are displayed below.

[[113]]

[1] 0.00000000 0.02941176 0.97058824

[[114]]

[1] 0.00000000 0.02941176 0.97058824

[[115]]

[1] 0.00000000 0.02941176 0.97058824

[[116]]

[1] 0.00000000 0.02941176 0.97058824

[[117]]

[1] 0.00000000 0.02941176 0.97058824

[[118]]

[1] 0.00000000 0.02941176 0.97058824

As we can see for the 118th observation there are probabilities for all 3 species and in 118th observation probability of 3rd species is 0.97 which is highest.

MISCLASSIFICATION ERROR

***Code:  
predection1 <- predict(tree, training\_data)***

***table1 <- table (Predicted = predection1, Actual = training\_data$Species)***

***table1***

***1 – sum (diag(table1))/sum(table1)***

Predict () function is used to predict the values. Model used is tree model and dataset is training data. Let us store the predictions in prediction 1. These predictions are final in case of first, second and third species. Next step is to make a table with prediction 1 and traning\_data$Species and store it in table 1. Here (prediction1) is the predicted value and the (training\_data$Species) is the actual values.

Ouput:

Predicted setosa versicolor virginica

setosa 42 0 0

versicolor 0 34 1

virginica 0 4 37

[1] 0.04237288

So, there are 42 observations for first species as predicted by the decision tree model. So, we have correct classification. Also, the observation for second and third species is 34 and 37. There are also some misclassifications. There is 1 observation which belongs to virginica species but it seems that it belongs to versicolor species. Similarly, there are 4 species which belongs to versicolor species and but seems to belong to the virginica species.

Thus, we calculate the misclassification error and that is 4% so the accuracy of model is about 96%.

Now we will calculate the misclassification error for testing data

***Code:***

***prediction2 <- predict(tree,testing\_data)***

***table2 <- table(Predicted = prediction2,Actual = testing\_data$Species)***

***table2***

***1 - sum(diag(table2))/sum(table2)***

The predictions are stored in (prediction2). Predictions are done using predict function. Tree model and testing data is used. Then a table is made with prediction2 and testing\_data$Species which is stored in table2.

Output:

Actual

Predicted setosa versicolor virginica

setosa 8 0 0

versicolor 0 11 0

virginica 0 1 12

[1] 0.03125

As we can see in the output only 1 misclassification error is there. 1 observation which belongs to versicolor species seems to be currently in virginica category. So, we calculate the error and it is 0.03125. This means that there is only 3% error and the accuracy of the model is 97%.

**[D – K MEANS CLUSTERING]**

Here we do not want to predict the species so we will remove it as we are trying to find the structure of rest of the features or numeric variables.

***Code:  
Iris.features <- iris***

***Iris.features$Species <- NULL***

So, here I have stored the dataset in iris.features and removed the species column. Following code will perform K-means clustering.

***Code:***

***result <- kmeans(Iris.features, 3)***

***result***

Here, kmeans() function is used what takes 2 parameters i.e. the dataset (Iris.features) and total number of species i.e. 3. These results are stored in (result).

Output:

K-means clustering with 3 clusters of sizes 50, 62, 38

Cluster means:

Sepal.Length Sepal.Width Petal.Length Petal.Width

1 5.006000 3.428000 1.462000 0.246000

2 5.901613 2.748387 4.393548 1.433871

3 6.850000 3.073684 5.742105 2.071053

Clustering vector:

[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 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

[71] 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3

[106] 3 2 3 3 3 3 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 2 2 3 3 3 3 3 2 3 3 3 3 2 3

[141] 3 3 2 3 3 3 2 3 3 2

Within cluster sum of squares by cluster:

[1] 15.15100 39.82097 23.87947

(between\_SS / total\_SS = 88.4 %)

Available components:

[1] "cluster" "centers" "totss" "withinss"

[5] "tot.withinss" "betweenss" "size" "iter"

[9] "ifault"

The output shows us 3 clusters each with the size of 50, 62 and 38 respectively. Also, we have got the mean Sepal.Length, Sepal.Width, Petal.Length and Petal.Width of all 3 species. Cluster vector shows the observation and its cluster number. For example first observation is in cluster 1 and last observation is in cluster 2. Then we have within cluster sum of squares by cluster and the available components. We can check the components using ***result$size, result$cluster.***

A table is created using the original iris dataset and results of Iris.features to check the species and cluster and how much they match.

***Code:***

***table(iris$Species,result$cluster)***

Output:

1 2 3

setosa 50 0 0

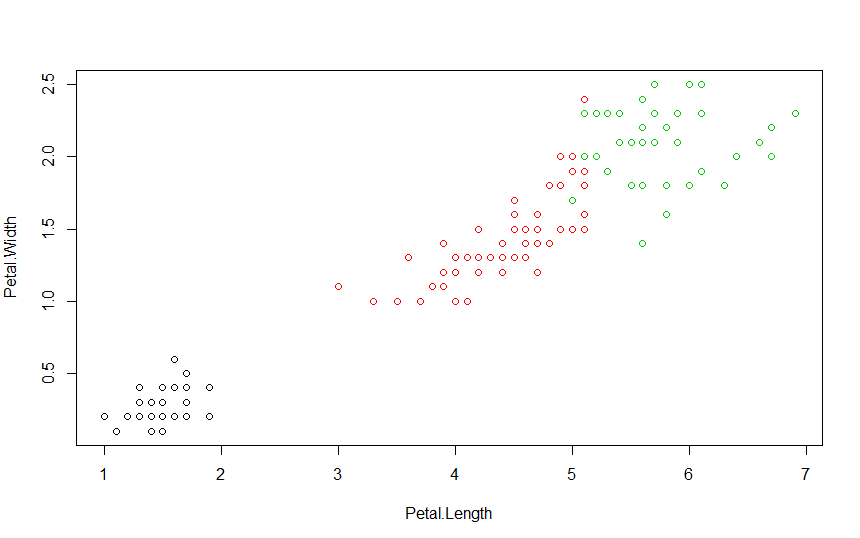
versicolor 0 48 2

virginica 0 14 36

Cluster 2 has 2nd and 3rd species 48,14 respectively while cluster 3 has 2,36 species respectively. Cluster 1 has only species of setosa.

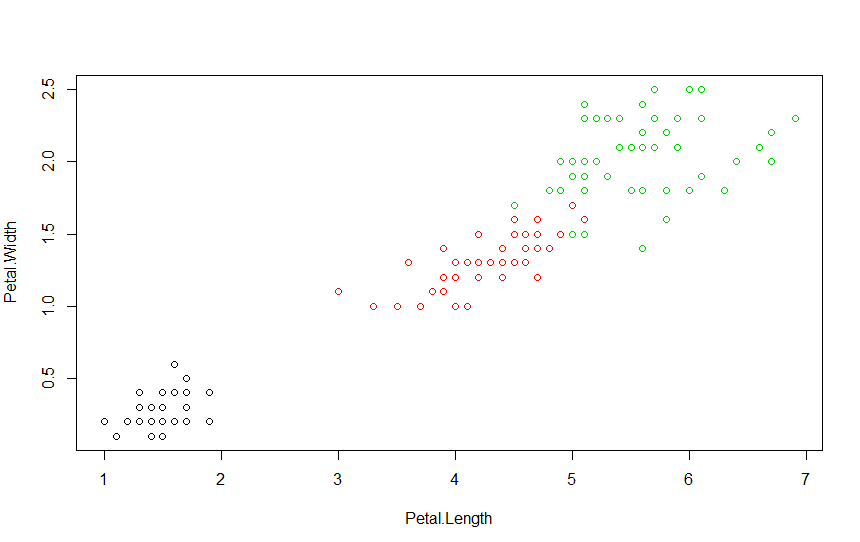
Visualzation:

***plot(iris[c("Petal.Length","Petal.Width")],col = result$cluster)***



Here the color will be based on the cluster. The above plot shows the colors of each cluster i.e. black, red and green. Groupings are almost accurate. X- axis has Petal Length and Y- axis has Petal Width. Now we will check the accuracy based on the species.

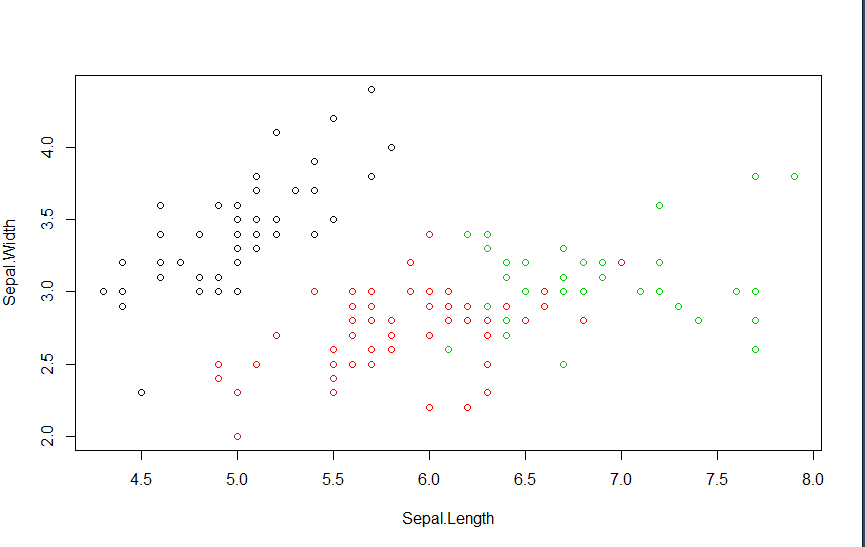
***plot(iris[c("Petal.Length","Petal.Width")],col = iris$Species)***

******

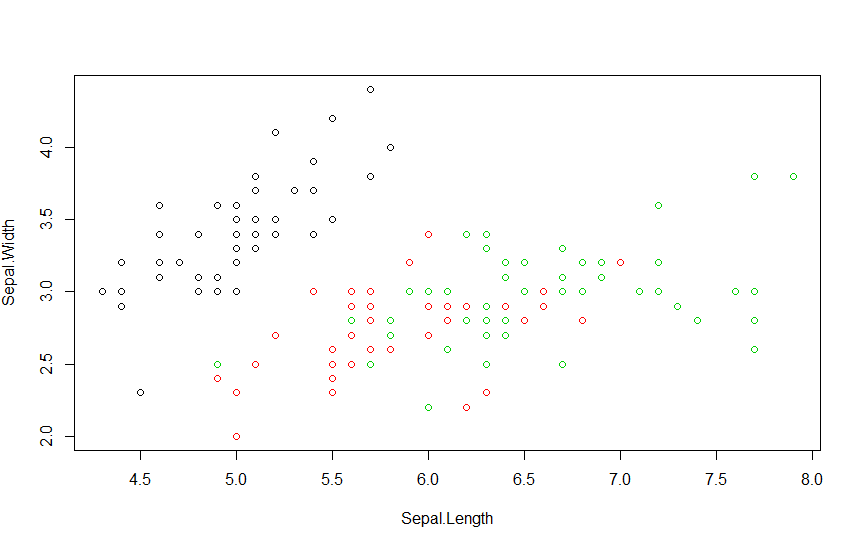
So, this cluster is also same with only minor changes.

Below is another plot with Sepal length on X-axis and Sepal Width on Y-axis.

***plot (iris[c ("Sepal.Length","Sepal.Width")],col = result$cluster)***



***plot(iris[c("Sepal.Length","Sepal.Width")],col = iris$Species)***

******

There is some overlapping between the groups and seems that k-means did effective work. Here, k-means identified the number of centroids and allocated every datapoint to the nearest cluster and keeps the centroids as small as possible.

[**E – DENSITY BASED CLUSTERING]**

***Code:***

***install.packages(fpc)***

***library(fpc)***

***install.packages(dbscan)***

***library(dbscan)***

***str(iris)***

***new <- iris [, -5]***

***new***

First step is to install required packages and activating it i.e. fpc and dbscan. Iris dataset is stored in (new) and the species column is removed. So new dataset has 150 observations with 4 columns.

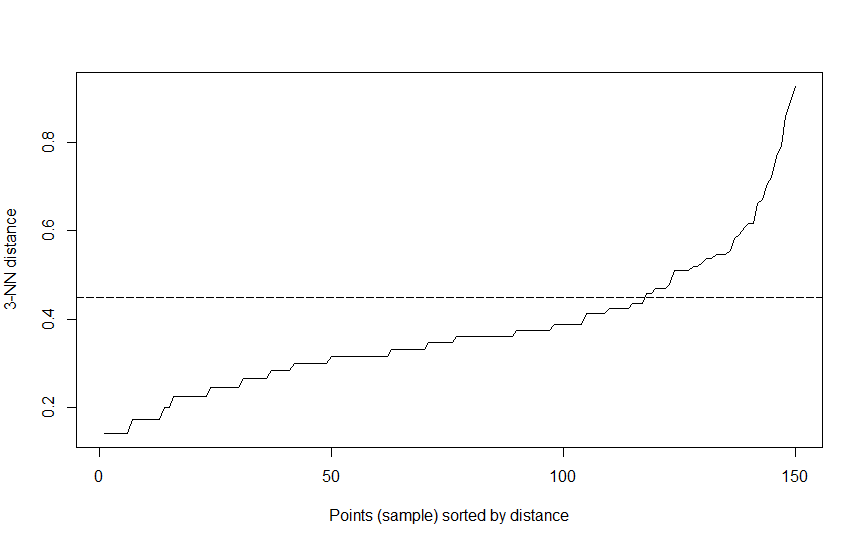
Before doing density based clustering we need to find optimal level.

***Code:***

***kNNdistplot(new, k =3)***

***abline(h = 0.45,lty = 5)***

Here kNNdistplot is used to visually see where the eps optimal value is. Data used is new dataset and k is 3.



As the graph shows the point where the value gradually increases, and it takes off is 0.45 and it could be optimal eps value. So a line is added using abline(). EPS value provides the maximum distance. If the eps value is low the clusters are defined as noise and if the eps value is high the denser clusters may be merged together.

DENSITY BASED CLUSTERING:

***Code:***

***set.seed(123)***

***f <- fpc::dbscan(new, eps = 0.45, MinPts = 4)***

***f***

Here the seed is set for consistency. Results from density based clustering from fpc is stored in f. So, fpc is referred and then dbscan function is called. Dbscan exists both for fpc and dbscan so here it is specified that dbscan is for fpc. Data used is new, eps value used is 0.45 and minpts which is the reachability of the minimum points within the distance of eps 0.45.

Output:

dbscan Pts=150 MinPts=4 eps=0.45

0 1 2 3

border 17 3 10 3

seed 0 45 71 1

total 17 48 81 4

So, here we have 150 observations and 3 clusters are formed. Cluster 0 denotes the noise. There are 17 points which are considered as noise. Cluster 1 has total 48 points, cluster 2 points has 81 points and cluster 3 has 4 points. Here border and seed related information is also specified. Similarly we can do density based clustering using dbscan package.

***Code:***

***d <- dbscan::dbscan(new, eps = 0.45, 4)***

***d***

dbscan stands for density based spatial clustering of applications with noise. Rest of the parameters takes same value.

Output:  
DBSCAN clustering for 150 objects.

Parameters: eps = 0.45, minPts = 4

The clustering contains 3 cluster(s) and 17 noise points.

0 1 2 3

17 48 81 4

Available fields: cluster, eps, minPts

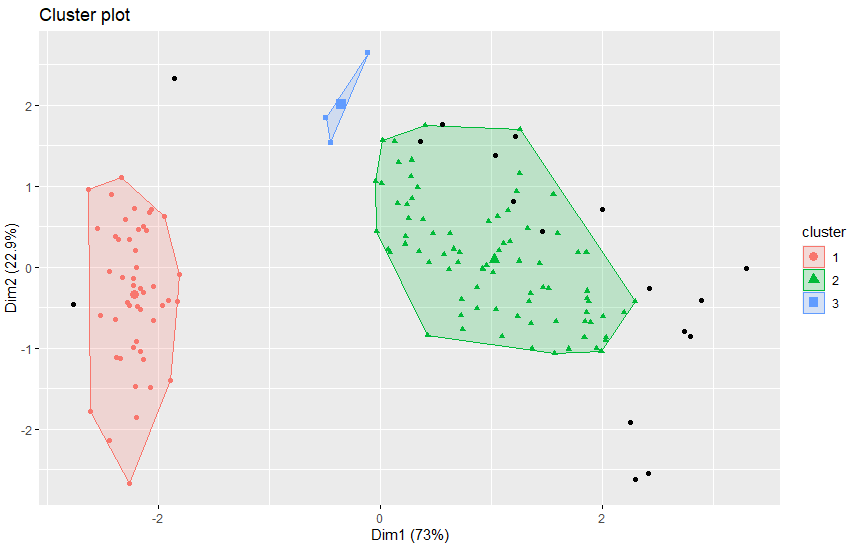
Here dbscan provides only the summary so output with fpc is better.

VISUALIZATION:

***Code:***

***fviz\_cluster(f, new, geom = "point")***

fviz\_cluster is used for the visualization of cluster plot. Information stored in (f) is used, dataset used is (new).



Here we can see 3 clusters. The first cluster is red one which has 48 points, second cluster has green color and its points are green with 81 points and third cluster has just 4 points in blue color.

17 points are noise and are in black color which does not belong to any cluster.

References

Garbade, M. J. (2018, September 12). Understanding K-means Clustering in Machine Learning. Retrieved from https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1

.