CLV kmeans clustering

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library (ggplot2)

clvdata <- read.csv("clvdata.csv") # load the clvdataset.csv
```

Step 2 – group the dataset by Customer_ID in order to get customer totals and Recency of orders.

```
RFM <- group_by (clvdata, CustomerID) %>% summarise( Monetary= sum(Revenue), Frequency = n(), Recency =
```

Step 3 – normalize the RFM data – This may not always be needed but we are going to cluster this dataset and clustering needs items to be scaled appropriately. We only need columns 2,3,4 normalized hence we identify them in the function.

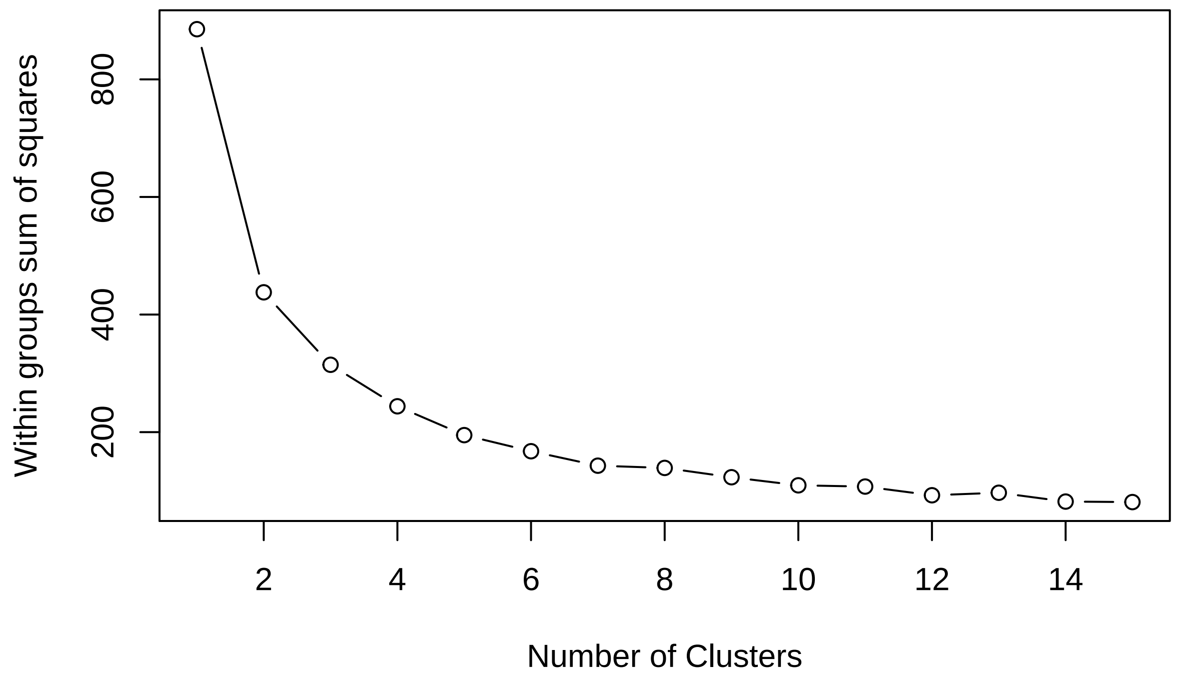
```
normalize <- function(x) { return (((x - min(x)) / (max(x) - min(x))) * 5)}
dfNorm <- as.data.frame(lapply(RFM[2:4], normalize))
```

Step 4 – run a quick k-means to identify and decide on a number of clusters in the data. We will use the familiar elbow plot to decide on the number of clusters. We use the within groups sum of squares to draw an elbow plot (testing between 2-15 clusters) and then use the elbow plot to make decisions on the number of clusters to extract.

```
wss <- (nrow(dfNorm)-1)*sum(apply(dfNorm,2,var))

for (i in 2:15) wss[i] <- sum(kmeans(dfNorm,centers=i)$withinss)

plot(1:15, wss, type="b", xlab="Number of Clusters", ylab="Within groups sum of squares")
```



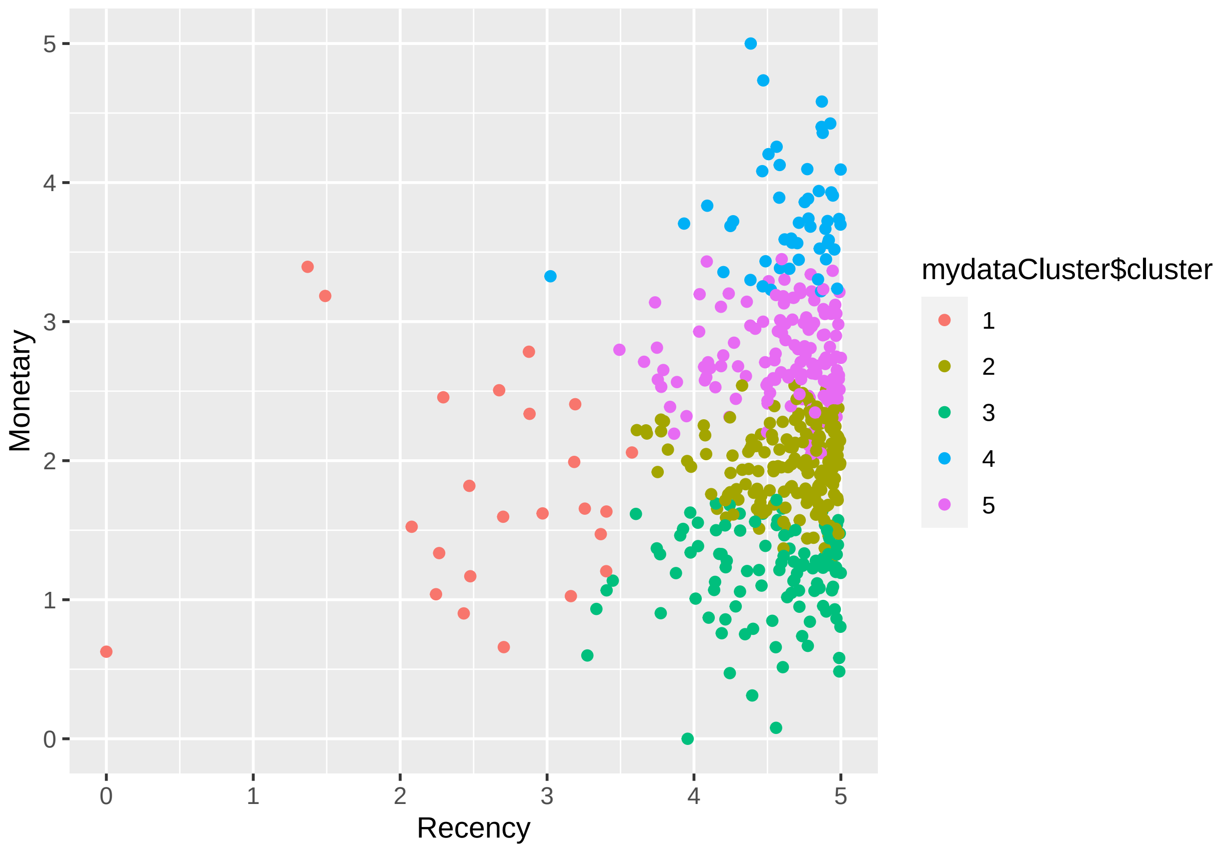
Step 5 – run k-means again with appropriate clusters and label the clusters as a factor to the dataset. I used 5 clusters guided by the elbow plot in the previous step. Note – the nstart option (set in the code below to 25) is a way to help the clustering process have more initial configurations and allow the function to report the one that is best (or more stable). In this case, adding nstart=25 will generate 25 initial random centroids and choose the best one for the algorithm. We store the cluster membership in another column called cluster.

```
mydataCluster <- kmeans(dfNorm, 5, nstart = 20)

mydataCluster$cluster <- as.factor(mydataCluster$cluster)
```

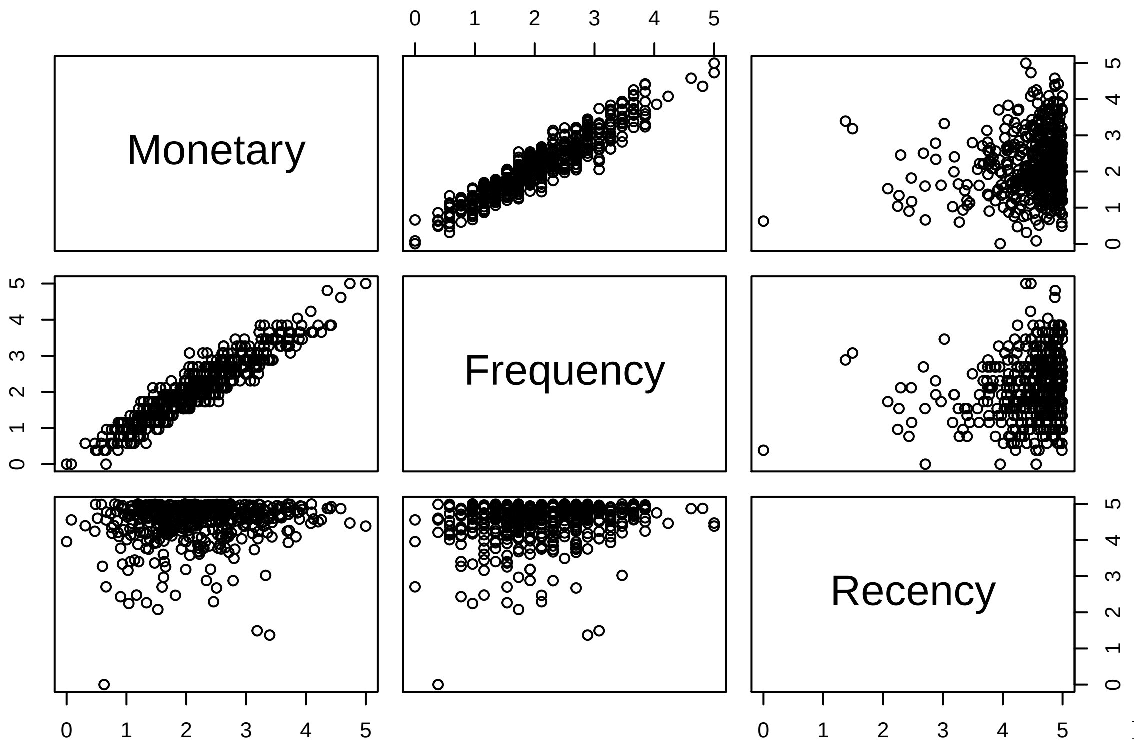
Step 6 – Plot the results to see cluster memberships.

```
ggplot(dfNorm, aes(Recency, Monetary, color = mydataCluster$cluster)) + geom_point()
```



Step 7 – run a pairs plot to see patterns in R F M , the pairs function allows the visualization of more than two variables at a time.

```
pairs (dfNorm)
```



Interpretation 1) The order information provided was processed and converted into recency, frequency and monetary figures. This was accomplished by grouping the dataset by using customerID, taking order date information to identify days between orders (recency), and summing order amounts to get monetary. 2) We tested the possibility of clustering the results obtained by checking the within group sumsofsquares with an elbow plot to guide the extraction of a specific number of clusters. In this dataset, 5 clusters appear appropriate. 3) We extracted five clusters - understandably we can see that one cluster stands as the most profitable with both high recency and high monetary values, whereas another rather dispersed clusters reveals customers with low recency and low monetary values. The other three clusters are somewhere in the middle as was to be expected. As a manager, you now have the ability to use the cluster membership stored in the dataset to treat each of these groups separately. 4) We used the pairs plot function to see interactions between Recency, Frequency and Monetary.