# BDBA: Weekly Assignment 3

*Rutger Smid, 487148 & Loic Roldan Waals, 409763*

*9-4-2018*

## Question 1

### (a) Compute the Centroids

We first compute the X1 and X2 values for the two centroids as follows:

**Computing Centroid 1:**

X1 = (1+5+6)/3 = 4

X2 = (4+1+2)/3 = 2.33

**Computing Centroid 2:**

X1 = (1+0+4)/3 = 1.66

X2 = (3+4+0)/3 = 2.33

Thus *Centroid 1* has the values of 4 and 2.33 for X1 and X2 respectively. For *Centroid 2* these values are 1.66 and 2.33

### (b) Reassigning Observations

First the distance of each observation to the two centroids is calculated. The distance from *Centroid 1* to *observation 1* is calculated with: sqrt((4 - 1)^2 + (2.33 - 4)^2) = 3.43. The same method is used to calculate the distance to *Centroid 2*, which gives us a value of 1.80. This allows us to conclude that *observation 1* is closest to *Centroid 2* and thus it will be assigned to it.

The same method is used for all the other observations, the new cluster labels are:

- observation 1 = Centroid 2
- observation 2 = Centroid 2
- observation 3 = Centroid 2
- observation 4 = Centroid 1
- observation 5 = Centroid 1
- observation 6 = Centroid 1

### (c) Recalculating Cluster Centroids

**Computing Centroid 1:**

X1 = (5+6+4)/3 = 5

X2 = (1+2+0)/3 = 1

**Computing Centroid 2:**

X1 = (1+1+0)/3 = 0,66

X2 = (4+3+4)/3 = 3,66

When reassigning the observations to the centroids we find that none changed. This means that the final values for *Centroid 1* are X1 = 5, X2 = 1 and that the final values for *Centroid 2* are X1 = 0.66, X2 = 3.66. The cluster labels are the same as in **Question 1 (a)**:

- observation 1 = Centroid 2
- observation 2 = Centroid 2
- observation 3 = Centroid 2
- observation 4 = Centroid 1
- observation 5 = Centroid 1
- observation 6 = Centroid 1

# Question 2

Accuracy is a simple performance measure and not suitable for classifying imbalanced data sets. Accuracy simply divides the number of correct predictions by the total number of predictions. Although it can give an indication of the quality of the model, it should always be compared with the baseline accuracy of a model/data set. Let's say you want to build a model which diagnoses whether a patient has disease X. It might not be a common disease so 999 out of 1000 patients will not have this disease, this is an example of an imbalanced data set. Then, a model which simply predicts that no patients have this disease will yield an accuracy of 99.9%. This sounds impressive, but the model is not very useful in practice. Looking back at Week 1's assignment, the final model had a accuracy of 70% by predicting *US* for all observations. This was simply due to the fact that 70% of the population would travel to the US.

An implication of this problem is that for imbalanced data sets, another performance metric should be used. A solution to this problem is the *Receiver Operating Characteristic* (ROC) curve which allows the comparison of models without knowing the class distribution. The ROC curve is plotted using thee true positive rate (TPR) against the false positive rate (FPR). The ROC curve allows us to compare different models and interpret their performance easily, even in the case of imbalanced data sets.

# Question 3

First we will collect all the tweets. We searched for the 500 most recent English tweets containing *blockchain, analytics* and *artificial reality*. After collecting the tweets they are exported as a CSV file in case we want to use the same tweets for the future. Finally we clean the data and prepare it for visualization. Only 500 tweets were used per topic due to limited computing power. In the section where we remove specific words, the following procedure was followed:

1. create the word cloud
2. check for any word that does not add meaning (e.g. "thanks")
3. generate the word cloud again and repeat the steps

Additionally the words that were used in the search queries were also removed as all the searched tweets contain the searched words. Names of twitter accounts were also deleted. The code below is used to generate the data sets of tweets:

```
library(twitteR)
library(tm)
```

```
## Loading required package: NLP
```

```r
setwd("C:/Users/Loic RW/Google Drive/Big Data and Business Analytics/Assignments/Assignment 3")

#set up necessary credentials
ck <- "EUENyJUa8hrF1UGX2hGtntsAt"
cs <- "UmARrjPzF2uTGOpzwmgoEvfDwT8kaIUjsiBr9ZLeV8bhfmWH5j"
at <- "384339592-ENpOhuPiFt1kCAGZQzo44kH8C8aaGgEkhFTPz38x"
as <- "gncNiTOeBvz52QIGdubR1Z2pjSneIWcqAKXajiVEKLvw1"

setup_twitter_oauth(ck, cs, access_token = at, access_secret = as)
```

```
## [1] "Using direct authentication"
```

```r
#collect the stream of tweets and storing it as a dataframe
t_stream <- searchTwitter('blockchain', resultType="recent", n=500, lang = "en")
blockchainTweets <- do.call("rbind", lapply(t_stream, as.data.frame))

t_stream <- searchTwitter('analytics', resultType="recent", n=500, lang = "en")
analyticsTweets <- do.call("rbind", lapply(t_stream, as.data.frame, lang = "en"))

t_stream <- searchTwitter('artificial intelligence', resultType="recent", n=500)
AITweets <- do.call("rbind", lapply(t_stream, as.data.frame))


#we save the files in case we want to analyse these tweets at later date
write.table(blockchainTweets, "blockchainTweets.csv", row.names = FALSE, col.names = TRUE, sep =";")
write.table(analyticsTweets, "analyticsTweets.csv", row.names = FALSE, col.names = TRUE, sep =";")
write.table(AITweets, "AITweets.csv", row.names = FALSE, col.names = TRUE, sep =";")


#only select the columns we will use.
blockchainTweets <- subset(blockchainTweets, select=c("text","created",
                                                      "screenName", "isRetweet","id"))
analyticsTweets <- subset(analyticsTweets, select=c("text","created",
                                                    "screenName", "isRetweet","id"))
AITweets <- subset(AITweets, select=c("text","created",
                                      "screenName", "isRetweet","id"))

#Before we can analyse the text we need to clean all the data, afterwards we create a wordcloud,
#this process is done for all 3 themes.

#this is for blockchain
#remove quotation marks
blockchainTweets[,1] <- gsub('"',"",blockchainTweets[,1])
#remove emoji's
blockchainTweets$text = sapply(blockchainTweets$text, function(row) iconv(row, "latin1", "ASCII", sub =
#create character vector
blockchainTweets_text = c(blockchainTweets$text)
#create corpus
tweets_corpus = Corpus(VectorSource(blockchainTweets_text))
#remove punctuation
tweets_corpus_clean = tm_map(tweets_corpus, removePunctuation)
#only lower case
tweets_corpus_clean = tm_map(tweets_corpus_clean, content_transformer(tolower))
```

```
#remove english stopwords
tweets_corpus_clean = tm_map(tweets_corpus_clean, removeWords, stopwords("english"))
#remove numbers
tweets_corpus_clean = tm_map(tweets_corpus_clean, removeNumbers)
#remove unnecessary white space
tweets_corpus_clean = tm_map(tweets_corpus_clean, stripWhitespace)
#remove obvious or nonsensical words
cleanBlockchain = tm_map(tweets_corpus_clean, removeWords, c("blockchain", "https", "murthaburke", "amp


#this is for analytics
#remove quotation marks
analyticsTweets[,1] <- gsub('"','',analyticsTweets[,1])
#remove emoji's
analyticsTweets$text = sapply(analyticsTweets$text, function(row) iconv(row, "latin1", "ASCII", sub = "
#create character vector
analyticsTweets_text = c(analyticsTweets$text)
#create corpus
tweets_corpus = Corpus(VectorSource(analyticsTweets_text))
#remove punctuation
tweets_corpus_clean = tm_map(tweets_corpus, removePunctuation)
#only lower case
tweets_corpus_clean = tm_map(tweets_corpus_clean, content_transformer(tolower))
#remove english stopwords
tweets_corpus_clean = tm_map(tweets_corpus_clean, removeWords, stopwords("english"))
#remove numbers
tweets_corpus_clean = tm_map(tweets_corpus_clean, removeNumbers)
#remove unnecessary white space
tweets_corpus_clean = tm_map(tweets_corpus_clean, stripWhitespace)
#remove obvious or nonsensical words
cleanAnalytics = tm_map(tweets_corpus_clean, removeWords, c("analytics", "https", "thanks", "mention",


#this is for AI
#remove quotation marks
AITweets[,1] <- gsub('"','',AITweets[,1])
#remove emoji's
AITweets$text = sapply(AITweets$text, function(row) iconv(row, "latin1", "ASCII", sub = ""))
#create character vector
AITweets_text = c(AITweets$text)
#create corpus
tweets_corpus = Corpus(VectorSource(AITweets_text))
#remove punctuation
tweets_corpus_clean = tm_map(tweets_corpus, removePunctuation)
#only lower case
tweets_corpus_clean = tm_map(tweets_corpus_clean, content_transformer(tolower))
#remove english stopwords
tweets_corpus_clean = tm_map(tweets_corpus_clean, removeWords, stopwords("english"))
#remove numbers
tweets_corpus_clean = tm_map(tweets_corpus_clean, removeNumbers)
#remove unnecessary white space
tweets_corpus_clean = tm_map(tweets_corpus_clean, stripWhitespace)
#remove obvious or nonsensical words
```

```
cleanAI = tm_map(tweets_corpus_clean, removeWords, c("facebook", "httpstcokrux", "intelligence", "artif
```

## (a) Graphical Representation of the Tweets

In this section we make a word cloud of the tweets, to maintain readability only 50 words are displayed at the same time. In the *blockchain* tweets we see that most tweets concern cryptocurrencies with words like *cryptocurrency, crypto* (shorthand for many cryptocurrencies) and *bitcoin*.

The *analytics* tweets we see a focus on data, facebook and academic research. Here one of the downsides of a word cloud comes in to play: we do not know when combinations of words are used (e.g. are *academic* and *research* appearing independently or together). This can be bothersome when interpreting the results.

The *artificial intelligence* tweets show a more varied distribution of words. Again here we see words that are often used together such as *machine learning*. In cases like these, having domain knowledge on the subject your analysing can help tremendously. Other words indicate that AI is an upcoming phenomenon, this is indicated by words like *just, first, new* and *opportunity*.

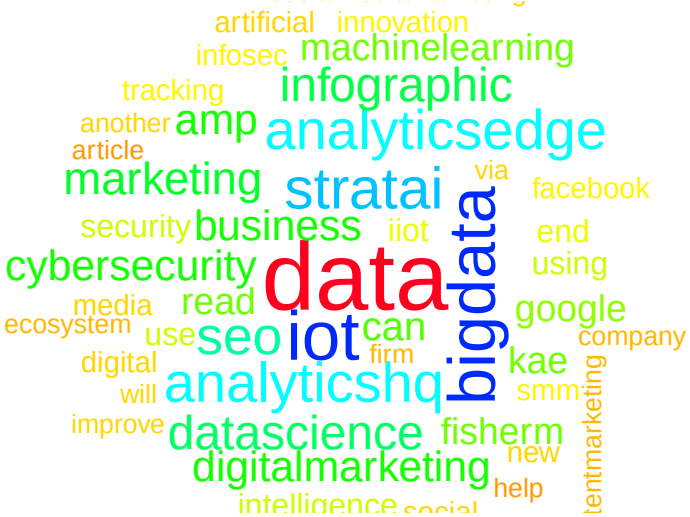
Additionally we wanted to add a geographical analysis of which places which topics were talked about the most, but our method for collecting the tweets yielded us with missing values for the *longitude* and *latitude* variables, making this analysis impossible.

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
wordcloud(cleanBlockchain, random.order = F, max.words = 50, scale = c(3, 0.5), colors = rainbow(50))
```

```
wordcloud(cleanAnalytics, random.order = F, max.words = 50, scale = c(3, 0.5), colors = rainbow(50))
```




```
wordcloud(cleanAI, random.order = F, max.words = 50, scale = c(3, 0.5), colors = rainbow(50))
```


### (b) Three User Timelines

In **Question 3, Part (a)** we found that several users were very present in discussing the topics we had selected. As they seem to be very active users we will be analysing their timelines. The users that we will investigate are *cubeyou, userexperienceu* and *bethereumteam.*

```
#get the timelines
cubeTimeline <- userTimeline("cubeyou")
userexperienceTimeline <- userTimeline("userexperienceu")
bethereumTimeline <- userTimeline("bethereumteam")

#put them into a data frame
dfcube <- twListToDF(cubeTimeline)
dfuser <- twListToDF(userexperienceTimeline)
dfbethereum <- twListToDF(bethereumTimeline)

#tweets had to be encoded as unknown characters were used
Encoding(dfcube$text) <- "latin1"
Encoding(dfuser$text) <- "latin1"
Encoding(dfbethereum$text) <- "latin1"

#show all the tweets for all 3 users
dfcube$text
```

```
##  [1] "Our blogpost explains why the female travelers are https://t.co/xU3ZRIo8hp's key to winning th
```

```
##  [2] "#NewBusiness opportunity; @hotelsdotcom recently launched a full review. Reserve your spot to
##  [3] "Don't miss this week's post on why @PapaJohns should pay more attention to the movie-goers/gam
##  [4] "#NewBusiness opportunity with @PapaJohns's! Get the insights now to win the place as their new
##  [5] "Check out this week's blogpost to find out how e-commerce giant @amazon can better target the
##  [6] "Don't miss out! Click to get the insights you need to win the pitch for Amazon's global media
##  [7] "Find out how digital artists may be the perfect influencers for @lenovo to reach the millennia
##  [8] "Check out this week's #NBPA for some neat insights on how you can help @lenovo achieve a more
##  [9] "Having a recognizable logo for your brand is super important! https://t.co/6aIJoW28Uo"
## [10] "Great Philosophy! https://t.co/62jquBYdTN"
## [11] "Check out how @SUBWAY should be appealing to their target market: the Female Millennials https
## [12] "@doner_agency @socialmediaweek @BuzzFeed @FOXSports @DollarShaveClub Sounds like it'll be a gr
## [13] "@InitiativeWW Congrats!"
## [14] "@nomadicagency congrats! looks awesome :)"
## [15] "Check out this week's #NPBA for a #NewBusiness opportunity with @SUBWAY! Find out how to help
## [16] "Click to learn why Gen Z is the new face of Snackers everywhere and how you can help @DietzandW
## [17] "Powerful. https://t.co/rwwSWdRVYW"
## [18] "@susanp_bvk @RoyalCaribbean just beautiful."
## [19] "@thinkargus amazing to watch!"
```

```
dfuser$text
```

```
##  [1] "An interview with a #User #Experience #Guru https://t.co/gWdsbQvhzd #Futurist #IoT #BlockChain
##  [2] "#Dependant on #dumb #data and is making #bad #choices? #Douglas #Adams https://t.co/dtdCOIiuOX
##  [3] "#Information Architecture (IA) the #classification of #information https://t.co/niQbqhRR5r #Fu
##  [4] "#Usable #security in investment banking and #wealth #management https://t.co/NhwzntBZDv #Futur
##  [5] "#Social #Media Key Opinion Leaders #KOL #Huawei #MWC https://t.co/GMpuWHRBAu #Futurist #IoT #B
##  [6] "Paradigm Interactions R&amp;amp;D company UbiNET issues two cryptocurrencies the #ThingCoin and #U
##  [7] "#Information #Architecture (IA) the #classification of #information Part 2 https://t.co/vWNkQn
##  [8] "#Minimum #Viable #Experience #MVE because the viability in an #MVP is not #Customer #Centeredâ
##  [9] "#Open #Networking #Ecosystem #Protocol #Patent https://t.co/VUpW9TYxlo #Futurist #IoT #BlockCha
## [10] "#SmartLiving or as we might live with #artificial #intelligence and an open #IoT in a #new #rea
## [11] "#ubinetus has released 200 #UbiNETcToken https://t.co/h1ZBG2bHVW #Futurist #IoT #BlockChain #A
## [12] "An interview with a #User #Experience #Guru https://t.co/gWdsbQvhzd #Futurist #IoT #BlockChain
## [13] "#Strategic and #lean #thinking in private investment and asset portfolio participants part 2â ¦
## [14] "Do #eCommerce #analytics prove #customer #affiliation? https://t.co/kULp6zxkvv #Futurist #IoT #
## [15] "#Digital #Transformation is #About #People https://t.co/zdPEmKNgIo #Futurist #IoT #BlockChain #
## [16] "#IoT People in #London https://t.co/gcjqouaRfj #Futurist #IoT #BlockChain #Agile #DevOps https
## [17] "UbiNET Platform from UbiNET Inc. with cryptocurrency and blockchain 2018 asset sale https://t.
## [18] "Interview with a #User #Experience #Guru part 2 https://t.co/gmupD3YdQc #Futurist #IoT #BlockC
## [19] "#Agile #User #stories is a #UX #method https://t.co/g4tuwmL1Ht #Futurist #IoT #BlockChain #Agi
## [20] "#Situational #awareness drives open #IoT #Ecosystems not #visual #interfaces https://t.co/I72F
```

```
dfbethereum$text
```

```
##  [1] "Our #communities are almost at 6-digit numbers! í ½í²¬\nJoin them: https://t.co/n5XxTqJJSL\n#C
##  [2] "Have you already registered on https://t.co/VupC9N9Dcm?\nOur public #presale is starting soon!
##  [3] "1 #ETH = 17 500 #BTHR tokens (without #bonus). í ¾í´©\nLearn more: https://t.co/EfoYf4bKw1\n#T
##  [4] "Got your #ETH ready? In our public #presale, we aren't limiting anyone! í ½íº \nRegister to get
##  [5] "@Marinewan It will be announced just before the presale :) stay tuned!!"
##  [6] "It's finally here - the #Bethereum public #presale will start in 3 days!\nRegister now to get
##  [7] "Betting has an inherent social aspect, yet most online betting is impersonal. \nWe want to cha
##  [8] "Have you wondered why the #BTHR token is a perfect fit for the #blockchain? \nRead about it he
##  [9] "@DrJamesJamieso1 Thanks for your support James ;)"
```

```
## [10] "@bitwitch08 No minimum in the public presale starting on 11th of April :)"
## [11] "@EEliwan Thanks for your help :)"
## [12] "As you can tell we're very proud of our ratings! í ¾í´©\nThese are the latest verdicts of the r
## [13] "#eSports are in our roadmap for the Q1 of 2019! í ¼í¾®\nThey are among the fastest growing and
## [14] "@junieeegasm 1 ETH = 17 500 BTHR + bonus :)"
## [15] "@EscapeThisHelll There are multiple ways of buying ETH, for instance you can also use a Bitcoi
## [16] "The public #presale is coming up in less than a week!\nWant to contribute and don't have an #E
## [17] "@Jean_Jaki @BEthereum Will be announced ;) we're already thinking of a few locations!"
## [18] "Another snapshot of our local #Blockchain meet-up.\nCan't wait to meet more of our global supp
## [19] "BREAKING NEWS:\n@HackedCom one of the most prestigious #ICO analysis sites just gave us their
```

# Question 4

## (a) Cluster the Songs

As a first step, we install all necessary packages in R. Subsequently we import the data set and remove the first column. This column is simply an index variable which is redundant as each row already has an index in R.

```
library(cluster)
library(class)
library(C50)
library(rpart)
library(rpart.plot)
```

```
#Loading Data & removing fist row (index variable)
library(readr)
spotify <- read_csv("spotify.csv", col_types = cols(X1 = col_skip()))
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
spotify <- spotify[complete.cases(spotify),]
```

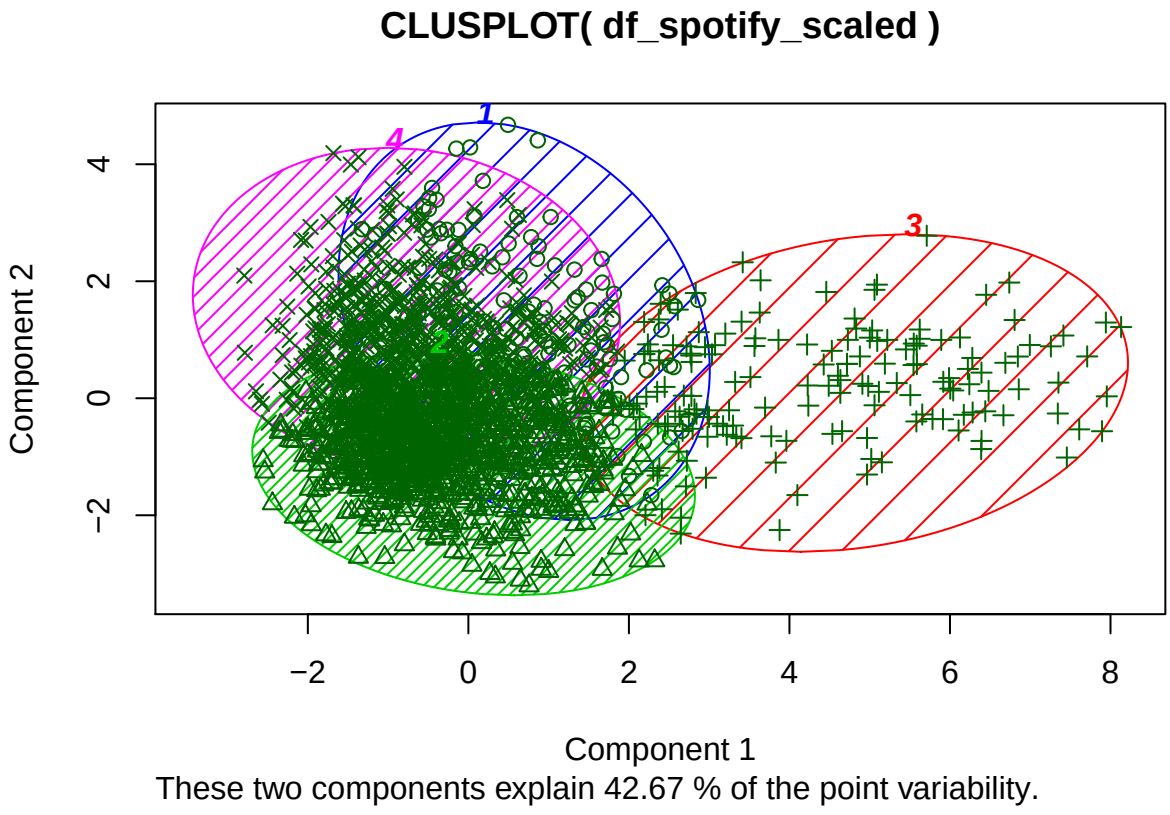
After loading the data, we inspect it and find that there are no missing values.

Since we are going to perform K means clustering, we are only going to use the numerical variables in this data set. To prevent skewed/imbalanced data to impact the model, these numerical variables have to be scaled. The variables *key, mode, timesignature* and *target* are integer variables but will not be used in this model as these are categorical variables according to Spotify (2018).

```
# Scaling all numerical variables
spotify_num <- spotify[,c(1:5 ,7,8,10,11,13)]
spotify_scaled <- scale(spotify_num)
df_spotify_scaled <- as.data.frame(spotify_scaled)
```

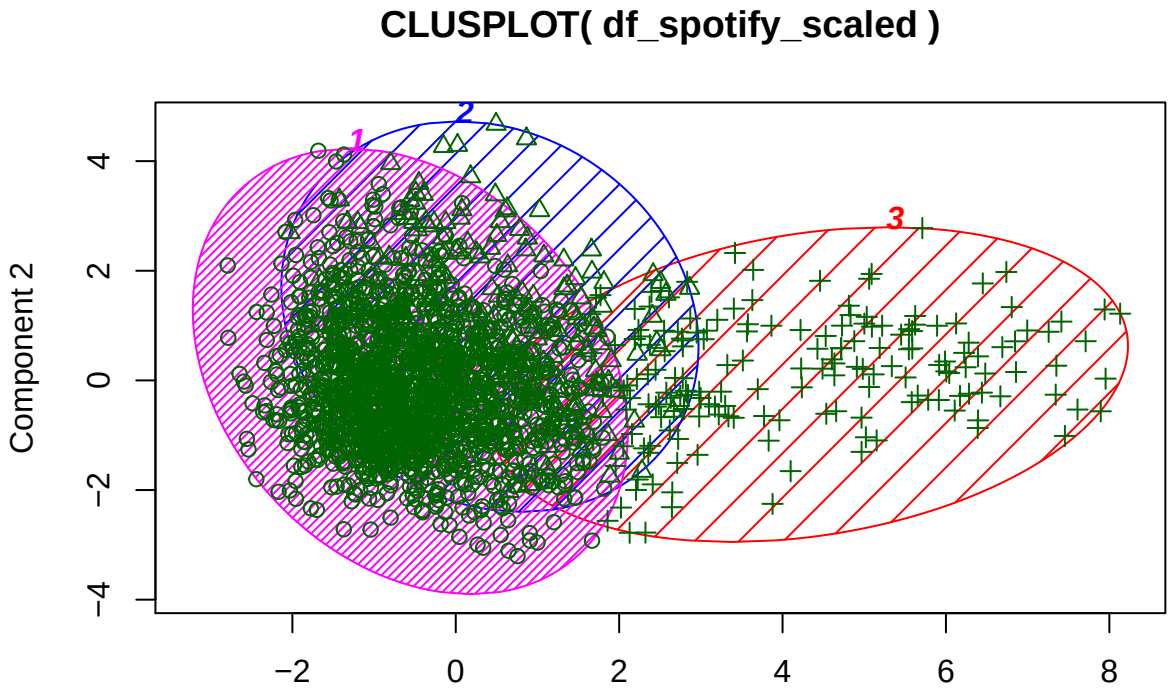
With K-means clustering, a number of clusters K has to be set arbitrarily. We are going to iteratively select K clusters and inspect the results using Principal Component Analysis. This method plots observations together on their scores on the two "most important" variables: the principal components. These are defined as the variables which explain the most variance in the data set. We initially start by selecting 4 clusters:

```r
# Find cluster solution, K = 4
rsltKmeans <- kmeans(df_spotify_scaled, 4)
# Cluster Plot against 1st 2 principal components
clusplot(df_spotify_scaled, rsltKmeans$cluster,
         color=TRUE, shade=TRUE,
         labels=4, lines=0)
```

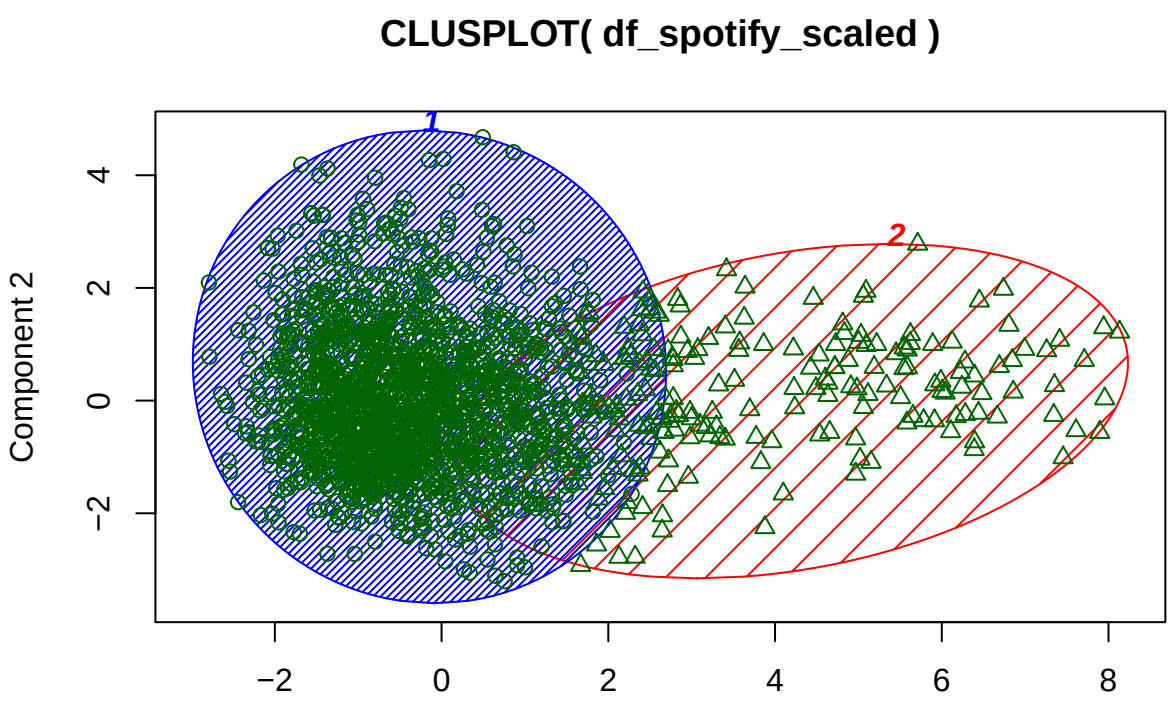



This first plot shows that the 2 selected components explain almost 42.67% of the point variability in the data set. The selected 4 clusters seems to fit the data not very well as there is significant overlap between the clusters. For this reason we retry with 3 and 2 clusters:

```r
# Find cluster solution, K = 3
rsltKmeans <- kmeans(df_spotify_scaled, 3)
# Cluster Plot against 1st 2 principal components
clusplot(df_spotify_scaled, rsltKmeans$cluster,
         color=TRUE, shade=TRUE,
         labels=4, lines=0)
```



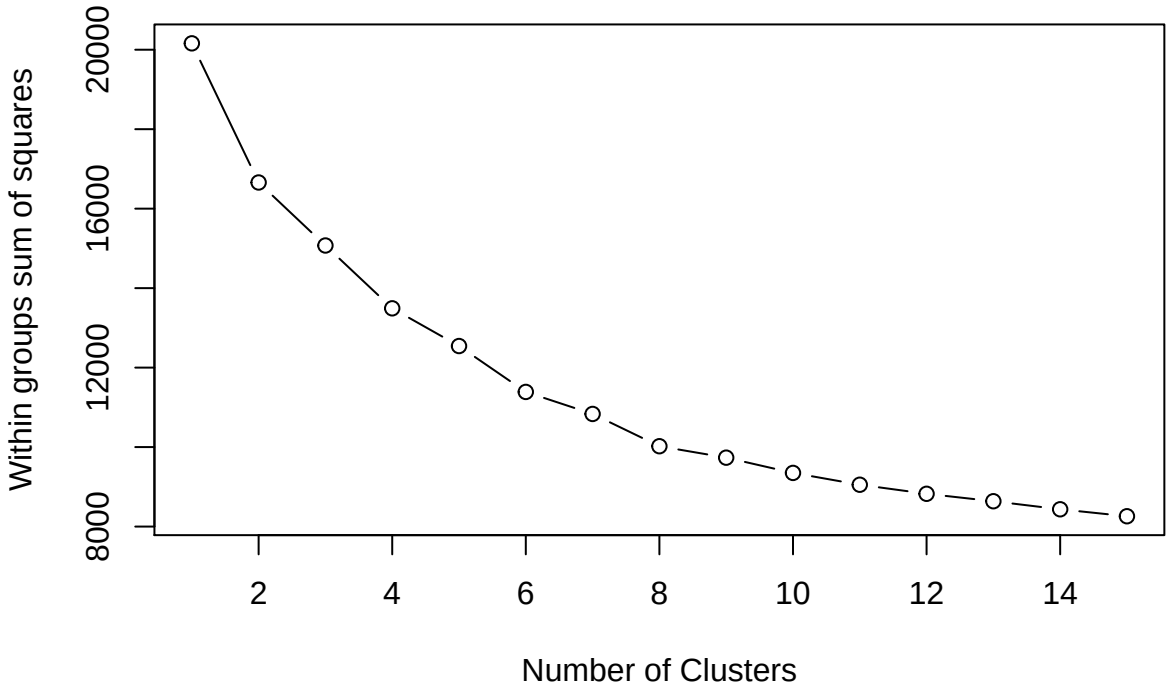

```
# Find cluster solution, K = 2
rsltKmeans <- kmeans(df_spotify_scaled, 2)
# Cluster Plot against 1st 2 principal components
clusplot(df_spotify_scaled, rsltKmeans$cluster,
         color=TRUE, shade=TRUE,
         labels=4, lines=0)
```




Due to the little overlap in the K = 2 graph, we believe that 2 clusters will fit this data set the best. Although using 3 clusters already improved the clarity of the plot compared to K = 4, there is still quite some overlapping sections present.

To strengthen our conclusion that we should use 2 clusters we also plot the within cluster variation (WSS) for a variety of clusters sizes (Elbow Method):

```
# The wss variables is initialised
wss <- (nrow(df_spotify_scaled)-1)*sum(apply(df_spotify_scaled,2,var))
# This for loop iterates the number of K clusters from 2 untill 15
for (i in 2:15) wss[i] <- sum(kmeans(df_spotify_scaled,
                                     centers=i)$withinss)
# Plotting the graph
plot(1:15, wss, type="b", xlab="Number of Clusters",
     ylab="Within groups sum of squares")
```

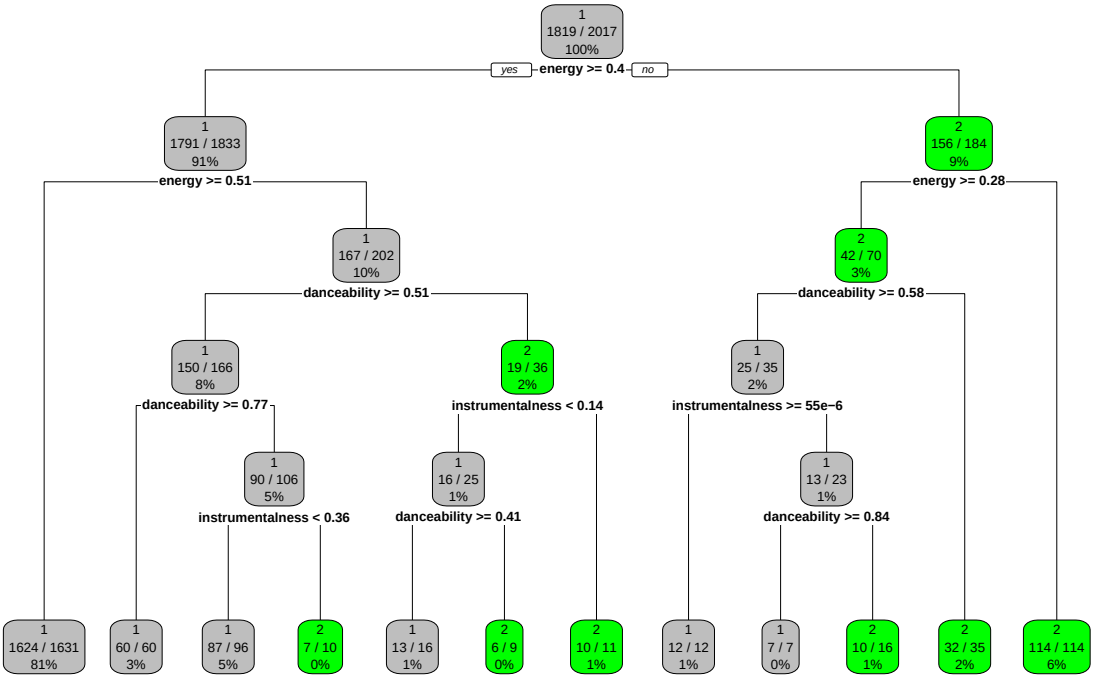



A steep decrease in the plot indicates the right number of clusters as extra clusters decrease a significant amount of the variation. The steepest decrease can clearly be found for 2 clusters. However, this method is often considered inaccurate, and thus only serves to reinforce our decision made above.

## (b) Explaining the Clusters.

To explain the different clusters we have build a decision tree model which predicts cluster membership. The final tree model has been developed after iterating various combinations of independent variables to the model. All variables which were not used in the decision tree (as they apparently did not add any information gain) have been excluded from the final model.

```
#Ensuring that clusters are set to 2.
rsltKmeans <- kmeans(df_spotify_scaled, 2)
# Add the cluster values to the original data set (the target variable)
spotify <- data.frame(spotify, cluster=as.factor(rsltKmeans$cluster))
# A decision tree model to explain Cluster memberships
mdlTree <- cluster ~ loudness + instrumentalness + danceability + energy + liveness + loudness
rsltTree <-rpart(mdlTree,
                 data = spotify,
                 method = "class",
                 parms=list(split="information"))
# Plot the decision tree
rpart.plot(rsltTree,
           box.col= c("grey","green","grey")[rsltTree$frame$yval],
           extra = 102)
```




To explain what songs are in which cluster, we follow the plotted decision tree top-down. It can instantly be noted that the energy variable provides significant information gain as it's used multiple times throughout the model. Songs in cluster 1 have high values for energy whereas songs in cluster 2 have significantly lower values. According to Spotify (2018), songs with higher values fore energy feel "fast, loud and noisy like death metal music while a Bach prelude scores low on the scale".

Music in cluster 2 is also found less 'danceable' according to the decision model. This can be expected as the danceability variable measures similar aspects as the energy variable such as tempo and beat strength. Thus songs in cluster 1 are more upbeat and danceable. Given the relative sizes of both clusters one could assume that songs from cluster 1 are the more mainstream, popular songs, whereas songs in cluster 2 might be songs of a more 'niche' genre like classical music.

Finally, the instrumentalness variable is used for multiple splits in the tree model but these are not easily interpreted as they are used in the lower areas of the model. To understand this variable, it is useful to know what the values represents. According to Spotify (2018), "the closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks". Not all splits of instrumentalness indicate the same direction (for two splits a lower value leads to cluster 1 whereas for 1 split a lower value leads to cluster 2. For this reason we argue that it is not directly possible to characterize clusters 1 and 2 based on this variable and we stick to using energy and danceability.

# References

Spotify, (2018) Get Audio Features for a Track https://developer.spotify.com/web-api/get-audio-features/