

**Institut Supérieur des Sciences Appliquées et Économiques**

**associé au Conservatoire National des Arts et Métiers**

**Entreposage et Fouille de données-STA 211**

**Devoir 1 :**

**Prétraitement des données**

**Prépare par : ……ELIO…BOU…SERHAL……….**

**Date :……LE…MARDI…14…FEVRIER…2023……………….**

# **Table de matières**

[**Liste des figures 2**](#_Toc127099125)

[**Objective 3**](#_Toc127099126)

[**Résumé ou partie théorie 3**](#_Toc127099127)

[**Déscription de la base de données 4**](#_Toc127099128)

[**Représentation graphiques de la base de données : 10**](#_Toc127099129)

[**Application la méthode sur la base de données 10**](#_Toc127099130)

[**Conclusion 10**](#_Toc127099131)

[**Références 10**](#_Toc127099132)

[**Annexes 10**](#_Toc127099133)

# **Liste de Tableau**

[**Tableau 1: Tableau des sentiments 18**](#_Toc127124107)

[**Tableau 2: Combinaison entre les sentiments et les phrases du texte 18**](#_Toc127124108)

[**Tableau 3: Les mots indiqués pour l'analyse des sentiments 19**](#_Toc127124109)

[**Tableau 4: Les resultats de l'analyse des sentiments 19**](#_Toc127124110)

[**Tableau 5: Rapport Final sur l'analyse des sentiments 20**](#_Toc127124111)

# **Liste de Figure**

[**Figure 2: Read Arabic text 6**](#_Toc127124146)

[**Figure 3: Cleaning Arabic Text 7**](#_Toc127124147)

[**Figure 4: Tranforming Arabic Text into Data Frame 8**](#_Toc127124148)

[**Figure 5: Translate Arabic into French-Arabic 9**](#_Toc127124149)

[**Figure 6: Data Frame structure 9**](#_Toc127124150)

[**Figure 7: Corpus Text 10**](#_Toc127124151)

[**Figure 8: Term-Document Matrix 11**](#_Toc127124152)

[**Figure 9: Sum of repeated term 11**](#_Toc127124153)

[**Figure 10: Frequency of each term 11**](#_Toc127124154)

[**Figure 11: Top 5 most frequent words 12**](#_Toc127124155)

[**Figure 12 Top 5 most frequent words 12**](#_Toc127124156)

[**Figure 14: mots avec une fréquence de 15 13**](#_Toc127124157)

[**Figure 13: mots avec une fréquence de 10 13**](file:///C:\Users\LF%20ELEC\Desktop\CNAM\STA-%20211\Projet%20STA211\projet_text_mining_arabic.docx#_Toc127124158)

[**Figure 15: mots avec une fréquence de 5 Figure 16: mots avec une frequence de 2 14**](#_Toc127124159)

[**Figure 17: Association(1) entre les terms selon la fréquence 15**](#_Toc127124160)

[**Figure 18: Association(2) entre les terms selon la fréquence 15**](#_Toc127124161)

[**Figure 19: Association(3) entre les terms selon la fréquence 16**](#_Toc127124162)

[**Figure 20: Association(4) entre les terms selon la fréquence 16**](#_Toc127124163)

[**Figure 21: Word Cloud Representation 17**](#_Toc127124164)

# **Objective**

Le **Process Mining** permet de produire des rapports flexibles et d'exprimer les analyses de **processus** sous la forme de visualisations et de tableaux de bord faciles à partager, à lire et à personnaliser, quel que soit le nombre de questions posées par votre supérieur

# **Résumé ou partie théorie**

1. ***Text mining : définition***

Le **texte mining**, ou fouille de textes, respecte deux étapes principales. La première étape, l'analyse, consiste à analyser les corpus de textes de manière à en reconnaître les mots, les phrases, les rôles grammaticaux ainsi que les relations et les sens de ces derniers entre eux. Le **Text Mining, ou analyse de texte**, consiste à transformer un texte non structuré en données structurées pour ensuite procéder à l’analyse.

1. ***Quelle est le lien entre machine Learning et Text mining ?***

**Le Data Mining est une ressource sur laquelle le Machine Learning peut s'appuyer pour accomplir ses fonctions**. Ils participent ensemble à améliorer la performance de l'entreprise et sa compétitivité

1. ***Qui utilise le data mining ?***

Les logiciels de data mining sont principalement utilisés par **les entreprises focalisées sur le comportement des consommateurs, à savoir dans les secteurs du retail, de la communication, du marketing et de la finance**.

1. ***Quelles sont les 2 catégories de data mining ?***

Types de data mining  
Le data mining comporte deux processus principaux : **l'apprentissage supervisé et non supervisé**.

1. ***Les avantages du Text Mining***

Le Text Mining présente de nombreux avantages, à l’heure où les entreprises et les individus génèrent **chaque jour d’immenses volumes** de données. En effet, près de 80% des données de texte sont non structurées. Il est donc impossible de les analyser sans recourir au Text Mining.

En guise d’exemple, on peut citer les emails,**les publications sur les réseaux sociaux**, les discussions sur les messageries, les requêtes auprès du service client, les sondages… il est très difficile de trier manuellement ces informations.

L’analyse de texte permet d’analyser de larges volumes de données en quelques secondes seulement, et donc **de gagner en productivité**. Ces analyses peuvent être effectuées en temps réel, et il est donc possible d’intervenir immédiatement en cas de détection de problème

# **Déscription de la base de données**

1. **Identification du texte à explorer.**:

Préparation du texte avant exploration. Si le texte apparaît dans plusieurs fichiers, enregistrez-les tous au même endroit. Dans le cas de bases de données, déterminez le champ contenant le texte.

Ici on va traiter un texte tiré des paroles de Papp-François parlons a propos de l’amour, du traitement , de dieux comment nous a aimé…

On va utiliser plusieurs librairies et plusieurs méthodes pour voir quelle sont les mots les plus répétées ? Et quelle est le but de cette parole ? Et sur quoi insiste-t-il ?et quels sont les sentiments des mots utilises ?

ETAPE 1 : Transformer le system locale de Rstudio en « Arabe »

ETAPE 2 : Utiliser les librairies suivantes : « "tm", "arabicStemR" »

ETAPE 2 : Lire le texte en utilisant la fonction « readLines »

On peut voir les 17premieres lignes du texte :

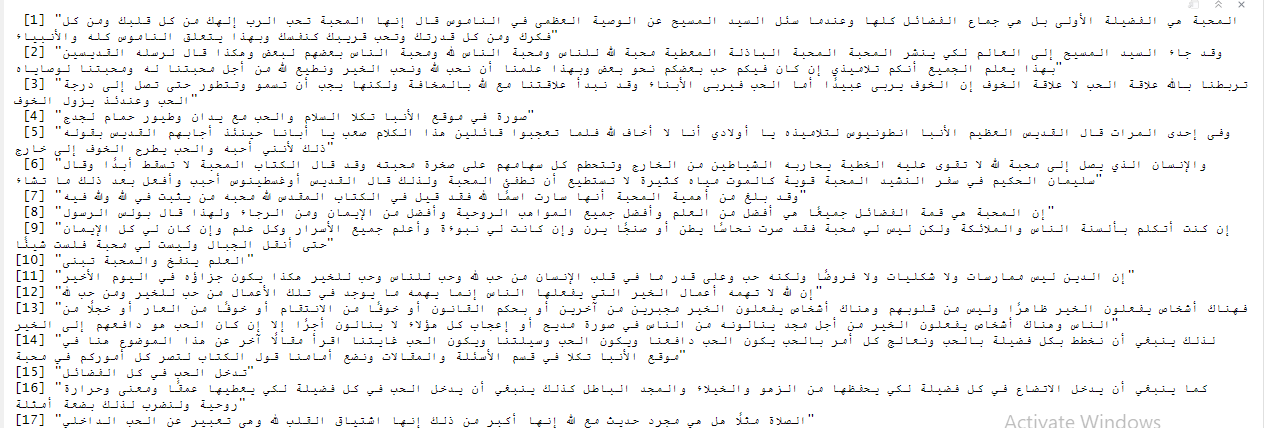


Figure 2: Read Arabic text

1. **Nettoyage de la base de donnée :**

Avant de commencer le travail il faut nettoyer notre base de donne, comment ?

en suivant les étapes suivantes :

ETAPE 1 : Eliminer les ponctuations en utilisant la fonction « removePunctuation »

ETAPE 2 : Eliminer les nombres en utilisant la fonction « removeNumbers »

ETAPE 3 : Eliminer les symboles en utilisant la fonction « removeNewlineChars »

ETAPE 4 : Eliminer les espaces vides en utilisant la fonction « stripWhitespace »

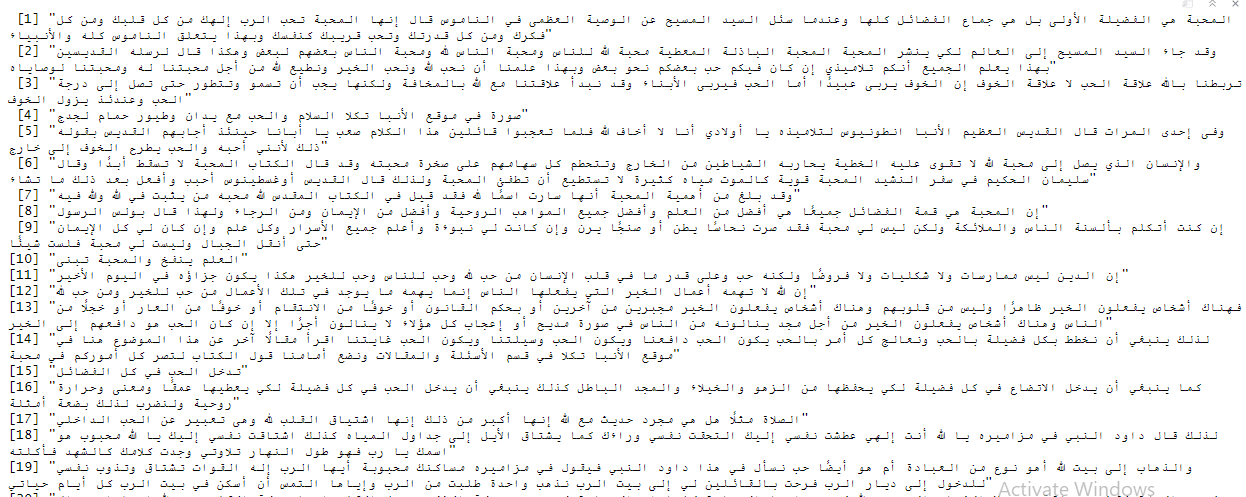


Figure 3: Cleaning Arabic Text

1. **Structure de la base de donnee :**

Apres le nettoyage de la base de donnee , maintenant notre texte est prêt à se traiter, mais avant de commencer le traitement il faut transformer : **texte en data frame** .

Dans notre Data Frame on a 2 colonne :

une qui nous indique l’identité du document (doc\_id) c’est–a-dire cette colonne numérote chaque ligne de ce texte, et l’autre (Arabic\_text) contienne chaque phrase en « arabe »

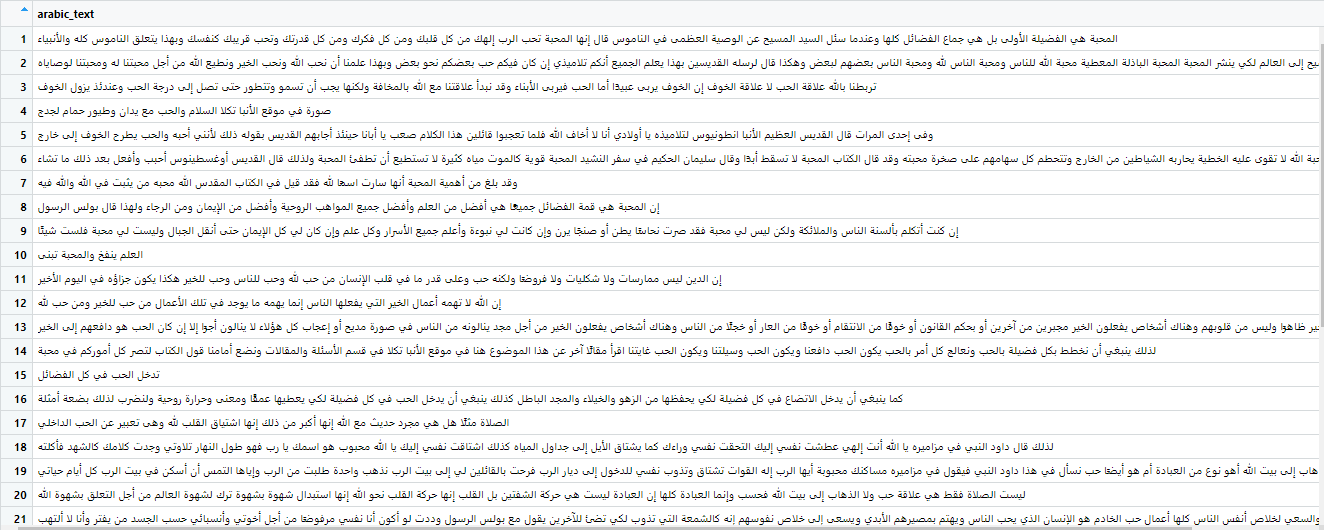


Figure 4: Tranforming Arabic Text into Data Frame

1. **Transformation : (texte 🡪 data frame)**

Apres le classement de notre base de donne (texte) en phrase dans le data frame, on doit maintenant transformer chaque phrase comment ?

ETAPE 1 : Créer une nouvelle colonne sur notre data frame « arabic\_text »

ETAPE 2 : Traduire chaque ligne en Anglais-Arabe en utilisant la fonction « Transliterate »

ETAPE 3 : View data pour visualiser la traduction de chaque ligne de ce text traite

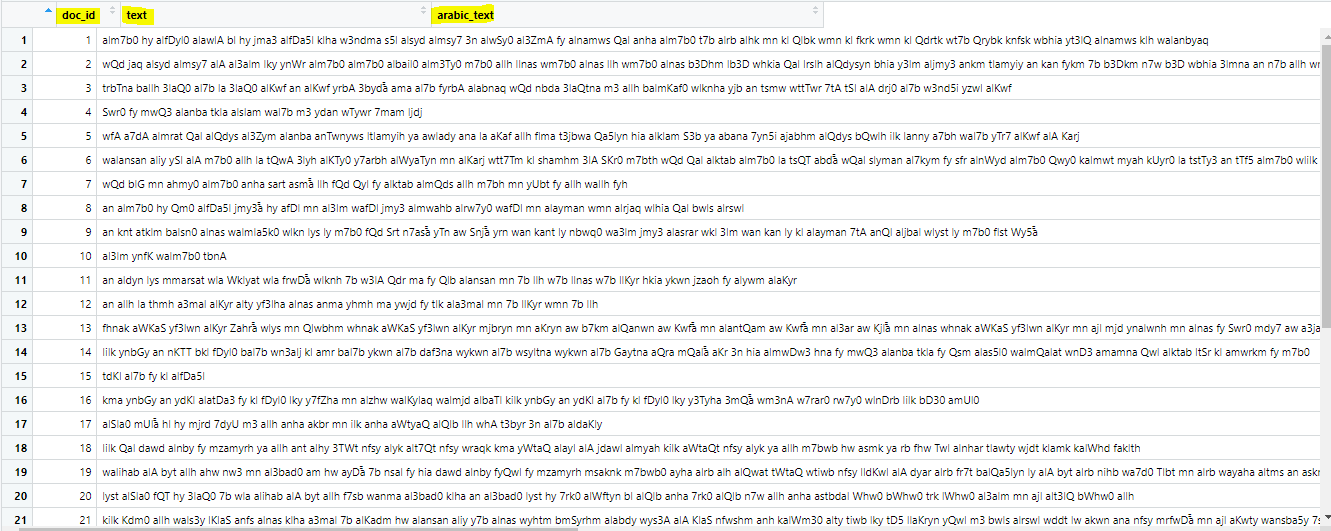
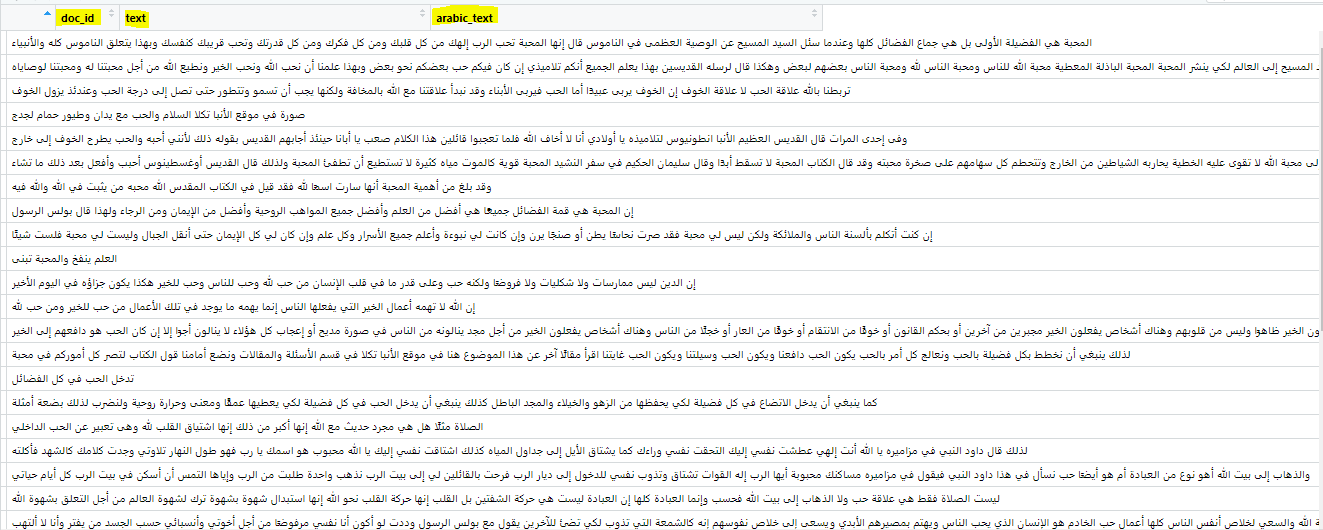


Figure 5: Translate Arabic into French-Arabic

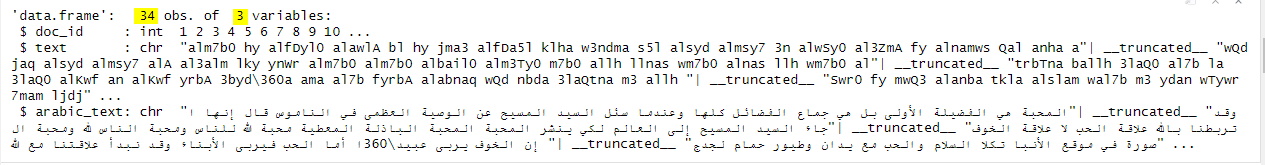


Figure 6: Data Frame structure

1. **Text Corpus :**

Notre base de données est stockée dans une Data Frame après transformation et réduction par élimination, alors c’est le temps de former un corpus

Définition d’un Corpus : c’est **un rassemblement de textes ou une collection de textes regroupés sur la base d'hypothèses de travail en vue de les interroger.** Autrement dit c’est le texte obtenu après la transformation et la réduction des mots.

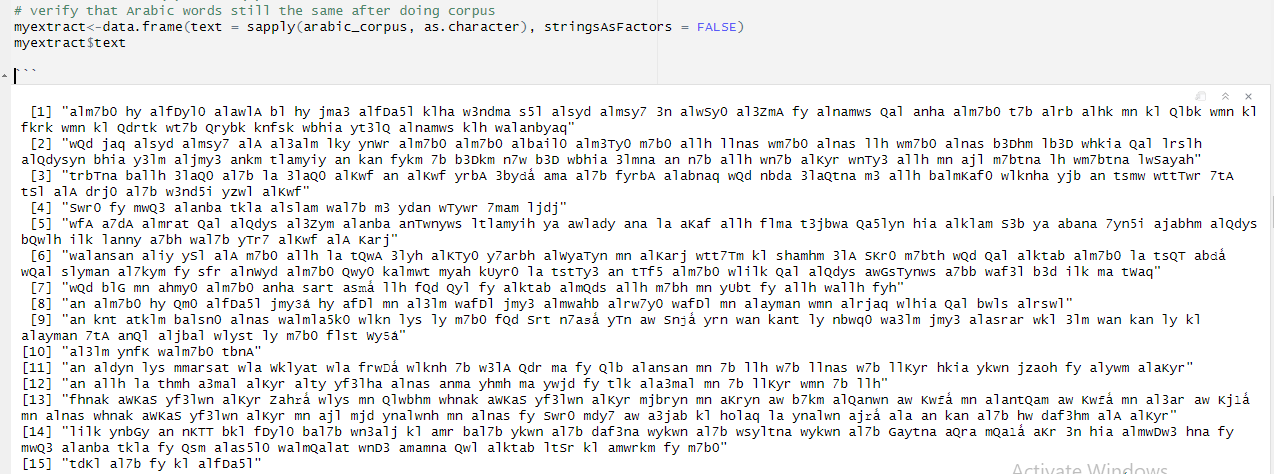


Figure 7: Corpus Text

1. **Transformation : (texte corpus 🡪 Data matrix)**

En transformant de nouveau la data frame en data matrix dans le but de :

ETAPE 1 :voir la dimension de notre base de donnée (ligne et colonne)

ETAPE 2 :Voir de combien de fois chaque mots est répété en utilisant la fct « sum »

ETAPE 3 : Voir la fréquence de chaque mot

Tout d’abord on transforme le text en une matrice dans e but de voir combient de fois chaque mot est répété dans chaque ligne ?

Chaque ligne du matrice : convient a un mot du texte

Chaque colonne du matrice : convient a une phrase du text

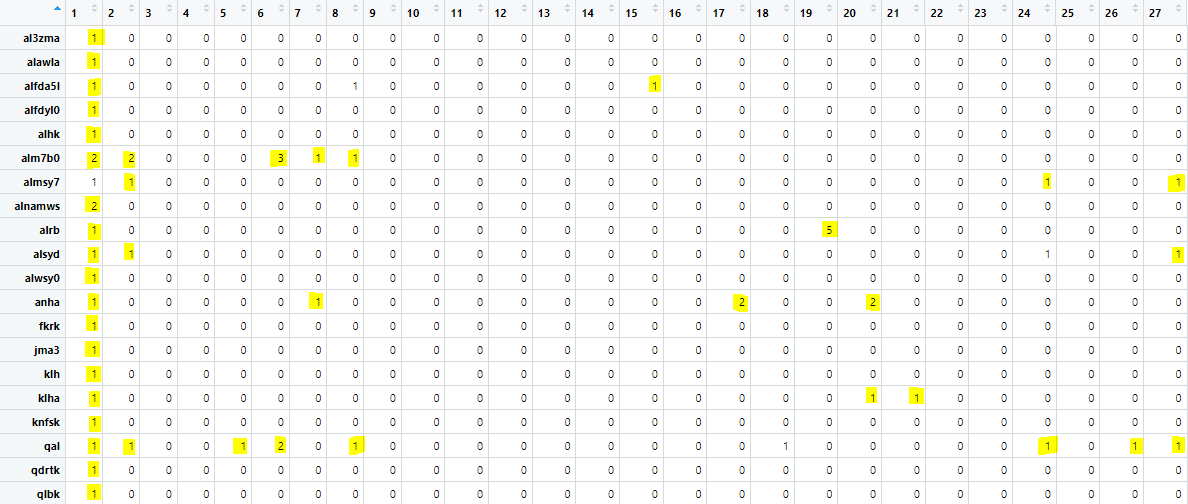


Figure 8: Term-Document Matrix

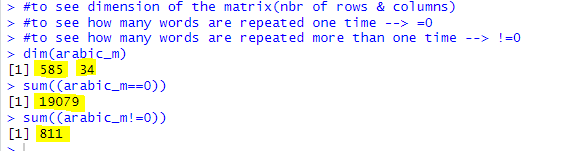


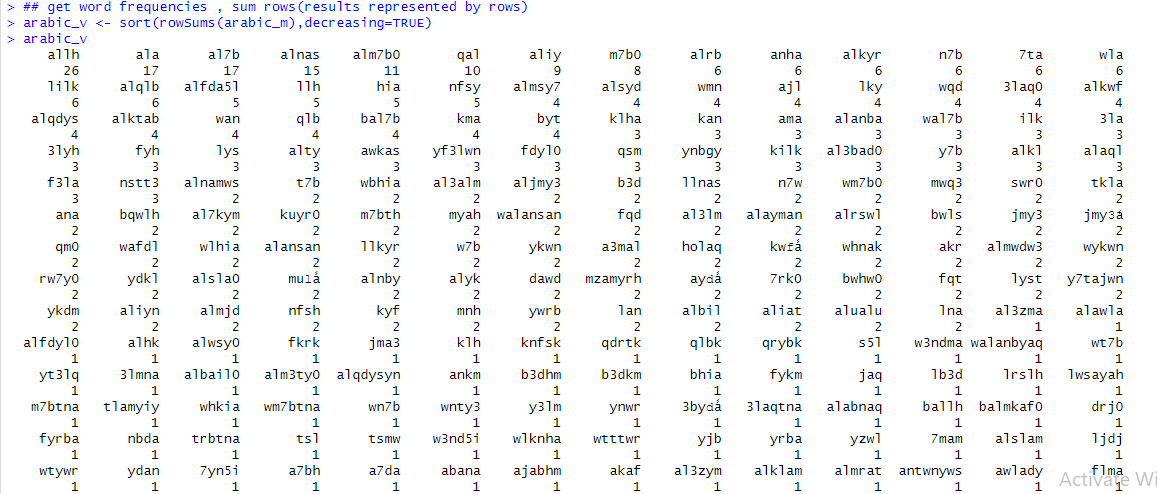
Figure 9: Sum of repeated term

Figure 10: Frequency of each term

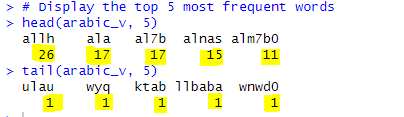


Figure 11: Top 5 most frequent words

# **Représentation graphiques de la base de données :**

1. **Visualisation :**

Comme nous avons dit, il existe beaucoup de libraires et des méthodes pour visualiser les mots répétés selon leurs fréquences :

METHODE 1 : En utilisant le « Bar plot »

On peut voir que le mot « allh » a une frequence de 27 , et les deux mots « ala » et « al7b » ont une frequences de 17, le mot « alnas » a une frequence 15 et le mot « alm7bo » de frequence 11

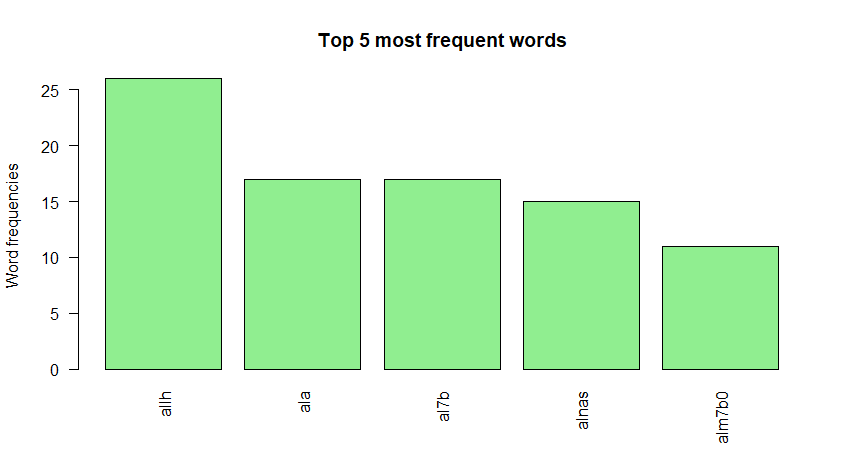


Figure 12 Top 5 most frequent words

METHODE 2 : En utilisant le «WORD CLOUD (1) »

Le principe du **nuage de mots** est basé sur une méthode d’analyse de textes qui nous permet de mettre en évidence les mots-clés les plus fréquemment utilisés dans un paragraphe de textes. Le nuage de mots est également appelé **word cloud** ou **tag cloud** en anglais. La procédure de création d’un nuage de mots est très simple avec le **logiciel R** si vous connaissez les différentes étapes à exécuter. Le package **tm** (pour text mining) et le package **wordcloud** (pour générer le nuage de mots clés) sont disponibles dans R pour nous aider à analyser des textes et de visualiser rapidement les mots-clés en nuage de mots.

# 3 raisons pour lesquelles vous devriez utiliser des nuages de mots pour présenter vos textes

1. Le **nuage de mots** est une méthode puissante pour l’analyse de textes. Il ajoute de la simplicité et de la clarté. Les mots-clés les plus utilisés ressortent mieux dans un nuage de mots.
2. Le nuage de mots est un outil de communication puissant. Il est facile à comprendre, à partager et est percutant
3. Le nuage de mots est visuellement plus agréable qu’une table de données remplie de textes

# Qui utilise les nuages de mots?

* Les chercheurs: pour la présentation des données qualitatives
* Les Marketers: pour mettre en évidence les besoins et les points d’insatisfaction des clients
* Les enseignants: pour soutenir des sujets essentiels
* Les politiciens et les journalistes
* Les réseaux sociaux: pour collecter, analyser et partager les sentiments des utilisateurs

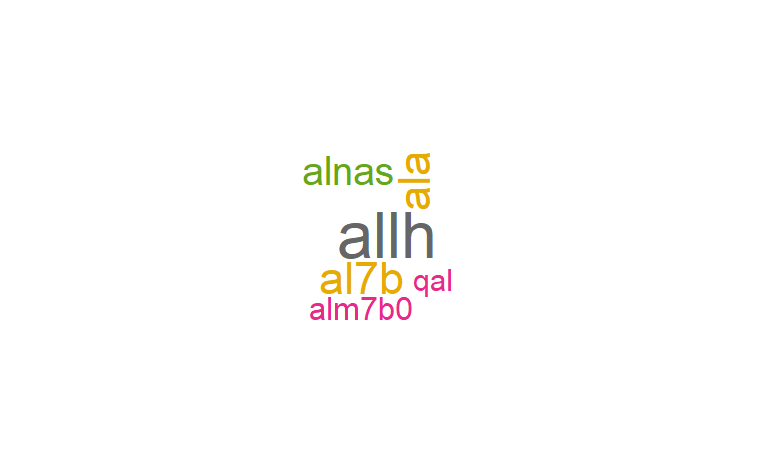
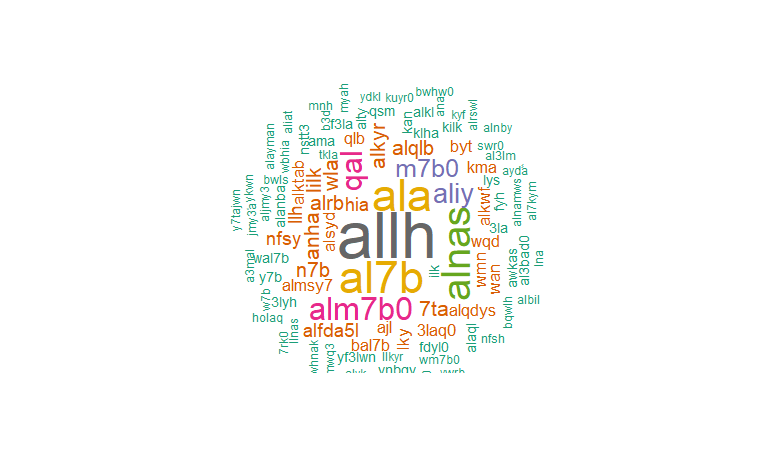


Figure 14: mots avec une fréquence de 15

Figure 13: mots avec une fréquence de 10



Figure 15: mots avec une fréquence de 5 Figure 16: mots avec une frequence de 2

METHODE 3 : En utilisant l’association entre les terms « findFreqTerm »

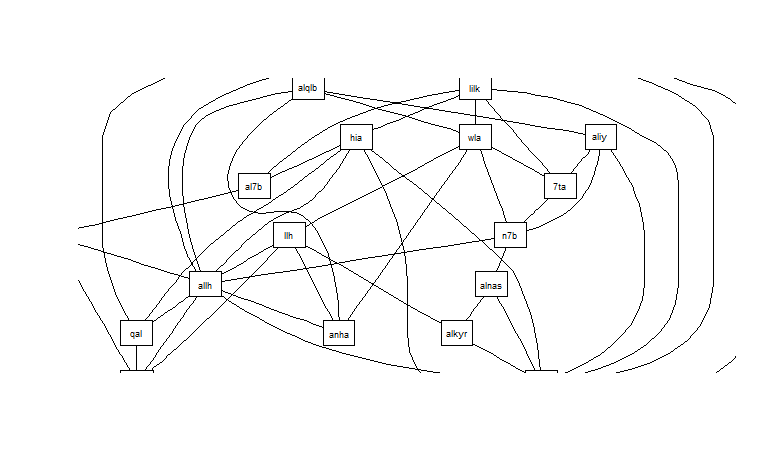


Figure 17: Association(1) entre les terms selon la fréquence

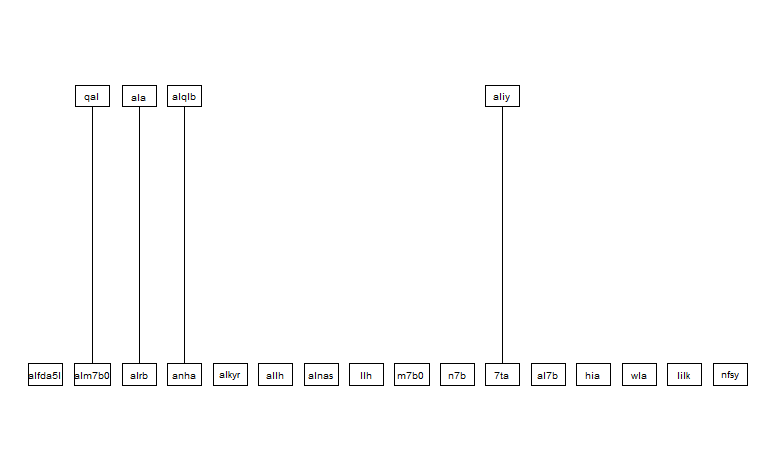


Figure 18: Association(2) entre les terms selon la fréquence

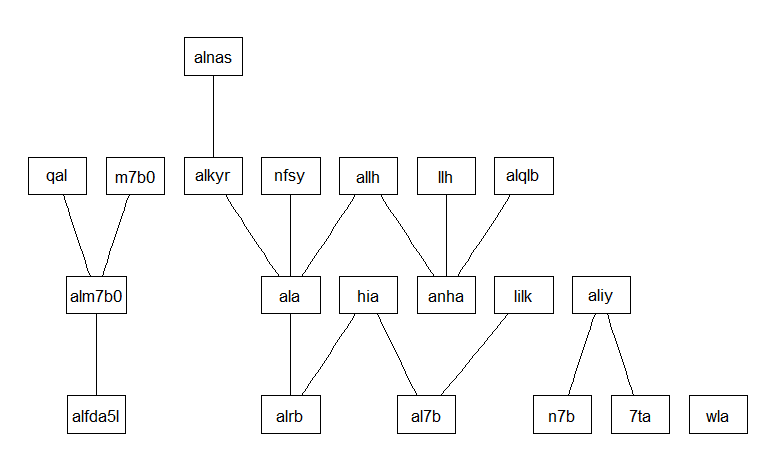


Figure 19: Association(3) entre les terms selon la fréquence

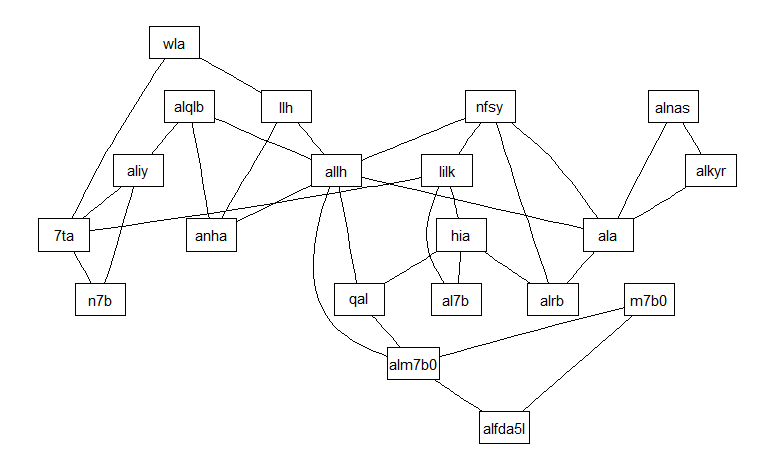


Figure 20: Association(4) entre les terms selon la fréquence

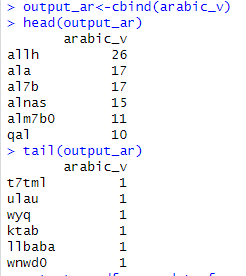
Dans la méthode 2 et 4 en regardant les figures on peut déduire que le mots le plus grand et gras correspond a une fréquence plus grande, alorsque dans la methode 3 on peut deduire l’association entre les termes selon leurs frequences.par exemple ici le mot « allh » est relie aux mots suivants : « nafsy, llah , anha , qal , alm7bo , ala , alqlb …»

METHODE 4 : En utilisant le «WORD CLOUD (2) »



Figure 21: Word Cloud Representation

On peut aussi voir ces mots dans une data frame qui est stockée comme fichier contenant tous les mots répétés et non répétés



1. **Stockage des résultats sous forme d’un texte**

On peut créer un fichier contenant les mots répétés dans un ordre croissant ou décroissant. Aussi on peut les transformer de l’Arabe en French-Arabie et les stockes aussi comme fichier en utilisant « reverse.transliterate ».

ETAPE 1 : Stocker les mots les plus fréquents dans le texte

ETAPE 2 : Stocker les mots traduits (Arabie 🡪 French-Arabie)

ETAPE 3 : stocker tous les termes de ce texte sous forme d’un tableau

# **Application de la méthode sur la base de données**

1. **L’analyse des sentiments du texte :**

L’analyse des sentiments d’un texte en arabe ce fait dans une manière différentes qu’un texte en français ou Anglais. Comment ?

ETAPE 1 : Créer un fichier contenant les sentiments choisit

Ici on a créé aléatoirement un fichier contenant 34 lignes parce que notre texte est forme de 34 phrases. C’est a seule methode utiliser pour faire l’analyse des sentiment d’un texte en Arabe.

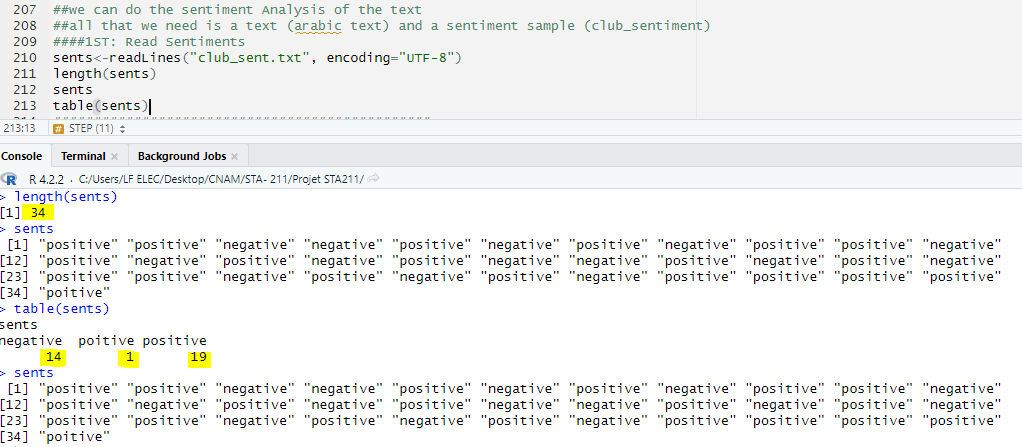


Tableau 1: Tableau des sentiments

ETAPE 2 : Traduire les sentiments en utilisant « Transliterate ou reverse.transliterate »

On ajoute une nouvelle colonne contenant les sentiments pour obtenir 3 différents colonnes « arabic\_text , sents , arabic\_text\_t ».



Tableau 2: Combinaison entre les sentiments et les phrases du texte

ETAPE 3 : Faire la partition de notre donnée (training set & testing set)

ETAPE 4 : Choisir les termes sur-lesquels on veut faire l’analyse des sentiments

ETAPE 5 : En utilisant la librairie « RTextTools » on trouve une prédiction des sentiments

Mais ce qu’il faut savoir que l’analyse des sentiments d’un texte Arabe est différent des autre textes (Anglais ou Français). Alors ici on doit préciser les mots qu’on veut les traités (on a pris un petit exemple contenant 13 mots sur lesquels on faire l’analyse des sentiments), en introduisant ces mots dans un vecteur. Ensuite on fait une combinaison entre les mots indiques et les sentiments pour obtenir les résultats suivant :

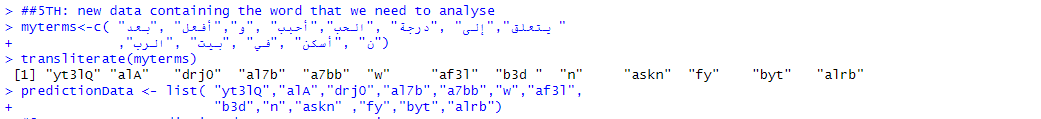


Tableau 3: Les mots indiqués pour l'analyse des sentiments

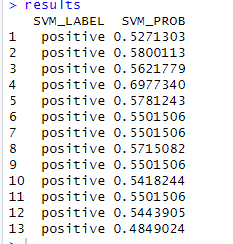


Tableau 4: Les resultats de l'analyse des sentiments

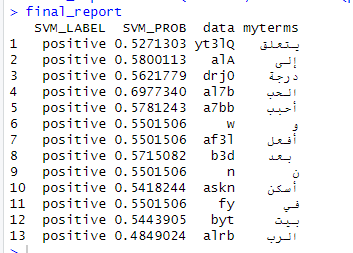


Tableau 5: Rapport Final sur l'analyse des sentiments

# **Remarque**

On peut traiter notre base de données en deux façons, mais dans les deux façons on obtiendra les mêmes résultats

TRAITEMENT 1 : Créer un « **texte corpus** » et appliquer (Etape 1 🡪 Etape 9)

TRAITEMENT 2 : Traduire le texte tout suit sans créer un texte corpus

# **Conclusion**

On utilisant Rstudio ona pu eliminer les mots qui ne sont pas importants comme les connecteurs , les guillemets, les nombres …. On constate que les mots les plus répétés forment un champ lexical sur l’amour « alm7bo » alors ce texte est en train de nous expliquer l’importance de l’amour dans la vie.

Selon l’auteur ici l’amour c’est une relation entre : Dieux « allah » et Homme/Femme « alnas »

# **Références**

1. Arabic Text Mining Using R Transliterate and Reverse Transliterate from and to Arabic R script used: [https://app.box.com/s/wv4zxash2c4l8jt...](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbm1xY1d0TWk4bjI4YUVvRUt4cUhtZnQ2aWpaZ3xBQ3Jtc0trWWRzbnBQUUtOTjM5Z0xUakR1QVlIaVU0NG9WWHZ2UUhqdFJneHRxdEdiakxsc0FXdDYwOXV0T3IyT19wT0xlNTZWUV9jVllVNWdHdXhoemJveEwxZVg5RkwyZHdZTHdMcDJMLUJKUHpJc2RGMmlvMA&q=https%3A%2F%2Fapp.box.com%2Fs%2Fwv4zxash2c4l8jt1wjljsd6hauf5ahkw&v=4ZoEh5WFqo4)
2. Arabic Text Mining Using R 2nd Edition corrected the error reported from Arabic Text Mining Using R [https://www.youtube.com/watch?v=4ZoEh...](https://www.youtube.com/watch?v=4ZoEh5WFqo4&t=0s) R script used [https://app.box.com/s/5epfffrddga16t1...](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbXdJcnZrZnZJbnluRFNoQklscDlrRzhHMzYyUXxBQ3Jtc0ttUHlCc2UtRkh4cXZ5UUJVeDljeXdvMnlDY0JUekdsVzVXejl2ZE9DM2Uwd0dld3hxcmlCZVdYeFZXZlA1bDZNTVBmU0Z4U00yLUJFSUZSMzNrS3NWQm9YaWcwbWVYYXRQQ0d6TjY1YmhQR1RtQnJzQQ&q=https%3A%2F%2Fapp.box.com%2Fs%2F5epfffrddga16t1g1xkhaibjtsq0bz97&v=RPo8AGYp0Ok)

# **Annexes**

---

title: "Text-Mining"

author: "ELIO"

date: "2023-02-06"

output:

html\_document: default

word\_document: default

---

To work on Rstudio text-mining we should respect the choosen laguage

Each language has it owns packages , libraries and methods

Here we will see how to deal with Arabic text:

```{r}

############# STEP (1) #############

## The following should work on Windows - first grab and save your existing locale

print(Sys.getlocale(category = "LC\_CTYPE"))

original\_ctype <- Sys.getlocale(category = "LC\_CTYPE")

## Switch to the appropriate local for the script

Sys.setlocale("LC\_CTYPE","arabic")

```

```{r}

############## STEP (2) ##############

##load required packages

library("tm")

library("arabicStemR")

library("wordcloud2")

library("NLP")

library("SnowballC")

library("wordcloud")

library("RColorBrewer")

library("syuzhet")

library("ggplot2")

```

```{r}

############ STEP (3) ###########

# read text with UTF-8 encoding

# to load the data into a corpus.

# Read the text file from local machine, choose file interactively

#arabic\_text <- readLines(file.choose())

arabic\_text<-readLines("arabic.txt", encoding="UTF-8")

# have a look !

arabic\_text

arabic\_text = arabic\_text[arabic\_text!=""]

arabic\_text

```

```{r}

######### STEP (4) #########

## start cleaning

arabic\_text<-removePunctuation(arabic\_text)

arabic\_text<-removeNumbers(arabic\_text)

arabic\_text<-removeNewlineChars(arabic\_text)

arabic\_text<-stripWhitespace(arabic\_text)

# Have a look!

arabic\_text

```

```{r}

######### STEP (5) : PART (A)#########

#transform the text into data frame

#contains 2 columns (doc\_id, Arabic\_text)

##1ST CHANGING STRUCTURE DATA TYPE (from text to dataframe)

##2ND VIEW THE DATA

myartxt = data.frame(arabic\_text, stringsAsFactors = F)

View(myartxt)

######### STEP (5) : PART (B)#########

#transform arabic text into french-arabic(par substitution)

#contains 3 columns (doc\_id, text, Arabic\_text)

##1ST WE ADD A NEW COLUMN

##2ND WE REPLACE ARABIC BY FRENCH-ARABIC(using "transliterate" function)

##3RD VIEW THE DATA

myartxt$doc\_id = c(1:nrow(myartxt))

myartxt$text = myartxt$arabic\_text

myartxt = myartxt[,c(2,3,1)]

myartxt$text = transliterate(myartxt$text)

View(myartxt)

```

```{r}

############ STEP (6): PART (A)##########

# create Arabic text corpus

str(myartxt)

arabic\_corpus <- Corpus(DataframeSource(myartxt) )

############ STEP (6): PART (B)##########

# verify that Arabic words still the same after doing corpus

myextract<-data.frame(text = sapply(arabic\_corpus, as.character), stringsAsFactors = FALSE)

myextract$text

```

```{r}

############ STEP (7) : PART(A) ############

#transform our text into a matrix

# Build a term-document matrix

arabic\_tdm <- TermDocumentMatrix(arabic\_corpus)

# visualize matrix(we can see frequency of each term in each row)

arabic\_m <- as.matrix(arabic\_tdm)

View(arabic\_m )

############ STEP (7) : PART(B) ############

#to see dimension of the matrix(nbr of rows & columns)

#to see how many words are repeated one time --> =0

#to see how many words are repeated more than one time --> !=0

dim(arabic\_m)

sum((arabic\_m==0))

sum((arabic\_m!=0))

############ STEP (7) : PART(C) ############

## get word frequencies , sum rows(results represented by rows)

arabic\_v <- sort(rowSums(arabic\_m),decreasing=TRUE)

arabic\_v

## get word frequencies , sum columns(results represented by columns)

arabic\_d <- data.frame(word = names(arabic\_v),freq=arabic\_v)

arabic\_d

# Display the top 5 most frequent words

head(arabic\_v, 5)

tail(arabic\_v, 5)

```

```{r}

############ STEP (8) : PART(A):METHOD 1: "BAR PLOT" ###########

#Plot the most frequent words

barplot(arabic\_d[1:5,]$freq, las = 2, names.arg = arabic\_d[1:5,]$word,

col ="lightgreen", main ="Top 5 most frequent words",

ylab = "Word frequencies")

############ STEP (8) : PART(B):METHOD 2: "WORD CLOUD" ###########

#generate word cloud

set.seed(1234)

wordcloud(words = arabic\_d$word, freq = arabic\_d$freq, min.freq = 5,

max.words=100, random.order=FALSE, rot.per=0.40,

colors=brewer.pal(8, "Dark2"))

############ STEP (8) : PART(C):METHOD 3: "GRAPH-PLOT" ###########

#Loading required package: grid

library(graph)

library(Rgraphviz)

freq.terms <- findFreqTerms(arabic\_tdm, lowfreq = 5)

plot(arabic\_tdm, term = freq.terms, corThreshold = 0.1, weighting = F)

############ STEP (8) : PART(D):METHOD 4: "WORD CLOUD" ###########

#step5:visualize the most repeated

wordcloud2(data = arabic\_d)

```

```{r}

############# STEP (9) #################

#creating a text file (output\_ar.txt) contaning all the transformed words with freq

#it will be saved in our folder

#View the data

output\_ar<-cbind(arabic\_v)

head(output\_ar)

tail(output\_ar)

output\_ar\_df = as.data.frame(output\_ar)

output\_ar\_df$arabic = row.names(output\_ar\_df)

for (i in 1:nrow(output\_ar\_df)){output\_ar\_df$arabic\_trans[i] = reverse.transliterate(output\_ar\_df$arabic[i]) }

View(output\_ar)

```

```{r}

############## STEP (10) ##################

###### SAME RESULTS TO STEP (7) BUT HERE THE STRUCTER IS "TABLE" OR IN STEP (7) IT WAS "MATRIX" ##########

# write output in arabic

## Now you can write your text out and have it look as you would expect

#using fct "write.table" in order transform the df into "table data matrix"

#this is the matrix containing 0 & 1 only

write.table(output\_ar\_df, "arabic\_output.txt", quote = FALSE, col.names = FALSE,

row.names = T, sep = "\t", fileEncoding = "UTF-8")

# prepare tdm for writing

#this is the matrix containing more than 0 & 1 only

mytdm <- cbind(arabic\_m, names(arabic\_m))

write.table(mytdm, "termDocMatrix.txt", quote = FALSE, col.names = FALSE,

row.names = T, sep = "\t", fileEncoding = "UTF-8")

View(mytdm)

```

```{r}

############ STEP (11)#################

##we can do the sentiment Analysis of the text

##all that we need is a text (arabic text) and a sentiment sample (club\_sentiment)

####1ST: Read Sentiments

sents<-readLines("club\_sent.txt", encoding="UTF-8")

length(sents)

sents

table(sents)

###############################################

##2ND: start transliteration part

mydf<-data.frame(arabic\_text,sents, stringsAsFactors = F)

mydf$arabic\_text\_t <-transliterate(mydf$arabic\_text)

View(mydf)

##3RD: taking a training set & testing set

library(RTextTools)

# Create the document term matrix

dtMatrix <- create\_matrix(mydf["arabic\_text\_t"])

# Configure the training data

str(mydf)

mydf$sents<-factor(mydf$sents)

container <- create\_container(dtMatrix, mydf$sents, trainSize = 1:34, virgin = F)

##4TH: train a SVM Model (supporting vector machine)

model <- train\_model(container, "SVM", kernel="linear", cost=1)

##5TH: new data containing the word that we need to analyse

myterms<-c( "يتعلق","إلى" ,"درجة" ,"الحب","أحبب" ,"و","أفعل" ,"بعد "

,"ن" ,"أسكن" ,"في" ,"بيت" ,"الرب")

transliterate(myterms)

predictionData <- list( "yt3lQ","alA","drj0","al7b","a7bb","w","af3l",

"b3d","n","askn" ,"fy","byt","alrb")

#6TH: create a prediction document term matrix

predMatrix <- create\_matrix(predictionData, originalMatrix=dtMatrix)

predSize = length(predictionData);

##the following gives error if none of test data matches with training data.

predictionContainer <- create\_container(predMatrix, labels=rep(0,predSize), testSize=1:predSize, virgin=FALSE)

##7TH:we will get the predict result of the sentiment

results <- classify\_model(predictionContainer, model)

results

final\_report<-cbind(results,data=unlist(predictionData), myterms)

View(final\_report)

```

THERE EXIST OTHER METHOD TO DO A TEXT MINING ON ARABIC TEXT

SO IN THE EXPLAINED METHOD WE WERE:

1ST:TRANSLATING EACH ROW

2ND:FORMING OUR CORPUS

BUT HERE WE CAN IMMEDIATELY TRANSFORM THE WHOLE TEXT USING "TRANSLITERATE"

BY FOLLOWING THESE STEPS:

```{r}

##########step1: translate all the rabic text###################

text<-readLines("arabic.txt", encoding="UTF-8")

tttext<-transliterate(text)

##########step2: transforming the text into df##################

myartxt = data.frame(tttext, stringsAsFactors = F)

myartxt$doc\_id = c(1:nrow(myartxt))

myartxt$text = myartxt$tttext

myartxt = myartxt[,c(2,3,1)]

###########step3: create the corpus without the terms that we don't want########

tap.corpus <- Corpus(DataframeSource(myartxt))

tap.corpus<-tm\_map(tap.corpus,content\_transformer(tolower))

tap.corpus <- tm\_map(tap.corpus, removeWords, c("wqd", "ama", "byn", "f7sb",

"fhw", "،" ))

############step4:transform the data into matrix to see freq of each term########

tap.tdm <- TermDocumentMatrix(tap.corpus)

tap.m <- as.matrix(tap.tdm)

tap.v <- sort(rowSums(tap.m),decreasing=TRUE)

tap.d <- data.frame(word = names(tap.v),freq=tap.v)

############step5:visualize the most repeated#################

wordcloud2(data = tap.d)

```