Text analysis

ZIDAN Lama, ZIDAN Loubna et TAMATEKOU Akouvi Merveille

13/02/2023

Text analysis:

1. Preprocessing a Corpus

- Chargement des packages

```
rm(list=ls())
library('wordcloud')
```

```
## Le chargement a nécessité le package : RColorBrewer
```

```
library(tm) # Framework for text mining
```

```
## Le chargement a nécessité le package : NLP
```

```
library(RTextTools) # a machine learning package for text classification written in R
```

```
## Le chargement a nécessité le package : SparseM
```

```
##
## Attachement du package : 'SparseM'
```

```
## L'objet suivant est masqué depuis 'package:base':
##
##     backsolve
```

```
library(qdapDictionaries)
library(dplyr) # Data preparation and pipes $>$
```

```
##
## Attachement du package : 'dplyr'
```

```
## Les objets suivants sont masqués depuis 'package:stats':
##
##     filter, lag
```

```
## Les objets suivants sont masqués depuis 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2) # for plotting word frequencies
```

```
##
## Attachement du package : 'ggplot2'
```

```
## L'objet suivant est masqué depuis 'package:NLP':
##
##     annotate
```

```
library(SnowballC) # for stemming
```

```
##
## Attachement du package : 'SnowballC'
```

```
## Les objets suivants sont masqués depuis 'package:RTextTools':
##
##     getStemLanguages, wordStem
```

Le document soumis à notre analyse est un corpus de texte. Selon le site de l'Etudiant un corpus est un regroupement de documents qui sont réunis dans un but précis (points communs ou oppositions à déceler).

C'est également un ensemble fini de textes choisi comme base d'une étude , c'est une collection d'articles d'actualités.

Un corpus peut être aussi une collection articles d'actualité de Reuters ou les œuvres publiées de Shakespeare. Ces corpus sont donc composés d'articles , des histoires. Chaque unité est appelée "un document".

Notre étude s'appuie sur une section du corpus du prince Machiavel. Ce texte est une monographie et est donc découpé en morceau, considéré comme un document.

1.1 Les sources et les lectures du corpus

Pour cette partie nous allons utiliser le package 'tm'.Il s'agit du text- mining . Le text-mining c'est l'ensemble des méthodes qui permettent d'analyser un texte avec des méthodes statistiques . Il existe deux méthodes de traitement de texte sur R . La méthode avec le package tm et la méthode du tidy text- mining

D'abord, nous allons exécuter les deux commandes getsources () et getReaders()qui permettent d'afficher le type de sources et les readers

```
getSources()
```

```
## [1] "DataframeSource" "DirSource"       "URISource"       "VectorSource"
## [5] "XMLSource"       "ZipSource"
```

```
getReaders()
```

```
##  [1] "readDataframe"           "readDOC"
##  [3] "readPDF"                 "readPlain"
##  [5] "readRCV1"                "readRCV1asPlain"
##  [7] "readReut21578XML"        "readReut21578XMLasPlain"
##  [9] "readTagged"              "readXML"
```

Cette partie correspond aux chargement de la base csv . Il s'agit d'un document . Chaque ligne ets un document et des colonnes pour le texte et les métadonnées .

```
docs.df <-read.csv("C:\\Users\\loubn\\Desktop\\Zidan_Zidan_Tamatekou_9_Code\\mach.csv", header=TRUE) #r
docs <- Corpus(VectorSource(docs.df$text))
docs
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 188
```

Le corpus est donc composé de 188 documents. L'inspection des documents du corpus va se faire avec la commande inspect ()

```
# see the 9th document
inspect(docs[9])
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 1
##
## [1]  One of the reasons why historians have praised Philopoemen, the leader of the Achaean League, is
```

Pour voir le text nous la commande a utilisé est `as.chracter`

```
 # see content for 9th document
as.character(docs[[9]])
```

```
## [1] " One of the reasons why historians have praised Philopoemen, the leader of the Achaean League, i
```

La commande inspect() permet de sélectionner le document que l'on voudrait afficher. Par exemple inspect (docs[16]) permet d'afficher le 16 ème doccument

```
# See the 16th document
inspect(docs[16])
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 1
##
## [1]  Therefore, since a ruler cannot both practise this virtue of generosity and be known to do so wi
```

1.2 Fonctions de prétraitement

Les différentes applications d'analyses de texte ont presque les memes fonctionnements. Dans un premier temps l'application permet de :

-Tokeniser le texte en unigrammes (ou bigrammes, ou trigrammes)

- Convertir tous les caractères en minuscules

-Supprimer la ponctuation

- Supprimer les chiffres
- Suppression des mots d'arrêt, y compris les mots d'arrêt personnalisés
- "Stemming" des mots, ou lemmitisation. Il existe plusieurs algorithmes de alogrithmes. Porter est le plus populaire.

7. Création d'une matrice document-terme
8. Pondération des caractéristiques
9. Suppression des termes épars

Le package Gettransformation permet de voir les transformations disponibles dans le package

```
getTransformations()
```

```
## [1] "removeNumbers"     "removePunctuation" "removeWords"
## [4] "stemDocument"      "stripWhitespace"
```

La fonction 'tm_map()'permet d'appliquer une transformation à tous les documents du corpus

```
docs <- tm_map(docs, content_transformer(tolower)) # convert all text to lower case
```

```
## Warning in tm_map.SimpleCorpus(docs, content_transformer(tolower)):
## transformation drops documents
```

```
as.character(docs[[9]])
```

```
## [1] " one of the reasons why historians have praised philopoemen, the leader of the achaean league, i
```

De plus nous pouvons supprimer les mots manquants, les poctuations avec la fonction 'tm_map'

```
# remove Puncturation
docs <- tm_map(docs, removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(docs, removePunctuation): transformation drops
## documents
```

```
as.character(docs[[9]])
```

```
## [1] " one of the reasons why historians have praised philopoemen the leader of the achaean league is
```

```
# remove Numbers
docs <- tm_map(docs, removeNumbers)
```

```
## Warning in tm_map.SimpleCorpus(docs, removeNumbers): transformation drops
## documents
```

```
as.character(docs[[9]])
```

```
## [1] " one of the reasons why historians have praised philopoemen the leader of the achaean league is
```

```
# remove common words
docs <- tm_map(docs, removeWords, stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(docs, removeWords, stopwords("english")):
## transformation drops documents
```

```
stopwords("english") # check out what was removed
```

```
##   [1] "i"          "me"         "my"         "myself"     "we"
##   [6] "our"        "ours"       "ourselves"  "you"        "your"
##  [11] "yours"      "yourself"   "yourselves" "he"         "him"
##  [16] "his"        "himself"    "she"        "her"        "hers"
##  [21] "herself"    "it"         "its"        "itself"     "they"
##  [26] "them"       "their"      "theirs"     "themselves" "what"
##  [31] "which"      "who"        "whom"       "this"       "that"
##  [36] "these"      "those"      "am"         "is"         "are"
##  [41] "was"        "were"       "be"         "been"       "being"
##  [46] "have"       "has"        "had"        "having"     "do"
##  [51] "does"       "did"        "doing"      "would"      "should"
##  [56] "could"      "ought"      "i'm"        "you're"     "he's"
##  [61] "she's"      "it's"       "we're"      "they're"    "i've"
##  [66] "you've"     "we've"      "they've"    "i'd"        "you'd"
##  [71] "he'd"       "she'd"      "we'd"       "they'd"     "i'll"
##  [76] "you'll"     "he'll"      "she'll"     "we'll"      "they'll"
##  [81] "isn't"      "aren't"     "wasn't"     "weren't"    "hasn't"
##  [86] "haven't"    "hadn't"     "doesn't"    "don't"      "didn't"
##  [91] "won't"      "wouldn't"   "shan't"     "shouldn't"  "can't"
##  [96] "cannot"     "couldn't"   "mustn't"    "let's"      "that's"
## [101] "who's"      "what's"     "here's"     "there's"    "when's"
## [106] "where's"    "why's"      "how's"      "a"          "an"
## [111] "the"        "and"        "but"        "if"         "or"
## [116] "because"    "as"         "until"      "while"      "of"
## [121] "at"         "by"         "for"        "with"       "about"
## [126] "against"    "between"    "into"       "through"    "during"
## [131] "before"     "after"      "above"      "below"      "to"
## [136] "from"       "up"         "down"       "in"         "out"
```

```
## [141] "on"         "off"        "over"       "under"      "again"
## [146] "further"    "then"       "once"       "here"       "there"
## [151] "when"       "where"      "why"        "how"        "all"
## [156] "any"        "both"       "each"       "few"        "more"
## [161] "most"       "other"      "some"       "such"       "no"
## [166] "nor"        "not"        "only"       "own"        "same"
## [171] "so"         "than"       "too"        "very"
```

```
as.character(docs[[9]])
```

```
## [1] " one   reasons  historians  praised philopoemen  leader   achaean league    peacetime   always
```

```
# remove own stop words
docs <- tm_map(docs, removeWords, c("prince"))
```

```
## Warning in tm_map.SimpleCorpus(docs, removeWords, c("prince")): transformation
## drops documents
```

```
as.character(docs[[9]])
```

```
## [1] " one   reasons  historians  praised philopoemen  leader   achaean league    peacetime   always
```

```
# strip white space
docs <- tm_map(docs, stripWhitespace)
```

```
## Warning in tm_map.SimpleCorpus(docs, stripWhitespace): transformation drops
## documents
```

```
as.character(docs[[9]])
```

```
## [1] " one reasons historians praised philopoemen leader achaean league peacetime always thinking mili
```

```
# stem the document
docs <- tm_map(docs, stemDocument)
```

```
## Warning in tm_map.SimpleCorpus(docs, stemDocument): transformation drops
## documents
```

```
as.character(docs[[9]])
```

```
## [1] "one reason historian prais philopoemen leader achaean leagu peacetim alway think militari matte
```

1.3 Création d'une MNT

Une matrice de termes (MNT) de documents est une matrice dont les lignes sont des documents et les colonnes des termes et un compte de la fréquence des mots comme cellules de la matrice.

La fonction `DocumentTermMatrix()` permet de créer la matrice :

```
dtm <- DocumentTermMatrix(docs)

dtm
```

```
## <<DocumentTermMatrix (documents: 188, terms: 2367)>>
## Non-/sparse entries: 11750/433246
## Sparsity           : 97%
## Maximal term length: 15
## Weighting          : term frequency (tf)
```

tm nous permet également de convertir un corpus en MNT en complétant les étapes de pré-traitement en une seule étape.

```
dtm <- DocumentTermMatrix(docs,
           control = list(stopwords = TRUE,
                          tolower = TRUE,
                          removeNumbers = TRUE,
                          removePunctuation = TRUE,
                          stemming=TRUE))
```

1.4 Pondération

L'application de podération tf-idf est une étape importante de pré-traitement

Tf-idf est une statistique numérique destinée à refléter l'importance d'un mot pour un document dans une collection ou un corpus

La valeur tf-idf augmente proportionnellement au nombre de fois qu'un mot apparaît dans le document, mais est compensée par la fréquence du mot dans le corpus, qui permet d'ajuster le fait que certains mots apparaissent plus fréquemment en général

```
dtm.weighted <- DocumentTermMatrix(docs,
           control = list(weighting =function(x) weightTfIdf(x, normalize = TRUE),
                          stopwords = TRUE,
                          tolower = TRUE,
                          removeNumbers = TRUE,
                          removePunctuation = TRUE,
                          stemming=TRUE))
```

```
## Warning in TermDocumentMatrix.SimpleCorpus(x, control): custom functions are
## ignored
```

La fonction inspect() peut etre utilisée avec d'autres commandes comme 'dtm' et 'dtm.weighted' pour comparer les lignes et colonnes spécifiques .Ces fonctions sont utilisées pour comparer les 5 premières des fichiers

```
inspect(dtm[1:5,1:5])
```

```
## <<DocumentTermMatrix (documents: 5, terms: 5)>>
## Non-/sparse entries: 3/22
## Sparsity           : 88%
## Maximal term length: 8
```

```
## Weighting          : term frequency (tf)
## Sample             :
##     Terms
## Docs ·whether abandon abil abject abl
##    1        0       0    0      0   0
##    2        0       0    1      0   0
##    3        0       0    0      0   0
##    4        0       0    1      0   1
##    5        0       0    0      0   0
```

```
inspect(dtm.weighted[1:5,1:5])
```

```
## <<DocumentTermMatrix (documents: 5, terms: 5)>>
## Non-/sparse entries: 3/22
## Sparsity           : 88%
## Maximal term length: 8
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
## Sample             :
##     Terms
## Docs ·whether abandon       abil abject        abl
##    1        0       0 0.00000000      0 0.00000000
##    2        0       0 0.04077356      0 0.00000000
##    3        0       0 0.00000000      0 0.00000000
##    4        0       0 0.03102336      0 0.02277345
##    5        0       0 0.00000000      0 0.00000000
```

2. Exploration du DTM

2.1 Dimensions

Le DTM nous permet de voir la structure des documents et de déterminer le nombre total de document

```
# how many documents? how many terms?
dim(dtm)
```

```
## [1]  188 2356
```

2.2 Les fréquences

Pour obtenir les fréquences des termes sous forme de vecteur,nous allons convertir la matrice des termes du document en une matrice en utilisant la commande 'colsums' pour additionner les colonnes .

```
 # how many terms?
freq <- colSums(as.matrix(dtm))
length(freq)
```

```
## [1] 2356
```

La commande order() permet d'ordonner les fréquences

```r
# order
ord <- order(freq)
ord
```

```
##    [1]    1    4    6    8   10   12   14   17   18   28   29   30   32   33
##   [15]   36   37   41   44   54   55   57   58   60   61   62   65   66   67
##   [29]   70   72   74   75   76   89   91   92   93   95   97  102  109  111
##   [43]  113  114  115  117  120  121  122  123  128  136  138  139  140  142
##   [57]  148  152  154  155  162  163  166  167  168  170  173  179  180  181
##   [71]  182  184  185  187  188  190  193  197  198  199  202  205  207  209
##   [85]  212  214  215  217  219  221  222  224  225  227  228  231  232  233
##   [99]  238  240  242  243  246  248  251  255  256  258  259  260  261  262
##  [113]  265  266  270  272  274  279  281  282  285  293  297  298  303  304
##  [127]  305  306  307  308  309  313  314  315  317  318  323  324  325  326
##  [141]  331  333  335  336  337  342  348  349  352  355  356  359  360  361
##  [155]  362  363  364  366  368  369  376  377  380  381  384  389  391  398
##  [169]  410  411  413  414  415  417  420  422  425  427  433  434  435  437
##  [183]  440  445  449  450  452  457  458  461  466  467  468  469  471  472
##  [197]  474  482  484  485  489  494  496  497  499  501  505  506  507  508
##  [211]  509  510  511  512  513  516  519  520  525  531  533  535  536  540
##  [225]  544  545  547  551  552  556  557  559  561  563  569  571  573  575
##  [239]  576  578  580  581  584  586  587  589  590  591  592  593  594  598
##  [253]  599  600  606  610  614  619  621  624  627  628  632  636  639  640
##  [267]  643  644  647  648  649  650  652  653  660  666  669  670  672  674
##  [281]  676  678  680  681  691  693  694  695  703  709  715  719  721  723
##  [295]  726  727  728  734  736  738  740  742  744  745  746  747  752  754
##  [309]  757  758  762  764  768  770  772  773  776  777  778  782  785  788
##  [323]  791  792  794  795  799  800  801  803  807  809  811  816  817  822
##  [337]  823  825  827  832  833  835  843  847  849  853  861  863  866  867
##  [351]  876  878  880  881  882  884  885  888  890  892  896  904  907  914
##  [365]  915  918  920  921  924  928  929  930  931  932  935  941  946  947
##  [379]  949  950  951  952  953  954  955  957  958  960  961  962  964  971
##  [393]  975  976  978  979  980  981  982  985  986  987  988  991  993  997
##  [407] 1001 1003 1005 1007 1010 1011 1017 1018 1021 1024 1027 1028 1030 1031
##  [421] 1032 1033 1034 1036 1037 1038 1041 1043 1044 1045 1048 1054 1057 1059
##  [435] 1061 1062 1063 1067 1073 1076 1081 1082 1083 1086 1087 1092 1097 1098
##  [449] 1099 1101 1105 1106 1107 1109 1110 1111 1112 1115 1116 1119 1121 1122
##  [463] 1123 1128 1129 1131 1132 1133 1134 1135 1138 1139 1140 1147 1148 1156
##  [477] 1160 1161 1163 1165 1170 1171 1176 1177 1179 1183 1189 1192 1194 1195
##  [491] 1197 1198 1202 1204 1205 1211 1212 1213 1215 1219 1220 1225 1227 1233
##  [505] 1235 1237 1238 1242 1243 1248 1249 1250 1251 1254 1255 1256 1257 1260
##  [519] 1265 1269 1270 1271 1272 1273 1275 1276 1278 1283 1285 1286 1288 1292
##  [533] 1295 1302 1307 1311 1312 1314 1317 1326 1328 1329 1330 1331 1334 1336
##  [547] 1337 1341 1342 1343 1344 1347 1350 1353 1354 1355 1357 1359 1361 1362
##  [561] 1364 1366 1369 1376 1377 1382 1384 1388 1391 1394 1395 1396 1397 1403
##  [575] 1404 1408 1411 1413 1414 1415 1416 1417 1419 1420 1425 1426 1432 1437
##  [589] 1440 1441 1442 1444 1445 1446 1458 1462 1465 1466 1468 1473 1475 1477
##  [603] 1478 1480 1481 1484 1487 1491 1492 1495 1497 1499 1500 1502 1503 1506
##  [617] 1511 1514 1516 1517 1519 1520 1526 1529 1530 1533 1534 1535 1536 1537
##  [631] 1538 1540 1541 1543 1546 1548 1551 1554 1559 1562 1567 1571 1573 1574
##  [645] 1575 1579 1580 1581 1582 1585 1586 1589 1590 1592 1593 1595 1596 1597
##  [659] 1601 1606 1611 1612 1615 1616 1618 1621 1622 1624 1631 1637 1639 1648
##  [673] 1649 1653 1655 1657 1658 1659 1660 1663 1664 1665 1667 1669 1672 1675
##  [687] 1677 1682 1683 1684 1687 1690 1692 1697 1698 1704 1706 1708 1709 1711
```

```
##  [701] 1712 1714 1715 1717 1718 1719 1727 1728 1729 1730 1731 1733 1739 1740
##  [715] 1741 1747 1753 1758 1761 1763 1775 1776 1788 1789 1796 1798 1799 1800
##  [729] 1801 1803 1807 1819 1821 1822 1823 1826 1831 1832 1835 1836 1838 1841
##  [743] 1842 1846 1853 1855 1866 1867 1869 1871 1872 1874 1879 1884 1887 1892
##  [757] 1894 1896 1901 1903 1904 1907 1908 1910 1911 1914 1915 1916 1926 1930
##  [771] 1931 1935 1936 1939 1941 1942 1944 1945 1946 1947 1948 1951 1953 1954
##  [785] 1955 1956 1957 1960 1966 1968 1969 1971 1972 1973 1974 1975 1977 1978
##  [799] 1979 1981 1984 1986 1989 1991 1993 1994 1995 1996 2006 2007 2008 2011
##  [813] 2012 2013 2014 2020 2021 2023 2027 2028 2030 2034 2038 2039 2040 2041
##  [827] 2043 2044 2045 2047 2049 2050 2051 2057 2060 2063 2065 2066 2067 2068
##  [841] 2069 2070 2077 2078 2079 2082 2084 2090 2091 2094 2095 2096 2097 2098
##  [855] 2101 2103 2104 2105 2107 2108 2109 2113 2116 2118 2121 2122 2124 2125
##  [869] 2126 2127 2130 2131 2132 2135 2137 2141 2143 2147 2149 2154 2155 2157
##  [883] 2159 2160 2162 2163 2164 2165 2166 2171 2173 2174 2175 2176 2177 2178
##  [897] 2182 2183 2184 2185 2187 2189 2192 2193 2194 2195 2196 2197 2199 2200
##  [911] 2203 2206 2207 2209 2216 2219 2221 2224 2226 2227 2229 2232 2235 2236
##  [925] 2237 2242 2246 2249 2251 2253 2255 2258 2259 2260 2261 2263 2269 2270
##  [939] 2272 2273 2275 2276 2277 2278 2283 2285 2286 2288 2290 2291 2293 2299
##  [953] 2305 2312 2317 2318 2319 2324 2325 2328 2329 2330 2331 2332 2335 2339
##  [967] 2340 2341 2342 2343 2344 2345 2346 2347 2348 2349 2350 2353 2354 2356
##  [981]    9   13   22   27   35   49   59   73   86  100  105  118  124  126
##  [995]  130  143  145  146  147  149  153  165  172  176  186  196  206  210
## [1009]  211  218  226  234  237  241  247  250  254  257  263  267  269  275
## [1023]  277  286  287  291  300  319  334  340  354  358  386  392  396  402
## [1037]  403  406  407  416  418  419  429  430  438  439  441  448  451  455
## [1051]  456  463  475  480  486  492  493  503  504  514  532  538  539  553
## [1065]  554  560  565  566  568  570  574  577  588  597  611  612  613  617
## [1079]  622  623  633  634  635  638  651  657  662  677  684  686  688  692
## [1093]  696  697  712  732  735  748  751  759  763  765  775  783  790  804
## [1107]  808  812  813  826  837  838  839  845  850  852  864  870  874  879
## [1121]  895  899  900  902  905  906  909  912  913  917  925  927  933  937
## [1135]  945  967  968  969  977  984  990  998 1002 1004 1008 1022 1023 1025
## [1149] 1026 1042 1049 1053 1056 1058 1071 1074 1075 1078 1088 1090 1103 1118
## [1163] 1126 1130 1167 1168 1173 1190 1206 1209 1229 1244 1247 1252 1262 1277
## [1177] 1279 1281 1282 1291 1296 1297 1299 1300 1304 1309 1319 1320 1323 1324
## [1191] 1327 1338 1346 1348 1351 1356 1365 1367 1371 1374 1390 1400 1405 1406
## [1205] 1409 1410 1428 1429 1452 1459 1461 1470 1482 1483 1486 1490 1501 1505
## [1219] 1513 1523 1528 1531 1542 1545 1552 1557 1584 1587 1591 1602 1603 1608
## [1233] 1613 1620 1623 1625 1628 1633 1634 1651 1676 1681 1696 1699 1702 1705
## [1247] 1707 1710 1723 1735 1736 1742 1751 1752 1760 1781 1782 1784 1791 1792
## [1261] 1794 1811 1812 1817 1818 1829 1843 1854 1858 1876 1878 1885 1886 1893
## [1275] 1899 1900 1909 1913 1943 1950 1958 1963 1964 1967 1976 1985 1988 2001
## [1289] 2009 2016 2025 2033 2037 2042 2046 2053 2055 2056 2062 2064 2071 2102
## [1303] 2106 2114 2117 2129 2140 2142 2150 2151 2172 2179 2180 2186 2191 2198
## [1317] 2201 2205 2212 2217 2222 2225 2239 2244 2245 2247 2248 2262 2266 2271
## [1331] 2274 2281 2282 2287 2304 2308 2313 2314 2315 2322 2336 2337 2338 2355
## [1345]   16   23   40   46   56   68   79   96   99  110  119  131  132  141
## [1359]  150  151  156  160  161  174  192  236  273  289  292  316  320  327
## [1373]  330  338  351  353  378  397  409  423  428  436  454  470  478  521
## [1387]  523  543  558  585  595  596  601  607  616  618  654  655  687  690
## [1401]  700  702  707  716  750  753  755  756  767  771  780  789  793  796
## [1415]  802  805  806  820  824  828  829  834  840  862  891  893  903  956
## [1429]  959  983 1015 1029 1055 1060 1069 1077 1080 1085 1091 1100 1102 1108
## [1443] 1114 1117 1120 1164 1166 1199 1200 1201 1224 1246 1266 1294 1372 1381
```

```
## [1457] 1389 1392 1393 1402 1430 1449 1464 1469 1472 1493 1496 1504 1510 1515
## [1471] 1522 1539 1572 1588 1605 1632 1640 1642 1643 1650 1666 1680 1685 1688
## [1485] 1694 1716 1720 1722 1748 1749 1757 1759 1766 1768 1769 1772 1777 1780
## [1499] 1783 1787 1806 1808 1816 1827 1845 1864 1891 1897 1918 1921 1949 1961
## [1513] 1999 2004 2017 2018 2019 2026 2031 2054 2058 2073 2086 2089 2099 2110
## [1527] 2112 2115 2128 2139 2144 2148 2161 2167 2181 2218 2220 2223 2231 2250
## [1541] 2254 2265 2301 2333    2   31   39   45   51   81   88  104  112  134
## [1555]  137  164  208  239  253  370  385  394  395  444  453  464  479  524
## [1569]  526  528  529  534  542  562  567  604  615  641  642  667  682  685
## [1583]  689  699  705  720  725  730  741  743  779  781  821  841  857  877
## [1597]  886  887  901  934  936  942  943 1000 1012 1035 1046 1050 1052 1072
## [1611] 1113 1144 1158 1180 1208 1210 1216 1230 1263 1298 1308 1325 1332 1333
## [1625] 1339 1380 1401 1427 1433 1438 1447 1450 1463 1476 1488 1498 1524 1525
## [1639] 1549 1553 1556 1563 1594 1607 1609 1626 1635 1644 1645 1646 1656 1670
## [1653] 1673 1713 1721 1725 1732 1746 1786 1797 1809 1825 1833 1857 1860 1888
## [1667] 1890 1925 1929 1934 1940 1990 2000 2002 2059 2080 2169 2202 2211 2228
## [1681] 2230 2252 2316 2326 2334    7   20   34   38   53   63   69  103  135
## [1695]  144  191  276  301  357  374  379  390  408  412  421  459  488  518
## [1709]  522  579  602  630  646  671  673  675  713  724  729  798  818  846
## [1723]  858  944  996 1040 1047 1068 1084 1143 1155 1172 1184 1185 1226 1236
## [1737] 1241 1335 1383 1418 1424 1436 1454 1485 1489 1508 1532 1561 1564 1662
## [1751] 1678 1734 1743 1750 1770 1771 1810 1820 1830 1837 1839 1861 1862 1882
## [1765] 1912 1922 1927 1937 1959 1980 2022 2048 2076 2087 2088 2153 2204 2210
## [1779] 2215 2234 2240 2257 2292 2294 2327   11   48  107  133  195  223  244
## [1793]  245  339  341  365  372  424  431  476  564  608  625  664  714  737
## [1807]  749  769  797  831  854  865  871  872  883 1070 1096 1142 1187 1214
## [1821] 1287 1340 1352 1379 1422 1431 1474 1507 1518 1527 1577 1598 1638 1738
## [1835] 1764 1765 1790 1795 1863 1868 1877 1905 1919 1938 1983 2005 2015 2052
## [1849] 2083 2136 2145 2214 2238 2264 2284 2300 2306 2320   19   78  157  249
## [1863]  290  296  299  350  375  383  399  401  465  491  603  626  665  683
## [1877]  731  869  923  948  966 1006 1169 1258 1259 1368 1434 1451 1453 1456
## [1891] 1544 1550 1578 1600 1668 1671 1686 1691 1695 1774 1840 1889 1895 1906
## [1905] 1962 2024 2032 2085 2119 2156 2188 2256 2295 2321   80  108  116  178
## [1919]  252  311  312  332  388  582  609  663  706  760  860  897  939  972
## [1933] 1014 1093 1141 1150 1186 1188 1191 1240 1310 1345 1370 1423 1435 1558
## [1947] 1778 1779 1785 1793 1933 2123 2158   42   43   64  177  189  264  294
## [1961]  322  367  446  447  462  473  481  483  500  541  555 1124 1223 1234
## [1975] 1274 1358 1399 1583 1604 1630 1859 1902 1928 2138 2298 2310   15   87
## [1989]  159  169  194  229  230  517  658  659  815  856  916 1094 1095 1127
## [2003] 1239 1280 1509 1555 1566 1851 1852 1856 1865 1873 1970 2146 2323  125
## [2017]  487  537  940  999 1065 1079 1145 1146 1159 1221 1439 1619 1647 1661
## [2031] 1674 1700 1737 1754 1762 1880 2190 2243 2307   47   52  171  268  271
## [2045]  302  345  546  661  739  875  898  919 1174 1322 1627 1745 1870 1923
## [2059] 2036 2170   90  387  404  426  490  498  711  784 1125 1321 1407 1421
## [2073] 1457 1617 1629 1701 1726 1755 2029 2309   94  280  717  766  830  873
## [2087]  908  911 1020 1066 1175 1479 1547 1610 1802 1813 1920 1924 1932 2168
## [2101]  216  629  631  851  974  995 1137 1178 1232 1293 1306 1467 1494 1599
## [2115] 1848 1883 1982 2100 2120  203  213  321  442  698  994 1051 1064 1089
## [2129] 1104 1315 1373 1471 1724 1767 1849 2003 2010 2111 2352  204  278  283
## [2143]  329  373  460  527  679  848 1019 1152 1203 1303 1313 1316 1679 1997
## [2157] 2241  371  382  393  583  708  722  733  855  910 1181 1193 1231 1455
## [2171] 1636 1641 1652 1703 1805 1881 1952 2296   98  288  295  989 1222 1228
## [2185] 1443 1844 1875 2081 2351   50  343  495  637  718  774  963 1850 1987
## [2199] 1998 2075 2280 2303   26   82  183  200  220  605 1182 1412 1654 1773
```

```
## [2213] 2208   83  400  515  761 1016 1039 1157 1217 1521 1834   77  235  310
## [2227]  530  548 1245 1290 1349 1570 1689 1828 2035  656  701 1386 1744 1756
## [2241]  344  502  710  844  894 1013 2092 2268  432  814 1151 1207 1560 2233
## [2255]  889 2134 2297   24  810 1196 1568 1824 1847 2061  328  549  704  970
## [2269]  973 1162 1804  101  127  550 1253 1318  347   84  668  859 1218 1305
## [2283] 2311  175  926 1378 1398  572  868  965 1569    3  346 1009 1154   21
## [2297] 1284 1375 2152  477  922 1149 1814  106  443  836 1289 2072   71 1565
## [2311]  158  620 1136  787  992 1917  819 2074  786 1614 1261 2133 1363 1264
## [2325] 1267 1693 1965  645 1992 1385 1460 1360  842 1153  284    5  129   25
## [2339]  938 1268 2289 2213 2279  201  405 2267 1898 1387 2093   85 1301 1512
## [2353] 1448 1576 2302 1815
```

```
# Least frequent terms
freq[head(ord)]
```

```
## ·whether   abject   ablest abovenam   absorb   access
##        1        1        1        1        1        1
```

```
# most frequent
freq[tail(ord)]
```

```
##   men peopl   one power  will ruler
##    95    98   168   169   251   280
```

2.3 Tracer les fréquences

Le graphique ci dessous montre la fréquence des termes Pour les mots qui sont utilisés 5 ou 10 fois

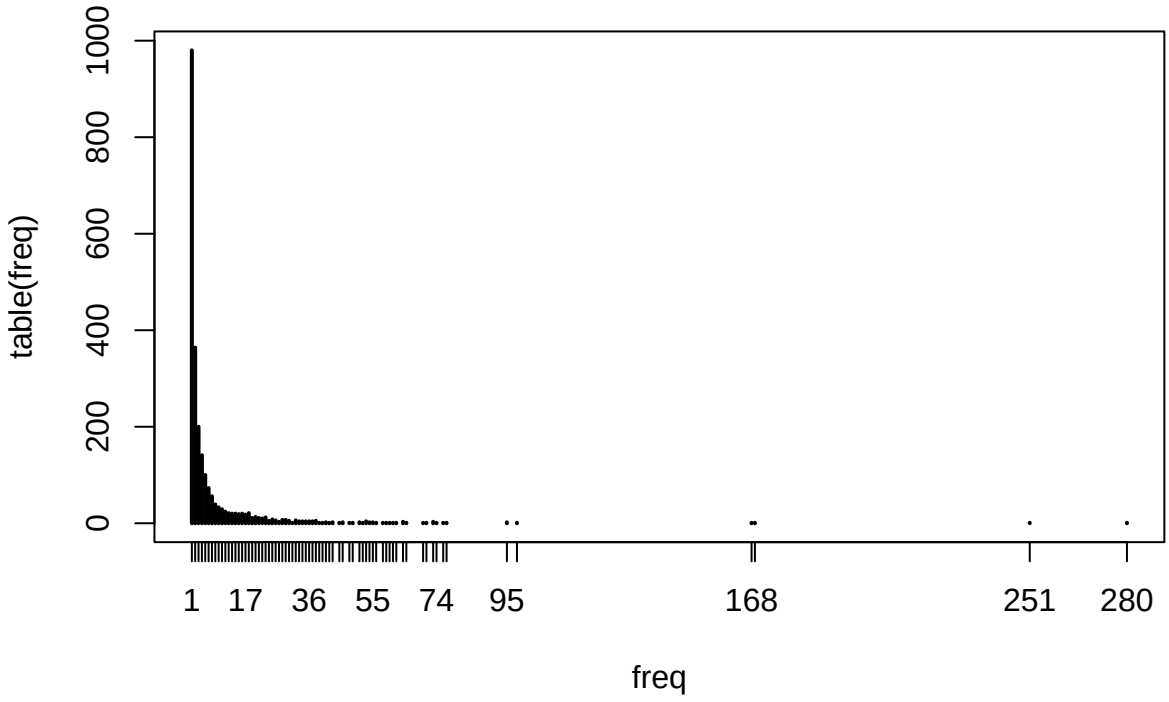
```
# frequency of frenquencies
head(table(freq),9)
```

```
## freq
##   1   2   3   4   5   6   7   8   9
## 980 364 200 141 100  73  56  39  33
```

```
tail(table(freq),9)
```

```
## freq
##  74  76  77  95  98 168 169 251 280
##   1   1   1   2   1   1   1   1   1
```

```
# plot
plot(table(freq))
```



Afin de montrer les termes les plus fréquents nous pouvons réorganiser les colonnes du DTM:

```
dtm.ordered <- dtm[,order(freq, decreasing = T)]
inspect(dtm.ordered[1:5,1:5])
```

```
## <<DocumentTermMatrix (documents: 5, terms: 5)>>
## Non-/sparse entries: 10/15
## Sparsity           : 60%
## Maximal term length: 5
## Weighting          : term frequency (tf)
## Sample             :
##     Terms
## Docs one peopl power ruler will
##    1   0     0     0     1    1
##    2   3     0     0     1    3
##    3   0     0     0     0    0
##    4   0     0     0     1    1
##    5   3     0     0     1    1
```

2.4 Exploration de la fréquence des mots

Pour explorer les mots et associations le package TM dispose de plusieurs commandes qui peuvent etre utiles:

```r
# Have a look at common words
findFreqTerms(dtm, lowfreq=100) # words that appear at least 100 times
```

```
## [1] "one"   "power" "ruler" "will"
```

```r
# Which words correlate with "war"?
findAssocs(dtm, "war", 0.3)
```

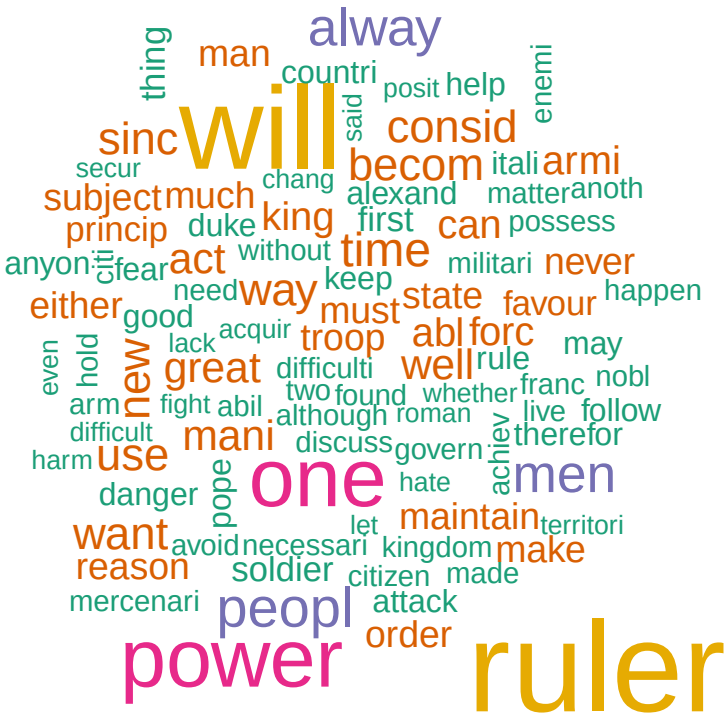
```
## $war
##     wage    fight  antioch     argu     brew    induc      lip    maxim
##     0.73     0.52     0.45     0.45     0.45     0.45     0.45     0.45
##  relianc     sage    trifl  postpon     mere     evil    avoid     flee
##     0.45     0.45     0.45     0.41     0.35     0.34     0.32     0.32
##   occupi     glad glorious    heard     hunt ineffect     knew   produc
##     0.32     0.30     0.30     0.30     0.30     0.30     0.30     0.30
##  tempori
##     0.30
```

Le graphique des nuages de mot permet d' afficher les termes les plus communs

```r
# plot the most frequent words
freq <- sort(colSums(as.matrix(dtm)),decreasing=TRUE)
head(freq)
```

```
## ruler  will power   one peopl alway
##   280   251   169   168    98    95
```

```r
# wordcoulds!
library(wordcloud)
set.seed(123)
wordcloud(names(freq), freq, max.words=100, colors=brewer.pal(6,"Dark2"))
```



2.5 Supprimer les termes épars.

Afin de supprimer les termes épars nous pouvons utiliser la fonction 'removeSparseTerms'

```
dtm.s <- removeSparseTerms(dtm,.9)
dtm # 2365 terms
```

```
## <<DocumentTermMatrix (documents: 188, terms: 2356)>>
## Non-/sparse entries: 11679/431249
## Sparsity           : 97%
## Maximal term length: 15
## Weighting          : term frequency (tf)
```

```
dtm.s # 135 terms
```

```
## <<DocumentTermMatrix (documents: 188, terms: 135)>>
## Non-/sparse entries: 4304/21076
## Sparsity           : 83%
## Maximal term length: 12
## Weighting          : term frequency (tf)
```

```
dtm.s.matrix <- as.matrix(dtm.s)
```