Assignment_1

Assignment 1

1. Create a repo for this project on github, and clone to your local machines.
2. Download the text as plain text from https://www.gutenberg.org/ebooks/100.txt.utf-8.
3. The following code will read the file into R. You will need to change the path in the setwd call to point to your local repo. Only use the given file name for the Shakespeare text file, to facilitate marking.

- setwd("put/your/local/repo/location/here") ## comment out of submitted
  a <- scan("shakespeare.txt",what="character",skip=83,nlines=196043-83, fileEncoding="UTF-8")

```
#setwd("put/your/local/repo/location/here") ## comment out of submitted
a <- scan("pg100.txt", what="character", skip=83, nlines=196043-83,
          fileEncoding="UTF-8")
```

4. Some pre-processing of a is needed.

a. The text contains stage directions, starting [and ending], but with one or two unmatched brackets. These stage directions need to be removed. A reasonable strategy is to locate all words in 'a' that contain [using grep. Then loop through the identified word locations in 'a' using grep again to search for the corresponding] within the next 100 words. In this way you can build a record of all the words corresponding to stage directions. Once this record is assembled, use it to delete the stage directions.

```
#locate all words in 'a' that contain '['
i_ob <- grep("[", a, fixed=TRUE) #ob: opening bracket
```

```
#create an empty list to store the locations of the unmatched ['
i_uob <- c()

for (i in i_ob) {
  #index of corresponding ']'
  i_uob_temp <- grep("]", a[i:(i+100)], fixed=TRUE)

  #if the corresponding ']' is not found within the next 100 words,
  if (length(i_uob_temp)==0) {
    #store the location of '['
    i_uob <- append(i_uob, i)
  }
}
```

```
a_s <- a

#delete the unmatched stage directions
a_s[i_uob] <- gsub('\\[', '', a[i_uob])
```

b. Words that are fully upper case are character names indicating who is speaking, or headings of various sorts. Similarly, numbers expressed as arabic numerals are not part of the text. All should be removed. You can identify the fully upper case words and numerals by testing whether words are equal to their upper case version produced by toupper. Note that "I" and "A" are special cases which should not be removed!

```
i_u <- c()

#locate words that are fully upper case, except for 'I' and 'A', and
#numbers expressed as arabic numerals
for (i in 1:length(a_s)) {
  if ((a_s[i]==toupper(a_s[i])) &
      (a_s[i]!='I') & (a_s[i]!='A')) {
    i_u <- append(i_u, i)
  }
}
a_s[i_u[1:10]]
```

```
[1] "THE"     "SONNETS" "1"       "2"       "3"       "4"       "5"
[8] "6"       "7"       "8"
```

```
a_s2 <- a_s

# delete character names and arabic numerals
a_s2 <- a_s[-i_u]
```

c. Using gsub you should remove "_" and "-" from words (the last one is debatable actually).

```
a_s2 <- gsub("_", "", a_s2)
a_s2 <- gsub("-", "", a_s2)
```

d. Write a function, called split_punct, which takes a vector of words as input along with a vector of punctuation marks (e.g. ",", "." etc.). The function should search for each word containing the punctuation marks, remove them from the word, and add the mark as a new entry in the vector of words, after the word it came from. The updated vector should be what the function returns. For example the input vector "An" "omnishambles," "in" "a" "headless" "chicken" "factory."
should become output vector
"An" "omnishambles" "," "in" "a" "headless" "chicken" "factory" "."
Functions grep, rep and gsub are the ones to use for this task. Beware that some punctuation marks are special characters in regular expressions, which grep and gsub can interpret. The notes tell you how to deal with this. The idea is that the model will treat punctuation just like words.

```
split_punct <- function(x) {
  puncs <- c(",", ".", ";", "!", ":", "?") #list of punctuations
  for (punc in puncs) {
    ii <- grep(punc, x, fixed=TRUE) ## which elements of text include punc.?
    if (length(ii)!=0) { #if x includes corresponding punc
      xs <- rep("", length(ii)+length(x)) ## vector to store the words and puncs
      iis <- ii+1:length(ii) ## where should puncs go in xs?
      xs[iis] <- substr(x[ii], nchar(x[ii]), nchar(x[ii])) ## insert puncs
      xs[-iis] <- gsub(paste('\\', punc, sep=""), "", x) ## insert words without puncs
      x <- xs
    }
  }
  return(x)
}

x <- c("An", "omnishambles,", "in", "a", "headless", "chicken", "factory.")
split_punct(x)
```

```
[1] "An"            "omnishambles" ","             "in"            "a"
[6] "headless"      "chicken"      "factory"      "."
```

e. Use your split_punct function to separate the punctuation marks, “,”, “.”, “;”, “!”, “:” and “?” from words they are attached to in the text.

```
a_s3 <- split_punct(a_s2)
a_s3[1:30]
```

```
 [1] "From"      "fairest"   "creatures" "we"        "desire"    "increase"
 [7] ","         "That"      "thereby"   "beauty’s" "rose"      "might"
[13] "never"     "die"       ","         "But"       "as"        "the"
[19] "riper"     "should"    "by"        "time"      "decease"   ","
[25] "His"       "tender"    "heir"      "might"     "bear"      "his"
```

f. For simplicity convert your cleaned word vector a to lower case, which is what will be used from here on.

```
a_s4 <- tolower(a_s3)
a_s4[1:30]
```

```
 [1] "from"      "fairest"   "creatures" "we"        "desire"    "increase"
 [7] ","         "that"      "thereby"   "beauty’s" "rose"      "might"
[13] "never"     "die"       ","         "but"       "as"        "the"
[19] "riper"     "should"    "by"        "time"      "decease"   ","
[25] "his"       "tender"    "heir"      "might"     "bear"      "his"
```