CT5102: Programing for Data Analytics

Assignment 8: Text Mining in R using **`stringr`**

The aim of this assignment is to use stringr functions (along with dplyr and ggplot2) to analyse text. The text is Chapter 2 (see github site) of Ulysses[1] by James Joyce. Make use of the file already on github[2] as a starting point, as it contains a set of *stopwords*[3] and invalid characters.

```
f_pre <- readLines("datasets/Text Mining/Ulysses/Chapter 02.txt")
```

This should create a vector of 429 elements, each for a line of text, for example:

```
> str(f_pre)
 chr [1:429] "YOU, COCHRANE, WHAT CITY SENT FOR HIM?" "-- Tarentum, sir."

> f_pre[1:10]
 [1] "YOU, COCHRANE, WHAT CITY SENT FOR HIM?"
 [2] "-- Tarentum, sir."
 [3] ""
 [4] "-- Very good. Well?"
 [5] ""
 [6] "-- There was a battle, sir."
 [7] ""
 [8] "-- Very good. Where?"
 [9] ""
[10] "The boy's blank face asked the blank window."
```

The steps are as follows:

1. Write a function to convert the vector of lines to a vector where each element is a word.

```
> f_pre_vec <- convert_to_words_vector(f_pre)
>
> str(f_pre_vec)
 chr [1:4512] "YOU," "COCHRANE," "WHAT" "CITY" "SENT" "FOR" "HIM?" "--" ...
>
> f_pre_vec[1:10]
 [1] "YOU,"      "COCHRANE," "WHAT"      "CITY"      "SENT"      "FOR"
 [7] "HIM?"      "--"        "Tarentum," "sir."
```

2. Write a function to pre-process the data, where this function will:

    - Remove invalid characters (specified in the variable `invalid_characters`)
    - Remove all empty strings
    - Convert each word to lowercase
    - Remove all stopwords (contained in variable `stopwords`)

```
str(f_post)
 chr [1:2261] "cochrane" "city" "sent" "tarentum" "sir" "good" "battle"
"sir"

3.
```

---

[1] http://www.online-literature.com/james_joyce/ulysses/2/

[2] https://github.com/JimDuggan/CT5102/blob/master/code/course/08%20stringr/08%20Assignment%20Start.R

[3] In computing, stop words are words which are filtered out before or after processing of natural language data (text)

4. Create a tibble with 3 columns. The first is for each word is the processed text, the second is a regex search pattern (note use of anchors) that can be used to search the text, and the third is the word length.

```
> ans
# A tibble: 1,332 x 3
   Words    Pattern    WLength
   <chr>    <chr>        <int>
 1 cochrane ^cochrane$       8
 2 city     ^city$           4
 3 sent     ^sent$           4
 4 tarentum ^tarentum$       8
 5 sir      ^sir$            3
 6 good     ^good$           4
 7 battle   ^battle$         6
 8 boys     ^boys$           4
 9 blank    ^blank$          5
10 face     ^face$           4
# ... with 1,322 more rows
```

5. Create a tibble that contains the frequency of word length occurrence for the text. The following data should be generated.

```
> freq
# A tibble: 16 x 2
   WLength WFrequency
     <int>      <int>
 1       1          2
 2       2          9
 3       3         66
 4       4        219
 5       5        267
 6       6        223
 7       7        220
 8       8        155
 9       9         83
10      10         43
11      11         24
12      12          9
13      13          9
14      14          1
15      16          1
16      18          1
```

6. Plot the results for the chapter text analysis as follows: