Homework 5 - Working with text in R

Due: November 12, 2019

Getting Started

In this lab, we will work on several simple use cases. While they are make-up cases, the skills you will use are essential when dealing with text. As usual, we start the lab by loading the necessary libraries

```
# You might need to install the packages if they are not available
library(rvest)
library(tidyverse)
```

Use case 1 - Cleaning Text

It is often a case where you need to join two tables where the key identifiers are name in string format. For example, we wish to join the following two table

County	State
De Witt County	IL
Lac qui Parle County	MN
Lewis And Clark County	MT
St John the Baptist Parish	LA

to be joined with

County	Population
DeWitt	16798
Lac Qui Parle	8067
Lewis & Clark	55716
St. John the Baptist	43044

Question 1: Our task in this use case is to perform string operations (e.g., replacements, deletions, transformations) on the `County` columns (in both tables), so that a `join` command will successfully join the tables.

```
counties1 <- tibble(
  county = c('De Witt County', 'Lac qui Parle County',
             'Lewis and Clark County', 'St John the Baptist Parish'),
  state = c("IL", "MN", "MT", "LA"))
counties2 <- tibble(
  county = c('DeWitt  ', 'Lac Qui Parle',
             'Lewis & Clark ', 'St. John the Baptist'),
  population = c(16798, 8067, 55716, 43044))

# Your code goes here

# Once you finish, the following code should give
```

```r
# the combined table
inner_join(counties1, counties2, by = "county")
```

Use Case 2: Extracting Fields

This case deals with the log file. The file `smalllog.txt` contains just 3 entries of the system and we wish to extract the date and time of the events logged in the log file.

```r
entries <- readLines("smallLog.txt")
entries
```

Question 2: Use regular expression to extract the date and time.

```r
# Your code does here
```

Working with date and time

Date and time are typically hard/frustrating to work with in R; they are most often represented in string forms while they are allowed some arithmetic computation. The R package `lubridate` provides many useful functions that make date and time manipulation much easier.

The remaining of this subsection gives some highlight of the packages

```r
# Install the package by uncomment the following line and run it
# install.packages("lubridate")
library(lubridate)
```

1. Easy and fast parsing of time-date: `ymd()`, `ymd_hms()`, `dmy()`, `dmy_hms()`, `mdy()`, . . .

```r
ymd(20101215)
mdy("4/1/17")
dmy("2/Feb/2005")
```

2. Simple functions to get and set components of a date-time, such as `year()`, `month()`, `mday()`, `hour()`, `minute()` and `second()`:

```r
bday <- dmy("14/10/1979")
month(bday)
wday(bday, label = TRUE)
year(bday) <- 2016
wday(bday, label = TRUE)
```

3. Helper functions for handling time zones: `with_tz()`, `force_tz()`

```r
time <- ymd_hms("2010-12-13 15:30:30")
time

# Changes printing
with_tz(time, "America/Chicago")

# Changes time
force_tz(time, "America/Chicago")
```

If you are interested to learn more, the best place to start is the date and times chapter in the book *"R for data science"*.

Question 3: Which date of the week are dates you extracted in question 2?

Use Case 3: Deriving Features

Local health departments frequently inspect restaurants in their communities to make sure the food is processed and prepared in the safe manner. In this use case, we are looking at the dataset about restaurants' violation as well as their safety scores.

```r
violations <- read_csv("violations.csv")
head(violations)
```

Question 4: Many violations have been corrected and recorded in the description. You are asked to extract and to store the date when the violations were corrected.

```r
# Your code goes here
```

Question 5: You are asked to list the 20 most common violations in the dataset. *Hint:* Remove the info about violation correction before doing any further analysis.

```r
# Your code goes here
```

The most time-consuming step in text analysis is to classify (or label) the description of the violation as one of a few categories. Another way is to label a description based on the existence of a few pre-specified words. For example, a violation is about the cleanliness if its description has "unclean", "inadequately cleaned", "unsanitary", etc. From the regular expression point of view, if a regex is "clean", it will detect both "unclean" and "inadequately cleaned". So, we can create a dummy variable for cleanliness as

```r
violations <- mutate(violations, is_clean = str_detect(description, "clean|sanit"))
```

Question 6: Create dummies for other below categories. *Note:* Beware of the case-sensitivity of strings.

Feature	selected words
High risk	high risk
Vermin	vermin
Surfaces	wall, ceiling, floor, surface
Humans	hands, glove, hair, nail
Permits	permit, certif

Question 7: The file `inspections.csv` contains the score for each restaurant after the inspection. Create the boxplot to compare the number of (type of) violations and the scores the restaurant received.