Uber_Project

Hurgland-Nick KELIET

26 mai 2019

R Markdown

Chargement des extensions nécessaire

```
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.1.0       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0
```

```
## -- Conflicts -------------------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

Load in R the .csv file ?

```
#Chargeons les différents dataset dans des dataframes.
 uber_data_apr14 <- read_csv("uber-raw-data-apr14.csv")
```

```
## Parsed with column specification:
## cols(
##   `Date/Time` = col_character(),
##   Lat = col_double(),
##   Lon = col_double(),
##   Base = col_character()
## )
```

```
 uber_data_may14 <- read_csv("uber-raw-data-may14.csv")
```

```
## Parsed with column specification:
## cols(
##   `Date/Time` = col_character(),
##   Lat = col_double(),
##   Lon = col_double(),
##   Base = col_character()
## )
```

```
 uber_data_jun14 <- read_csv("uber-raw-data-jun14.csv")
```

```
## Parsed with column specification:
## cols(
##   `Date/Time` = col_character(),
##   Lat = col_double(),
##   Lon = col_double(),
##   Base = col_character()
## )
```

```
uber_data_jul14 <- read_csv("uber-raw-data-jul14.csv")
```

```
## Parsed with column specification:
## cols(
##   `Date/Time` = col_character(),
##   Lat = col_double(),
##   Lon = col_double(),
##   Base = col_character()
## )
```

```
uber_data_aug14 <- read_csv("uber-raw-data-aug14.csv")
```

```
## Parsed with column specification:
## cols(
##   `Date/Time` = col_character(),
##   Lat = col_double(),
##   Lon = col_double(),
##   Base = col_character()
## )
```

```
uber_data_sep14 <- read_csv("uber-raw-data-sep14.csv")
```

```
## Parsed with column specification:
## cols(
##   `Date/Time` = col_character(),
##   Lat = col_double(),
##   Lon = col_double(),
##   Base = col_character()
## )
```

Bind all the data files into one. We may use the bind_rows() function under the dplyr library in R.

```
# Concaténons les 6 tables.
uber <- bind_rows(uber_data_apr14,uber_data_may14,uber_data_jun14,uber_data_jul14,uber_data_aug14,uber_
```

get the summary of the data to get an idea of what you are dealing with.

```
summary(uber)
```

```
##   Date/Time              Lat             Lon              Base
##  Length:4534327     Min.   :39.66   Min.   :-74.93   Length:4534327
##  Class :character   1st Qu.:40.72   1st Qu.:-74.00   Class :character
##  Mode  :character   Median :40.74   Median :-73.98   Mode  :character
```

```
##                     Mean   :40.74   Mean   :-73.97
##                     3rd Qu.:40.76   3rd Qu.:-73.97
##                     Max.   :42.12   Max.   :-72.07
```

```
#Ici on peut voir dans le summary que la taille total de nos lignes combinées est bien la bonne.
```

DATA PREPARATION

This step consists of cleaning and rearranging your data so that you can work on it more easily. It's a good idea to first think of the sparsity of the dataset and check the amount of missing data . You can see that the first column is Date.Time. To be able to use these values, you need to separate them. So let's do that, you can use the lubridate library for this. Lubridate makes it simple for you to identify the order in which the year, month, and day appears in your dates and manipulate them.

```
## Première étape, nous allons séparer la colonne date.time en deux colonnes (date et time) avec Lubrid
## Créons une nouvelle colonne
col_date_time <- uber$`Date/Time`

## Séparons les colonnes et convertissons les en dataframes pour notre futur Dataframe
# Date
date_time <- mdy_hms(col_date_time)
Month <- month(date_time)
Month <- as.data.frame(Month)
Day <- day(date_time)
Day <- as.data.frame(Day)
Year <- year(date_time)
Year <- as.data.frame(Year)

# Time
Hour <- hour(date_time)
Hour <- as.data.frame(Hour)
Minute <- minute(date_time)
Minute <- as.data.frame(Minute)
Second <- second(date_time)
Second <- as.data.frame(Second)

# Récupérons la date pour en extraire le Weekday.
Date <- lubridate::as_date(date_time)

# unclass(Date)
Weekday <- lubridate::wday(Date)
Weekday <- as.data.frame(Weekday)

# Récupérons kes colonnes Latide, Longitude et Base afin de créer notre nouveau tableau.
Lat <- uber$Lat
Lat <- as.data.frame(Lat)
Lon <- uber$Lon
Lon <- as.data.frame(Lon)
Base <- uber$Base
Base <- as.data.frame(Base)


## Créons une nouvelle base de Donnée
## (Ici nous allons utilser la fonction cbind pour concatener nos différentes variables)
## A remarquer que nous aurons pu utiliser rbind qui marcherait parfaite sans que l'on ai
```

```
## à convertir nos différents variables en dataframe or bind_cols nécessite des dataframes,
## d'où la convertion des différentes valeurs en dataframes avec la fonction 'as.data.frames'.
uber_final <- bind_cols(Lat,Lon,Base,Year,Month,Day,Weekday,Hour,Minute,Second)
```