MongoDB CW

Wednesday, 2 December, 2015 11:57

Data Inspection and Cleaning

Downloaded the data and did a first inspection

- Changed column names for simplification directly in the CSV
- Opened in R and loaded the dataset
  > mblog <- read.csv("C:/Users/miballe/Desktop/mblog/mblog.csv")

  1460535 observations of 6 variables. This looks good according to the specification.
- Summary

```
> summary(mblog)
         id              idmember                         timestamp                   text              geolat           geolng
#Quest14 :      15   Min.    :-2.147e+09                        :     247                 :     674   Min.    :49.42   Min.    :-7.6333
---      :      13   1st Qu.: 4.461e+07   2014-06-26 19:08:25:     122   ðŸ˜”          :     175   1st Qu.:51.50   1st Qu.:-2.3084
#Heroes  :       6   Median : 1.755e+08   2014-06-29 20:23:23:      49   Wow          :      92   Median :52.06   Median :-1.3607
#Quest14 :       5   Mean   : 2.569e+08   2014-06-22 23:53:12:      41   What a goal:      82   Mean    :52.47   Mean    :-1.4163
#Bobon   :       4   3rd Qu.: 3.671e+08   2014-06-26 19:08:26:      39   ðŸ˜´          :      81   3rd Qu.:53.43   3rd Qu.:-0.1733
#Heroes  :       4   Max.   : 1.979e+09   2014-06-22 23:53:11:      35   :(           :      74   Max.    :59.32   Max.    : 1.8607
(Other)  :1460488   NA's    :247          (Other)               :1460002   (Other)      :1459357   NA's    :1101    NA's    :1101
```

Missing Values

- Id
  - There are strings and other type of values not looking like an ID. As per inspection, and ID has 18 digits. Checking all rows not having that length.
    > badids <- mblog[(nchar(as.character(mblog$id)) != 18),]
  - Data doesn't seem to be recoverable. Text, geolon and geolng are empty for all fields. Timestamp, idmember and id contain invalid data. It looks like rows were shifted 3 rows to the left and existing data is not enough to recover missing values. Getting rid of all those values by creating a new variable gblog where filtered items will be stored from now and on.
    > gblog <- mblog[(nchar(as.character(mblog$id)) == 18),]
  - Even if IDs with length != 18 were removed, there may be other invalid items with that length, but not an ID. Filtered using the regular expression [0-9] to extract and inspect values containing characters different than numbers
    badids <- gblog[grep("[0-9]",gblog$id, invert=TRUE),]
    > view(badids)
  - Found 8 records with invalid data. Filtering them out.
    > gblog <- gblog[grep("[0-9]", gblog$id, invert=FALSE),]
    Gblog had 1459873 and then 1459865
  - When checking the summary of gblog, found that there are many reported duplicated IDs. Isolated for inspection all candidate duplicated rows. Found 217, removing all duplicates.
    > gblog <- gblog[!duplicated(gblog$id),]
    1459865 --> 1459648 --> OK
  - After a summary found that there are still ID values with some characters. Removed them by using a regular expression.
    > gblog <- gblog[grep("[A-z]", gblog$id, invert=TRUE),]
- Idmember

```
> summary(gblog$idmember)
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-2.147e+09  4.466e+07  1.755e+08  2.569e+08  3.672e+08  1.979e+09
```

  - There are negative values. A detailed inspection shows that those values seem to be valid, but positive.

```
> gblog$idmember <- ifelse(gblog$idmember > 0, gblog$idmember, gblog$idmember * -1)
> summary(gblog$idmember)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
8.550e+02 4.742e+07 1.852e+08 3.046e+08 3.839e+08 2.147e+09
```

  - After the substitution all member id's seem to be ok
- Timestamp
  - After verifying that there are no missing values, this field is converted to the POSIXlt data type in R and then check in the summary that the data is consistent.

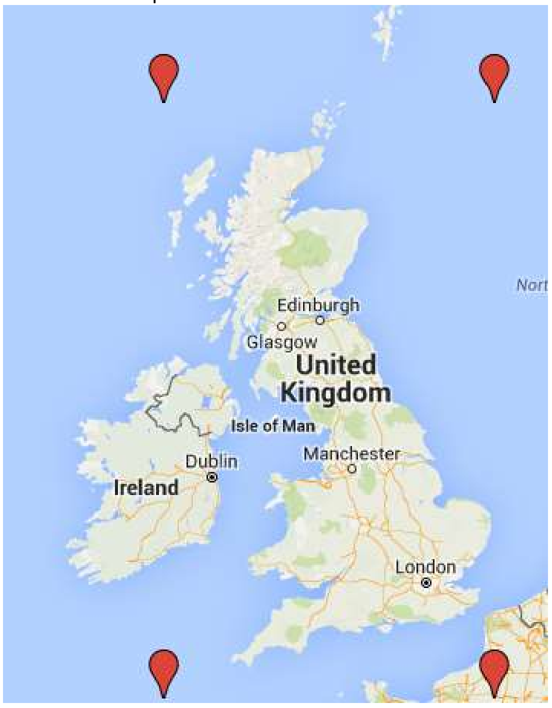
```
> gblog$timestamp <- as.POSIXlt(gblog$timestamp)
> summary(gblog)
                 id                idmember             timestamp                                    text                 geolat              geolng
 4808477014926458888:        1   Min.   :8.550e+02   Min.   :2014-06-22 23:00:00   ðŸ˜”               :    175   Min.   :49.42   Min.   :-7.6333
 4808477017065594888:        1   1st Qu.:4.742e+07   1st Qu.:2014-06-24 21:49:19   Wow               :     92   1st Qu.:51.50   1st Qu.:-2.3084
 480847702046310400:        1   Median :1.852e+08   Median :2014-06-26 21:55:54   What a goal       :     82   Median :52.06   Median :-1.3607
 480847702406991872:        1   Mean   :3.046e+08   Mean   :2014-06-27 01:21:38   ðŸ˜´               :     81   Mean   :52.47   Mean   :-1.4162
 480847703786913792:        1   3rd Qu.:3.839e+08   3rd Qu.:2014-06-28 21:58:51   :(                :     74   3rd Qu.:53.43   3rd Qu.:-0.1733
 480847704214761472:        1   Max.   :2.147e+09   Max.   :2014-06-30 21:59:59   ðŸ˜´ðŸ˜´ðŸ˜´:     64   Max.   :59.32   Max.   : 1.8607
 (Other)            :1459638                                                        (Other)           :1459076   NA's   :427    NA's   :427
```

- Text
  - In the text there's no particular rule that the field has to follow more than having at least 1 character. Checked this by selecting all items that have text length = 0
    > badset <- gblog[which(nchar(as.character(gblog$text)) == 0),]

The output is 0

- GeoLat & GeoLng
    - According to the summary, there are 427 entries without geographical location information. Inspected those values by filtering the empty ones and ensure that empty geolat are exactly the same as empty geolng. If so, those fields will be kept in the dataset, but ignored in MongoDB if the operation requires geo-location.
    - According to the summary, the max and min values for geolat and geolng are:
      Max(geolat): 59.32
      Min(geolat): 49.42
      Max(geolng): 1.8607
      Min(geolng):-7.6333

      Checked in a map and these are the limits:



If data is from UK only, it seems to be OK. Even if there's limit point in France, it doesn't necessarily means that such point is in the dataset. Just the max and min values.

The final dataset has 1,459,644 records. Original had 1,460,535.

Saved by running.
```
> write.csv(gblog, "gblog.csv")
```

MongoDB

To insert the data in MongoDB it's necessary to open a SSH session to the server, upload the file using FTP. (I used bitvise for this)

Switched to the path where the CSV file is located and executed:
```
$ mongoimport -d courseworks -c mblog --type csv --file ./gblog.csv --headerline
```

The database name is "courseworks", the collection name is "mblog".

This command detected 250 import errors due to non-valid UTF-8 characters. Reported 1,459,394 imported objects. It seems that there are lost 1,141 records. This will be cheked in more detail after finalizing the queries construction.

After the data is loaded, it's important to change the timestamp data type. For this, the following MongoDB query makes the trick.

```
var cursor = db.mblog.find()
while (cursor.hasNext()) {
    var doc = cursor.next();
    db.mblog.update({_id : doc._id}, {$set : {timestamp : new Date(doc.timestamp)}}, { multi: true })
}
```

It may report only one change, but it actually changed all entries and reported only one changed in the last loop execution.