01 - Panoramica Dati, Esplorazione e Statistiche riassuntive

Data Science case study for a Microsoft 213x Project NYC Taxi BigData Analysis
Lorenzo Negri, April 2018
Applications used: Microsoft R Open, Visual Studio, RevoScaleR libraries

Panoramica Dati

Ogni record (dati riga) nel file mostra un viaggio in taxi "giallo" a New York, con le seguenti variabili (dati colonna) registrate: *VendorID,* Un codice che indica il provider TPEP che ha fornito il record (1 = Creative Mobile Technologies, LLC; 2 = VeriFone Inc); *tpep_pickup_datetime* & *tpep_dropoff_datetime*, la data e l'ora in cui il/i passeggero/i sono stati prelevati e lasciati (in formato m/d/yyyy h:mm); *Passenger_count*, il numero di passeggeri per viaggio; *Trip_distance*, la distanza percorsa (in miglia come registrato dal tassametro); *Pickup_longitude* & *Pickup_latitude*, *Dropoff_longitude* & *Dropoff_ latitude*, la latitudine e la longitudine in cui i passeggeri sono stati prelevati e fatti scendere; le informazioni di pagamento quali il tipo di pagamento (*Payment_type*, 1=carta di credito, 2=contanti, 3=Nessun addebito, 4=Controversia, 5= sconosciuto, 6=scatto annullato) e il costo del viaggio (*Fare_amount*) suddiviso per tipologia di tariffa (*RateCodeID*, 1=tariffa standard, 2=JFK, 3=Newark, 4=Nassau o Westchester, 5=tariffa negoziata, 6=giro in gruppo); *Store_and_fwd_flag*, indicatore binario se il record di viaggio è stato trattenuto nella memoria del veicolo prima di inviarlo al venditore, ovvero "salva e inoltra", perché il veicolo non aveva una connessione al server (Y or N); *Extra*, extra e supplementi vari (include l'ora di punta +$ 0,50 e +$ 1 per la corsa notturna); MTA_tax, Tassa MTA di $ 0,50 che viene esclusa automaticamente quando la tipologia di cliente non è soggetta; *Improvement_surcharge*, una sovrattassa di miglioramento di $ 0,30 per la valutazione del viaggio; *Tip_amount*, la mancia: questo campo viene popolato automaticamente per le mance da carta di credito (le mance in contanti non sono incluse); *Tolls_amount*, importo totale di tutti i pedaggi pagati in viaggio; *Total_amount*, l'importo totale addebitato ai passeggeri (escluso le mance in contanti).

Il *dataset* è composto di un totale di circa settanta milioni di osservazioni (dati riga) con 19 variabili (dati colonna).

Esplorazione e Statistiche riassuntive

L'esplorazione iniziale dei dati avviene caricando i file in R. Per preparare l'ambiente di lavoro in modo da poter analizzare i dati, abbiamo bisogno di un set di third-party packages. Il pacchetto `RevoScaleR` è preinstallato con Microsoft R Server (MRS) installato in Visual Studio, poiché non può essere scaricato da CRAN, tutti gli altri pacchetti mostrati di seguito invece sono pacchetti di terze parti che possono essere scaricati e installati da CRAN utilizzando il comando `install.packages`.

Inoltre, mentre carichiamo i pacchetti con la linea di comando, definiamo già alcune opzioni per rendere successivamente più semplice la visualizzazione di dati o i risultati.

```
options(max.print = 1000, scipen = 999, width = 100)
library(RevoScaleR)
rxOptions(reportProgress = 1) # riduce la qtà di output che RevoScaleR produce
library(dplyr)
options(dplyr.print_max = 200)
options(dplyr.width = Inf) # mostra tutte le colonne di un oggetto tbl_df
library(stringr)
library(lubridate)
library(rgeos) # spatial package
library(sp) # spatial package
library(maptools) # spatial package
library(ggmap)
library(ggplot2)
library(gridExtra) # per mettere insieme i grafici
library(ggrepel) # evitare la sovrapposizione di testo nei grafici
library(tidyr)
library(seriation) # pacchetto per il riordino di una distance matrix
```

Carico adesso in R le prime 1000 righe dei dati utilizzando la funzione `read.table`. E per evitare conversioni di fattori non necessarie, avendo già esaminato le variabili, assegno le tipologie di classi alle colonne archiviandole in un oggetto chiamato `col_classes` che poi passeremo attraverso `read.table`.

```
col_classes <- c('VendorID' = "factor",
                 'tpep_pickup_datetime' = "character",
                 'tpep_dropoff_datetime' = "character",
                 'passenger_count' = "integer",
                 'trip_distance' = "numeric",
                 'pickup_longitude' = "numeric",
                 'pickup_latitude' = "numeric",
                 'RateCodeID' = "factor",
                 'store_and_fwd_flag' = "factor",
                 'dropoff_longitude' = "numeric",
                 'dropoff_latitude' = "numeric",
                 'payment_type' = "factor",
                 'fare_amount' = "numeric",
                 'extra' = "numeric",
                 'mta_tax' = "numeric",
                 'tip_amount' = "numeric",
                      'tolls_amount' = "numeric",
                      'improvement_surcharge' = "numeric",
                      'total_amount' = "numeric")
```

È buona norma caricare un piccolo campione di dati come `data.frame` in R su cui eseguire delle verifiche. Quando vogliamo applicare una funzione ai dati XDF, possiamo prima applicarla a `data.frame` dove è più facile e veloce individuare errori prima di applicarlo a tutto il dataset.

```
input_csv <- 'yellow_tripdata_2016-01.csv'
# prendiamo una parte dei dati e li carichiamo come data.frame (per fare test)
nyc_sample_df <- read.csv(input_csv, nrows = 1000, colClasses = col_classes)
head(nyc_sample_df)
```

```
VendorID tpep_pickup_datetime tpep_dropoff_datetime passenger_count trip_distance
1        2  2016-01-01 00:00:00   2016-01-01 00:00:00               2          1.10
2        2  2016-01-01 00:00:00   2016-01-01 00:00:00               5          4.90
3        2  2016-01-01 00:00:00   2016-01-01 00:00:00               1         10.54
4        2  2016-01-01 00:00:00   2016-01-01 00:00:00               1          4.75
5        2  2016-01-01 00:00:00   2016-01-01 00:00:00               3          1.76
6        2  2016-01-01 00:00:00   2016-01-01 00:18:30               2          5.52
  pickup_longitude pickup_latitude RatecodeID store_and_fwd_flag dropoff_longitude
1        -73.99037        40.73470          1                  N         -73.98184
2        -73.98078        40.72991          1                  N         -73.94447
3        -73.98455        40.67957          1                  N         -73.95027
4        -73.99347        40.71899          1                  N         -73.96224
5        -73.96062        40.78133          1                  N         -73.97726
6        -73.98012        40.74305          1                  N         -73.91349
  dropoff_latitude payment_type fare_amount extra mta_tax tip_amount tolls_amount
1         40.73241            2         7.5   0.5     0.5          0            0
2         40.71668            1        18.0   0.5     0.5          0            0
3         40.78893            1        33.0   0.5     0.5          0            0
4         40.65733            2        16.5   0.0     0.5          0            0
5         40.75851            2         8.0   0.0     0.5          0            0
6         40.76314            2        19.0   0.5     0.5          0            0
  improvement_surcharge total_amount
1                   0.3          8.8
2                   0.3         19.3
3                   0.3         34.3
4                   0.3         17.3
5                   0.3          8.8
6                   0.3         20.3
```

Possiamo quindi vedere le prime righe dei dati e sembra che tutto sia stato caricato correttamente.

Ora carico tutti i dati usando MRS. MRS ha due modi per gestire i file:

1. Può funzionare direttamente con i file, il che significa che può leggere e scrivere direttamente in file.
2. Può convertire i file in un formato chiamato XDF (XDF sta per *external data frame*).

Scelgo la seconda opzione. Per convertire file in XDF, usiamo la funzione `rxImport`. Ponendo `append="rows",` possiamo anche combinare più file in un singolo file XDF.

```
input_xdf <- 'yellow_tripdata_2016.xdf'
library(lubridate)
most_recent_date <- ymd("2016-07-01") # il giorno dei mesi è irrilevante

st <- Sys.time()
for(ii in 1:6) { # i dati di ogni mese aggiunti ai dati del primo mese
```

```
  file_date <- most_recent_date - months(ii)
  input_csv <- sprintf('yellow_tripsample_%s.csv', substr(file_date, 1, 7))
  append <- if (ii == 1) "none" else "rows"
  rxImport(input_csv, input_xdf, colClasses = col_classes, overwrite = TRUE, append = append)
  print(input_csv)
}
Sys.time() - st # memorizza il tempo necessario per importare i dati
```

```
Rows Processed: 10906858
[1] "yellow_tripdata_2016-01.csv"
Rows Processed: 11382049
[1] "yellow_tripdata_2016-02.csv"
Rows Processed: 12210952
[1] "yellow_tripdata_2016-03.csv"
Rows Processed: 11934338
[1] "yellow_tripdata_2016-04.csv"
Rows Processed: 11836853
[1] "yellow_tripdata_2016-05.csv"
Rows Processed: 11135470
[1] "yellow_tripdata_2016-06.csv"
```

Ho usato il pacchetto *lubridate* per semplificare in modo automatico il loop tra i file CSV.

Summary Statistics

Possiamo ora vedere un riepilogo delle statistiche utilizzando la funzione `rxSummary` con `nyc_xdf`. La funzione `rxSummary` si applica con la classica notazione utilizzata da molte funzioni R. In questo caso, per esempio la formula *~ fare_amount* significa che vogliamo vedere un riepilogo solo per quella colonna, aggiungendo con + altre variabili, possiamo aggiungere tutte le voci che ci interessano. Proprio come la funzione di riepilogo in R base, `rxSummary` ci mostrerà un output diverso a seconda del tipo di colonna. In questo caso vado ad inserire tutte le colonne numeriche che interessano e una categorica (*payment_type*).

```
input_xdf <- 'yellow_tripdata_2016.xdf'

nyc_xdf <- RxXdfData(input_xdf)

system.time(
  rxsum_xdf <- rxSummary(~passenger_count
                         + trip_distance
                         + fare_amount
                         + extra
                         + mta_tax
                         + tip_amount
                         + tolls_amount
                         + improvement_surcharge
                         + total_amount
                         + payment_type, nyc_xdf) # statistical summaries
)
rxsum_xdf
```

```
Rows Processed: 3467953
   user  system elapsed
   0.00    0.00    0.92
Call:
rxSummary(formula = ~passenger_count + trip_distance + fare_amount +
    extra + mta_tax + tip_amount + tolls_amount + improvement_surcharge +
    total_amount + payment_type, data = nyc_xdf)

Summary Statistics Results for: ~passenger_count + trip_distance + fare_amount +
    extra + mta_tax + tip_amount + tolls_amount + improvement_surcharge +
    total_amount + payment_type
Data: nyc_xdf (RxXdfData Data Source)
File name: yellow_tripdata_2016.xdf
Number of valid observations: 3467953

 Name                  Mean       StdDev       Min    Max         ValidObs MissingObs
 passenger_count     1.6602820    1.31044643    0.0        9.00 3467953  0
 trip_distance       6.5103520 6445.86584647    0.0 12000004.50 3467953  0
 fare_amount         12.8835511  11.39395462 -376.0     2550.20 3467953  0
 extra                0.3324486    0.43799020   -4.5       50.01 3467953  0
 mta_tax              0.4974139    0.03861298   -1.0        3.00 3467953  0
 tip_amount           1.8009263    2.64878399  -35.0      998.14 3467953  0
 tolls_amount         0.3159671    1.58133619  -10.5      613.50 3467953  0
 improvement_surcharge 0.2996687   0.01449831   -0.3       11.64 3467953  0
 total_amount         16.1304688  13.93443837 -376.3     2551.00 3467953  0

Category Counts for payment_type
Number of categories: 4
Number of valid observations: 3467953
Number of missing observations: 0

 payment_type Counts
 2            1148718
 1            2300794
 3              13696
 4               4745
```

Possiamo notare come vi siano degli errori nelle rilevazioni, ad esempio la distanza massima percorsa per una corsa taxi di milioni di miglia, oppure la tariffa di una corsa in dollari in negativo. Probabilmente sono operazioni effettuate manualmente dai taxisti, oppure errori del sistema di telemetria e GPS. Salta all'occhio anche la mancia massima di $ 998.14 e il costo del pedaggio totale per una corsa di $ 613.50.

I metodi di pagamento più utilizzati sono stati l=carta di credito e 2=contanti. Quasi l4000 di nessun addebito e quasi 5000 controversie in sei mesi di servizio taxi a New York.

1. Nel set di dati dei Taxi di New York, certi outlier posso essere imputati a:
2. Un passeggero potrebbe prendere un taxi e usarlo tutto il giorno per fare più commissioni, chiedendo all'autista di aspettarlo.
3. Un passeggero potrebbe voler dare 5 dollari in di mancia e l'autista premendo per sbaglio due volte 5, aggiunge 55 dollari a un viaggio che ne costa 40 dollari.
4. Un passeggero potrebbe litigare con un autista e andarsene senza pagare.
5. Viaggi con più passeggeri potrebbero avere una persona che paga per tutti o ognuno paga per se stesso, con alcuni che pagano con una carta e altri che usano denaro contante.
6. Un autista può accidentalmente mantenere il contatore in funzione dopo aver fatto scendere qualcuno.
7. Gli errori di registrazione della macchina possono comportare l'assenza di dati o di dati errati. In tutti questi casi un outlier potrebbe essere rumore per un'analisi e un punto di interesse per qualcos'altro.