Quantifying crowd size with mobile phote and Twitter data - Final Report

Ogi Moore and Connor Smith

12/5/2016

Introduction

We elected to replicate the findings of Federico Botta, Helen Susannah Moat, and Tobias Preis's paper on Quantifying crowd size with mobile phone and *Twitter* data. In the paper, they look at a number of soccer games with a known attendence and known phone, internet and twitter acitivity; and they evaluate the similar phone and internet and twitter acitivty in comparison to a number of flights over a several week period.

Data Import

The data is in very good shape, but we do need to tell `R` that the timestamps are in-fact times, and not just generic strings.

```
san_siero.attendees = read.csv('./data/Attendees_San_Siro.csv')
san_siero.phone_data = read.csv('./data/San_Siro_Mobile_Phone_Data.csv')
san_siero.twitter_data = read.csv('./data/San_Siro_Twitter_Data.csv')
linate.data = read.csv('./data/Linate_Data.csv')
linate.flight_schedule = read.csv('./data/Linate_Flights_Schedule.csv')

# Converting to dates
san_siero.phone_data$Timestamp = as.Date(strptime(san_siero.phone_data$Timestamp,
                                                  "%Y-%m-%d %H:%M:%S"))
san_siero.twitter_data$Timestamp = as.Date(san_siero.twitter_data$Timestamp)
san_siero.attendees$Date = as.Date(san_siero.attendees$Date)
```

The soccer game raw data is comprised of 3 seperate files, so we need to merge them together based on the relevant timestamps.

```
san_siero.daily_data <- aggregate(san_siero.phone_data$Calls.and.SMS.Activity,
                                  by=list(Category=san_siero.phone_data$Timestamp),
                                  FUN=max)
san_siero.daily_data <- rename(san_siero.daily_data, c("x"="Calls.and.SMS.Activity"))
san_siero.daily_data$Internet.Activity = aggregate(san_siero.phone_data$Internet.Activity,
                                         by=list(Cateogry=san_siero.phone_data$Timestamp),
                                         FUN=max)$x
san_siero.daily_data$Twitter.Activity = san_siero.twitter_data$Twitter.Activity
san_siero.daily_data <- rename(san_siero.daily_data, c("Category"="Date"))
soccer_data <- merge(san_siero.daily_data, san_siero.attendees,
                                        by="Date")

kable(head(soccer_data),
            format='pandoc',
            caption='Soccer Game Data',
            centering=TRUE)
```

Table 1: Soccer Game Data

Date	Calls.and.SMS.Activity	Internet.Activity	Twitter.Activity	Attendees.at.San.Siro
2013-11-02	180.050	104.640	85	44261
2013-11-09	97.693	100.350	51	39775
2013-11-15	222.520	137.080	117	49000
2013-11-23	79.276	77.290	73	34848
2013-12-01	102.930	106.180	78	43607
2013-12-04	88.803	44.783	29	12714

Soccer Games Dataset

The authors performed a linear regression comparing calls and SMS activity, Internet activity, *Twitter* activity to the number of attendees. With `R` we are able to perform the same linear regression analysis with ease.

```
lm_paper_results <- c(0.771, 0.937, 0.855)
attendees_v_phone <- lm(soccer_data$Attendees.at.San.Siro ~
                          soccer_data$Calls.and.SMS.Activity)
attendees_v_internet <- lm(soccer_data$Attendees.at.San.Siro ~
                             soccer_data$Internet.Activity)
attendees_v_twitter <- lm(soccer_data$Attendees.at.San.Siro ~
                            soccer_data$Twitter.Activity)


lm_duplication_results <- c(round(summary(attendees_v_phone)$adj.r.squared, 3),
                            round(summary(attendees_v_internet)$adj.r.squared, 3),
                            round(summary(attendees_v_twitter)$adj.r.squared, 3))

lm_results <- data.frame(lm_paper_results,
                         lm_duplication_results,
                         row.names=c('Calls and SMS Data',
                                     'Internet Activity',
                                     'Twitter Activity'))

kable(lm_results,
            format='pandoc',
            centering=TRUE,
            caption='Linear Regression R^2^ Values',
            col.names = c('Published Results', 'Duplication Results'))
```

Table 2: Linear Regression R^2 Values

	Published Results	Duplication Results
Calls and SMS Data	0.771	0.771
Internet Activity	0.937	0.937
Twitter Activity	0.855	0.855

We can see that our R^2 values match up exactly, and next we check their Spearman correlation values.

```
cor_paper_results <- c(0.927, 0.976, 0.924)
cor_duplication_results <- c(round(cor(soccer_data$Attendees.at.San.Siro,
```

```r
                                      soccer_data$Calls.and.SMS.Activity,
                                      method='spearman'), 3),
                            round(cor(soccer_data$Attendees.at.San.Siro,
                                      soccer_data$Internet.Activity,
                                      method='spearman'), 3),
                            round(cor(soccer_data$Attendees.at.San.Siro,
                                      soccer_data$Twitter.Activity,
                                      method='spearman'), 3))

cor_results <- data.frame(cor_paper_results,
                          cor_duplication_results,
                          row.names=c('Calls and SMS Data',
                                      'Internet Activity',
                                      'Twitter Activity'))

kable(cor_results,
            format='pandoc',
            caption='Spearman Correlation Values',
            col.names = c('Published Results', 'Duplication Results'))
```

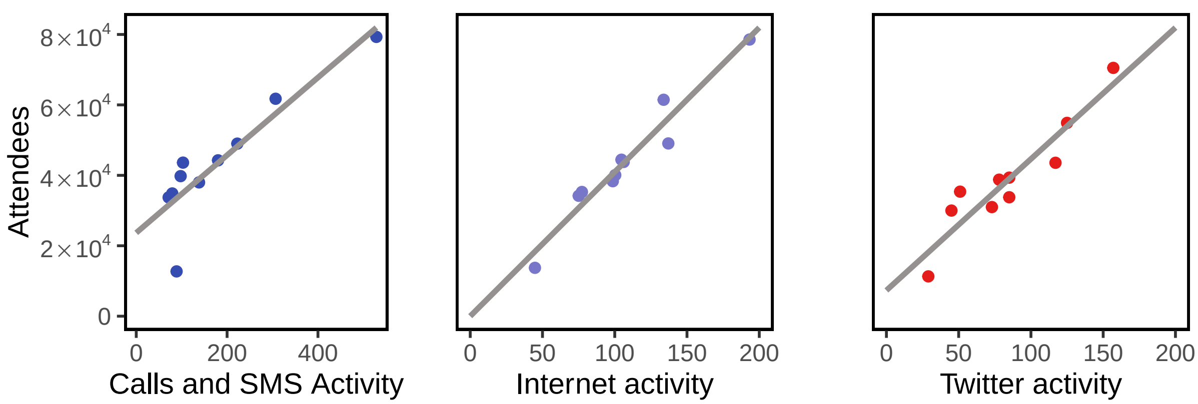
Table 3: Spearman Correlation Values

	Published Results	Duplication Results
Calls and SMS Data	0.927	0.927
Internet Activity	0.976	0.976
Twitter Activity	0.924	0.924

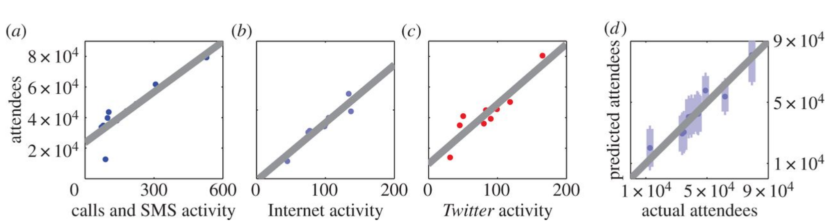
Our spearman correlation values match up precisely as well. Since we have come to the same conclusions, we can replicate their plot.

```r
# Thanks to: https://stackoverflow.com/questions/10762287/how-can-i-format-axis-labels-with-exponents-w
scientific_formatter <- function(x){
  output <- gsub("e", " %*% 10^", scientific_format()(x))
  output <- gsub("^+", "^", output, fixed=TRUE)
  output <- gsub("%*% 10^00", "", output, fixed=TRUE)
  formatted_output <- parse(text=output)
  return(formatted_output)
}

p1 <- ggplot(soccer_data, aes(Calls.and.SMS.Activity,
                              Attendees.at.San.Siro)) +
        geom_point(col='#354CB0') +
      scale_y_continuous(labels=scientific_formatter) +
      stat_smooth(method ='lm', se=FALSE, colour='#959190', fullrange=TRUE)  +
      expand_limits(x=0, y=0) +
      labs(x='Calls and SMS Activity', y='Attendees') +
      theme(panel.background = element_blank(),
            panel.border = element_rect(colour='black', fill=NA, size=1),
            panel.grid.major = element_blank())

p2 <- ggplot(soccer_data, aes(Internet.Activity,
                              Attendees.at.San.Siro)) +
        geom_point(colour='#7876C9') +
```

```
      stat_smooth(method ='lm', se=FALSE, colour='#959190', fullrange=TRUE)  +
      expand_limits(x=c(0, 200), y=0) +
      labs(x='Internet activity') +
      theme(aspect.ratio=1,
            panel.background = element_blank(),
            axis.text.y = element_blank(),
            axis.ticks.y = element_blank(),
            axis.title.y = element_blank(),
            panel.border = element_rect(colour='black', fill=NA, size=1))

p3 <- ggplot(soccer_data, aes(Twitter.Activity,
                              Attendees.at.San.Siro)) +
        geom_point(col='#E41D1A') +
      stat_smooth(method ='lm', se=FALSE, colour='#959190', fullrange=TRUE) +
      expand_limits(x=c(0, 200), y=0) +
      labs(x='Twitter activity') +
      theme(aspect.ratio=1,
            panel.background = element_blank(),
            axis.text.y = element_blank(),
            axis.ticks.y = element_blank(),
            axis.title.y = element_blank(),
            panel.border = element_rect(colour='black', fill=NA, size=1))

p1 <- ggplot_gtable(ggplot_build(p1))
p2 <- ggplot_gtable(ggplot_build(p2))
p3 <- ggplot_gtable(ggplot_build(p3))

maxHeight = unit.pmax(p1$heights[2:3], p2$heights[2:3], p3$heights[2:3])
p1$heights[2:3] <- maxHeight
p2$heights[2:3] <- maxHeight
grid.arrange(p1, p2, p3, ncol=3, nrow=1, respect=TRUE)
```



For comparison, here is the plot attached to the paper:



Airport Dataset

In the airport dataset the authors took a different method to approximating the crowd size. They approximated the number of people at the airport based on the number of flights to and from the airport. More specifically, they summed the flights departing in the two hour window following the time of interest and the number of incoming flights in the hour leading up to the time of interest. The raw data provides the number of flights arriving and departing the airport on an hour by hour basis over a 1 week period.

```
kable(head(linate.flight_schedule),
           format='pandoc',
           caption="Linate Flight Schedule Data",
           centering=TRUE)
```

Table 4: Linate Flight Schedule Data

Timestamp	Departures	Arrivals
2014-05-05 00:00:00	0	0
2014-05-05 01:00:00	0	0
2014-05-05 02:00:00	0	0
2014-05-05 03:00:00	0	0
2014-05-05 04:00:00	0	0
2014-05-05 05:00:00	0	0

The authors also provide a relative quantity of calls and SMS activity and internet activity, as well as Twitter activity

```
kable(head(linate.data),
           format='pandoc',
           caption='Linate Phone Data',
           centering=TRUE)
```

Table 5: Linate Phone Data

Timestamp	Calls.and.SMS.Activity	Internet.Activity	Twitter.Activity
2013-11-01 00:00:00	133.940	1599.8	0
2013-11-01 01:00:00	87.867	1247.0	0
2013-11-01 02:00:00	134.630	1210.1	0
2013-11-01 03:00:00	41.017	1159.6	0
2013-11-01 04:00:00	100.430	1575.1	2
2013-11-01 05:00:00	463.340	3730.6	0

The reader may notice here that the dates of the time-stamps do not match up (they are off by 6 months). The authors explain that the way they compensate for this is that they line up the days of the week from the flights data, and assume that the flight schedule remains fairly consistent week for week. They excluded November 1^{st}, 2^{nd}, and 3^{rd}, as well as December 30^{th} and 31^{st} as they were holidays.

As the authors decided to look at the number of incoming flights up to an hour before, and the number of departing flights for two hours following, this made for having to modify the raw data substantially. Furthermore, the authors then decided to average the calls and sms activity, internet activity and twitter activity associated with any given hour and weekday over the two month span. This kind of data wrangling is outside of our skill set in `R`, however we were able to make the modifications necessary in `Python`. Should a reviewer wish to rerun this python code, they will need the Pandas library installed. The `Python` code outputs a file titled 'Linate_wrangled.csv' which we will import into `R` to generate our statistics with.

```
import numpy as np
import pandas as pd
import datetime as dt
linate_sched_data = pd.read_csv('./data/Linate_Flights_Schedule.csv',
                                parse_dates=[0],
                                infer_datetime_format=True,
                                index_col=0)
linate_sched_data['Day'] = linate_sched_data.index.weekday_name
linate_sched_data['Hour'] = linate_sched_data.index.hour

linate_sched_data['Flights'] = np.roll(linate_sched_data['Departures'], -2) + \
                               np.roll(linate_sched_data['Departures'], -1) + \
                               np.roll(linate_sched_data['Arrivals'], 1)
linate_flight_data = linate_sched_data.groupby(['Day', 'Hour']).sum()
linate_flight_data.drop(['Arrivals', 'Departures'], inplace=True, axis=1)
linate_phone_data = pd.read_csv('./data/Linate_Data.csv',
                               parse_dates=[0],
                               infer_datetime_format=True)

days_to_skip = pd.to_datetime(['2013-11-01',
                               '2013-11-02',
                               '2013-11-03',
                               '2013-12-30',
                               '2013-12-31']).date
linate_phone_data = \
    linate_phone_data[linate_phone_data['Timestamp'].dt.date.isin(days_to_skip) == False]
linate_phone_data.set_index('Timestamp', drop=True, inplace=True)
linate_phone_data['Day'] = linate_phone_data.index.weekday_name
linate_phone_data['Hour'] = linate_phone_data.index.hour
linate_avg_phone_data = pd.DataFrame(linate_phone_data.groupby(['Day', 'Hour'],
                                                                sort=True).mean())
result = pd.concat([linate_flight_data, linate_avg_phone_data], axis=1)
result.to_csv('./data/Linate_wrangled.csv')
```

We can now import the wrangled CSV file that our python code generated and move on with our analysis.

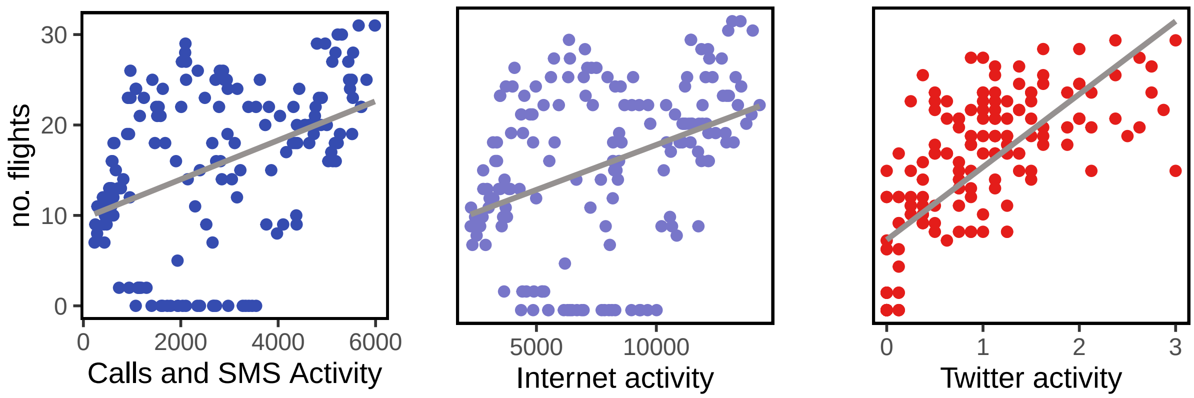
```
flight_data <- read.csv('./data/Linate_wrangled.csv')
# kable(head(linate_flight_data),
#           format='pandoc',
#           caption='Linate Flight Data Cleaned Up',
#           centering=TRUE)

lm_paper_results <- c(0.175, 0.143, 0.510)
flights_v_phone <- lm(flight_data$Flights ~
                      flight_data$Calls.and.SMS.Activity)
flights_v_internet <- lm(flight_data$Flights ~
                         flight_data$Internet.Activity)
flights_v_twitter <- lm(flight_data$Flights ~
                        flight_data$Twitter.Activity)



lm_duplication_results <- c(round(summary(flights_v_phone)$adj.r.squared, 3),
                            round(summary(flights_v_internet)$adj.r.squared, 3),
```

```
                             round(summary(flights_v_twitter)$adj.r.squared, 3))

lm_results <- data.frame(lm_paper_results,
                         lm_duplication_results,
                         row.names=c('Calls and SMS Data',
                                     'Internet Activity',
                                     'Twitter Activity'))

kable(lm_results,
            format='pandoc',
            centering=TRUE,
            caption='Linear Regression R^2^ Values',
            col.names = c('Published Results', 'Duplication Results'))
```

Table 6: Linear Regression R^2 Values

	Published Results	Duplication Results
Calls and SMS Data	0.175	0.175
Internet Activity	0.143	0.143
Twitter Activity	0.510	0.510

```
p1 <- ggplot(flight_data, aes(Calls.and.SMS.Activity, Flights)) +
      geom_point(col='#354CB0') +
      stat_smooth(method ='lm', se=FALSE, colour='#959190', fullrange=TRUE) +
      labs(x='Calls and SMS Activity', y='no. flights') +
      theme(aspect.ratio=1,
            panel.background = element_blank(),
            panel.border = element_rect(colour='black', fill=NA, size=1))

p2 <- ggplot(flight_data, aes(Internet.Activity, Flights)) +
      geom_point(colour='#7876C9') +
      labs(x='Internet activity') +
      stat_smooth(method ='lm', se=FALSE, colour='#959190', fullrange=TRUE)  +
      theme(aspect.ratio=1,
            panel.background = element_blank(),
            axis.text.y = element_blank(),
            axis.ticks.y = element_blank(),
            axis.title.y = element_blank(),
            panel.border = element_rect(colour='black', fill=NA, size=1))

p3 <- ggplot(flight_data, aes(Twitter.Activity, Flights)) +
      geom_point(col='#E41D1A') +
      labs(x='Twitter activity') +
      stat_smooth(method ='lm', se=FALSE, colour='#959190', fullrange=TRUE) +
      theme(aspect.ratio=1,
            panel.background = element_blank(),
            axis.text.y = element_blank(),
            axis.ticks.y = element_blank(),
            axis.title.y = element_blank(),
            panel.border = element_rect(colour='black', fill=NA, size=1))
grid.arrange(p1, p2, p3, ncol=3, nrow=1, respect=TRUE)
```



For comparison, here is the plot attached to the paper