Public Transport Journeys by Type of Transport in London

Filip Twardzik

April 5, 2018

1. About the dataset

This dataset comes from the Transport of London and is available here. It consists of aggregated journeys by mode of transport (Underground, Overground, Docklands Light Rail, Tram, Bus) given in millions and by time with granularity of 4 weeks periods. In following chapters I will take a closer look at these statistics in order to perform a basic descriptive analysis.

2. Understanding the structure

2.1 Load the data set and take a look at its structure to identify the most relevant columns.

```
# load the csv file from the URL
df <- read.csv(file="https://files.datapress.com/london/dataset/public-transport-journeys-type-transpor
# show the columns of the data set as a knitr table
kable( col.names = "Columns in the dataset", names(df) )
```

Columns in the dataset
Period.and.Financial.year
Reporting.Period
Days.in.period
Period.beginning
Period.ending
Bus.journeys..m.
Underground.journeys..m.
DLR.Journeys..m.
Tram.Journeys..m.
Overground.Journeys..m.
Emirates.Airline.Journeys..m.

```
# create a vector that contains the most relevant columns
cols = c("Period.beginning","Period.ending", "Bus.journeys..m.","DLR.Journeys..m.","Reporting.Period",
         "Tram.Journeys..m.","Overground.Journeys..m.", "Underground.journeys..m.")
# filter the data set in order for it to contain only the previously selected columns
df <- df[,cols[1:length(cols)]]
```

2.2 Preview of the first and last rows in the data set.

```
# show the first and last 5 rows from the data set
kable(head(df[,cols[1:4]], n=5))
```

Period.beginning	Period.ending	Bus.journeys..m.	DLR.Journeys..m.
01-Apr-10	01-May-10	189.1	6.3

Period.beginning	Period.ending	Bus.journeys..m.	DLR.Journeys..m.
02-May-10	29-May-10	181.6	5.8
30-May-10	26-Jun-10	175.9	5.8
27-Jun-10	24-Jul-10	183.4	6.1
25-Jul-10	21-Aug-10	160.4	5.8

```
kable(head(df[,cols[5:length(cols)]], n=5))
```

Reporting.Period	Tram.Journeys..m.	Overground.Journeys..m.	Underground.journeys..m.
1	2.3	NA	90.5
2	2.2	NA	84.5
3	2.1	NA	84.3
4	2.1	NA	86.5
5	2.0	NA	82.9

```
kable(tail(df[,cols[1:4]], n=5))
```

	Period.beginning	Period.ending	Bus.journeys..m.	DLR.Journeys..m.
100	12-Nov-17	09-Dec-17	183.5	9.8
101	10-Dec-17	06-Jan-18	144.8	7.4
102	07-Jan-18	03-Feb-18	175.0	9.3
103	04-Feb-18	03-Mar-18	168.3	9.3
104	04-Mar-18	31-Mar-18	NA	NA

```
kable(tail(df[,cols[5:length(cols)]], n=5))
```

	Reporting.Period	Tram.Journeys..m.	Overground.Journeys..m.	Underground.journeys..m.
100	9	2.4	15.5	115.1
101	10	1.9	11.6	87.7
102	11	2.2	14.8	105.5
103	12	2.2	14.1	105.3
104	13	NA	NA	NA

We can already identify some NA values for Overground in first periods as also for all modes of transport in the most recent period. First observation is explained directly by the information on the source website: *"Reliable Overground journey numbers have only been available since October 2010"*. Lack of data for the last period is explainable just by the fact that at the moment of writing this notebook the aggregated numbers were most probably just not yet available (compare the period: 04.03.2018 - 31.03.2018 with the publication date: 04.04.2018).

3. Preparing the data

3.1 Rename the columns

This step helps to achieve naming consistency and easier, more intuitive further use.

```
# redefine the column names vector
cols <- c("period_number","period_begin","period_end","bus_journeys","dlr_journeys",
          "overground_journeys","tram_journeys","underground_journeys")
# rename selected columns
names(df)[names(df)=="Reporting.Period"] <- "period_number"
names(df)[names(df)=="Period.beginning"] <- "period_begin"
names(df)[names(df)=="Period.ending"] <- "period_end"
names(df)[names(df)=="Bus.journeys..m."] <- "bus_journeys"
names(df)[names(df)=="DLR.Journeys..m."] <- "dlr_journeys"
names(df)[names(df)=="Tram.Journeys..m."] <- "tram_journeys"
names(df)[names(df)=="Overground.Journeys..m."] <- "overground_journeys"
names(df)[names(df)=="Underground.journeys..m."] <- "underground_journeys"
```

3.2 Slice off the last reporting period

Since it contains only NA values it is of no use for further analysis.

```
df <- head(df, n=-1)
```

3.3 Replace all remaining NA values with 0

It will help to avoid losing whole rows with missing values durin later analysis.

```
df <- replace_na(df, list("bus_journeys"=0, "dlr_journeys"=0, "tram_journeys"=0,
                          "overground_journeys"=0,"underground_journeys"=0 ))
```

3.4 Normalize date columns

Transform "period_begin" and "period_end" columns so that they follow one standard date format. Furthermore create new columns for extracted day, month and year from the date for later aggregation purposes. To achieve this write a helper function that transforms a date string into the desired format and three further functions for extraction of day, month and year from the date.

```
# function converting date format dd-mmm-yy to dd-mm-yyyy
normalize_date <- function( date ) {
  months <- c("Jan"="01","Feb"="02","Mar"="03","Apr"="04","May"="05","Jun"="06",
              "Jul"="07","Aug"="08","Sep"="09","Oct"="10","Nov"="11","Dec"="12")
  # retrieve month part of the date
  mon <- substr(date, 4,6)
  # replace month string with appriopiate numerical representation
  date <- str_replace(date,  mon, months[mon])
  # convert two digit  to full four digits year representation
  date <- paste( substr(date,1,6),"20",substr(date,7,8), sep="")
  return( date )
}
#extract year, month and day from a date
extract_day <- function( date ) return( as.numeric(substr(date,1,2)))
extract_month <-function( date ) return(as.numeric(substr(date,4,5)))
extract_year <- function( date ) return(as.numeric(substr(date,7,10)))

# normalize date
df["period_begin"] <- apply(df["period_begin"], 1, normalize_date )
df["period_end"] <- apply(df["period_end"], 1, normalize_date )
# create new columns for extracted time data as day,month,year
```

```
df["period_begin_day"] <- apply(df["period_begin"], 1, extract_day )
df["period_begin_month"] <- apply(df["period_begin"], 1, extract_month )
df["period_begin_year"] <- apply(df["period_begin"], 1, extract_year )
df["period_end_day"] <- apply(df["period_end"], 1, extract_day )
df["period_end_month"] <- apply(df["period_end"], 1, extract_month )
df["period_end_year"] <- apply(df["period_end"], 1, extract_year )
```

3.5 Create columns with calculated timestamp

Reason behind that is to represent date not only as a categorical value but also as a numerical quantity. To achieve this write a helper function that converts string representation of date to the unix timestamp.

```
# function calculating a timestamp for a given date
date_to_timestamp <- function( date ) {
  # convert modified date to numerical timestamp format
  return( as.numeric(as.POSIXct( date , format="%d-%m-%Y")) )
}
# create new columns for timestamps
df["period_begin_timestamp"] <- apply(df["period_begin"], 1, date_to_timestamp )
df["period_end_timestamp"] <- apply(df["period_end"], 1, date_to_timestamp )
```

3.6 Preview of the transformed data

```
# show first and last 5 rows of the tranformed data including newly added columns
kable(head(df[,cols[1:4]], n=5))
```

period_number	period_begin	period_end	bus_journeys
1	01-04-2010	01-05-2010	189.1
2	02-05-2010	29-05-2010	181.6
3	30-05-2010	26-06-2010	175.9
4	27-06-2010	24-07-2010	183.4
5	25-07-2010	21-08-2010	160.4

```
kable(head(df[,cols[5:length(cols)]],n=5))
```

dlr_journeys	overground_journeys	tram_journeys	underground_journeys
6.3	0	2.3	90.5
5.8	0	2.2	84.5
5.8	0	2.1	84.3
6.1	0	2.1	86.5
5.8	0	2.0	82.9

```
kable(head(df[,c("period_begin_timestamp","period_end_timestamp")]))
```

period_begin_timestamp	period_end_timestamp
1270072800	1272664800
1272751200	1275084000
1275170400	1277503200
1277589600	1279922400
1280008800	1282341600

period_begin_timestamp	period_end_timestamp
1282428000	1284760800

```
kable(head(df[,c("period_begin_day","period_begin_month","period_begin_year")]))
```

period_begin_day	period_begin_month	period_begin_year
1	4	2010
2	5	2010
30	5	2010
27	6	2010
25	7	2010
22	8	2010

```
kable(head(df[,c("period_end_day","period_end_month","period_end_year")]))
```

period_end_day	period_end_month	period_end_year
1	5	2010
29	5	2010
26	6	2010
24	7	2010
21	8	2010
18	9	2010

4. Descriptive Analysis

4.1 Summary statistics

To gain more insight in the data set perform summary statistics as the first step.

```
journey_cols <- c("dlr_journeys","underground_journeys","overground_journeys","tram_journeys","bus_jour
kable(summary(df[journey_cols]) )
```

dlr_journeys	underground_journeys	overground_journeys	tram_journeys	bus_journeys
Min. : 4.800	Min. : 72.50	Min. : 0.00	Min. :1.800	Min. :140.1
1st Qu.: 7.000	1st Qu.: 90.40	1st Qu.: 8.70	1st Qu.:2.100	1st Qu.:173.7
Median : 8.200	Median : 97.10	Median :10.60	Median :2.200	Median :180.3
Mean : 8.033	Mean : 97.63	Mean :10.17	Mean :2.243	Mean :178.5
3rd Qu.: 9.200	3rd Qu.:105.60	3rd Qu.:12.65	3rd Qu.:2.400	3rd Qu.:186.8
Max. :10.600	Max. :116.70	Max. :15.70	Max. :2.800	Max. :207.5

4.2 Number of journeys as a function of time

```
ggplot(df, aes(x = period_end_timestamp)) +
  geom_line(aes(y = dlr_journeys), colour = "#00A4A7") +
  geom_line(aes(y = underground_journeys), colour = "#E32017") +
  geom_line(aes(y = overground_journeys), colour = "#FF6600") +
  geom_line(aes(y = tram_journeys), colour = "#66CC00") +
  geom_line(aes(y = bus_journeys), colour = "#FF4040") +
```

```
  ylab(label="Journeys in millions") +
  xlab("Time periods (28 days) ")
```

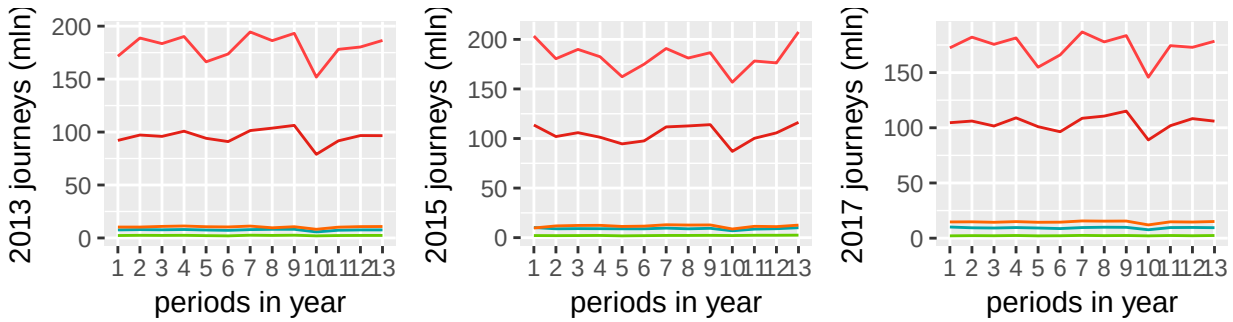
As we can clearly see, there are 8 distinctive minimums on the graph. Let's investigate them in that we take a closer look at shorter time periods: years 2013, 2015, 2017

```
# Journeys in 2013
g1 <- ggplot( data = df[ df["period_end_year"]==2013, ], aes(x = period_number)) +
  geom_line(aes(y = dlr_journeys), colour = "#00A4A7") +
  geom_line(aes(y = underground_journeys), colour = "#E32017") +
  geom_line(aes(y = overground_journeys), colour = "#FF6600") +
  geom_line(aes(y = tram_journeys), colour = "#66CC00") +
  geom_line(aes(y = bus_journeys), colour = "#FF4040") +
  ylab(label="2013 journeys (mln)") +
  scale_x_discrete(name ="periods in year", limits=c("1","2","3","4","5","6","7","8","9","10","11","12"

# Journeys in 2015
g2 <- ggplot(df[ df["period_end_year"]==2015, ], aes(x = period_number)) +
  geom_line(aes(y = dlr_journeys), colour = "#00A4A7") +
  geom_line(aes(y = underground_journeys), colour = "#E32017") +
  geom_line(aes(y = overground_journeys), colour = "#FF6600") +
  geom_line(aes(y = tram_journeys), colour = "#66CC00") +
  geom_line(aes(y = bus_journeys), colour = "#FF4040") +
  ylab(label="2015 journeys (mln)") +
  scale_x_discrete(name ="periods in year", limits=c("1","2","3","4","5","6","7","8","9","10","11","12"

# Journeys in 2017
```

```
g3 <- ggplot(df[ df["period_end_year"]==2017, ], aes(x = period_number)) +
  geom_line(aes(y = dlr_journeys), colour = "#00A4A7") +
  geom_line(aes(y = underground_journeys), colour = "#E32017") +
  geom_line(aes(y = overground_journeys), colour = "#FF6600") +
  geom_line(aes(y = tram_journeys), colour = "#66CC00") +
  geom_line(aes(y = bus_journeys), colour = "#FF4040") +
  ylab(label="2017 journeys (mln)") +
  scale_x_discrete(name ="periods in year", limits=c("1","2","3","4","5","6","7","8","9","10","11","12"
grid.arrange(g1, g2, g3,ncol = 3)
```



Eeach year TfL experiences a strong decrease in usage of the transport network during the 10th period in journeys on all transport modes, altough this behaviour is most visible on buses and on the Underground. This is the period spanning over Christmas holidays and New Year, which gives us direct explanation to this downhill on graph.

```
kable( df[ df$period_number==10,c("period_begin","period_end") ] )
```

	period_begin	period_end
10	12-12-2010	08-01-2011
23	11-12-2011	07-01-2012
36	09-12-2012	05-01-2013
49	08-12-2013	04-01-2014
62	07-12-2014	03-01-2015
75	13-12-2015	09-01-2016
88	11-12-2016	07-01-2017
101	10-12-2017	06-01-2018

4.3 Studying and comparison of the relative usage of transport modes

What is also visible on the graph, is the great difference between the number of journeys by Undeground and Buses in comparion with other transport modes (Overground, DLR, Tram). Wa cannot draw any conclusions about the importance of the mode though, because all subnetworks vary strongly in the length and area they cover which results in different amounts of kilometers they make each period.

4.3.1 Total kilometers operated by each mode of transport.

Numbers are given in millions and come from the annual TfL report for season 2015/2016.

```
total_km_operated <- data.frame( "transport_mode"=c("underground","overground","dlr","tram","bus"),
                        "km_operated"=c(82.5,10.5,6.0,3.0,492))
kable(total_km_operated)
```

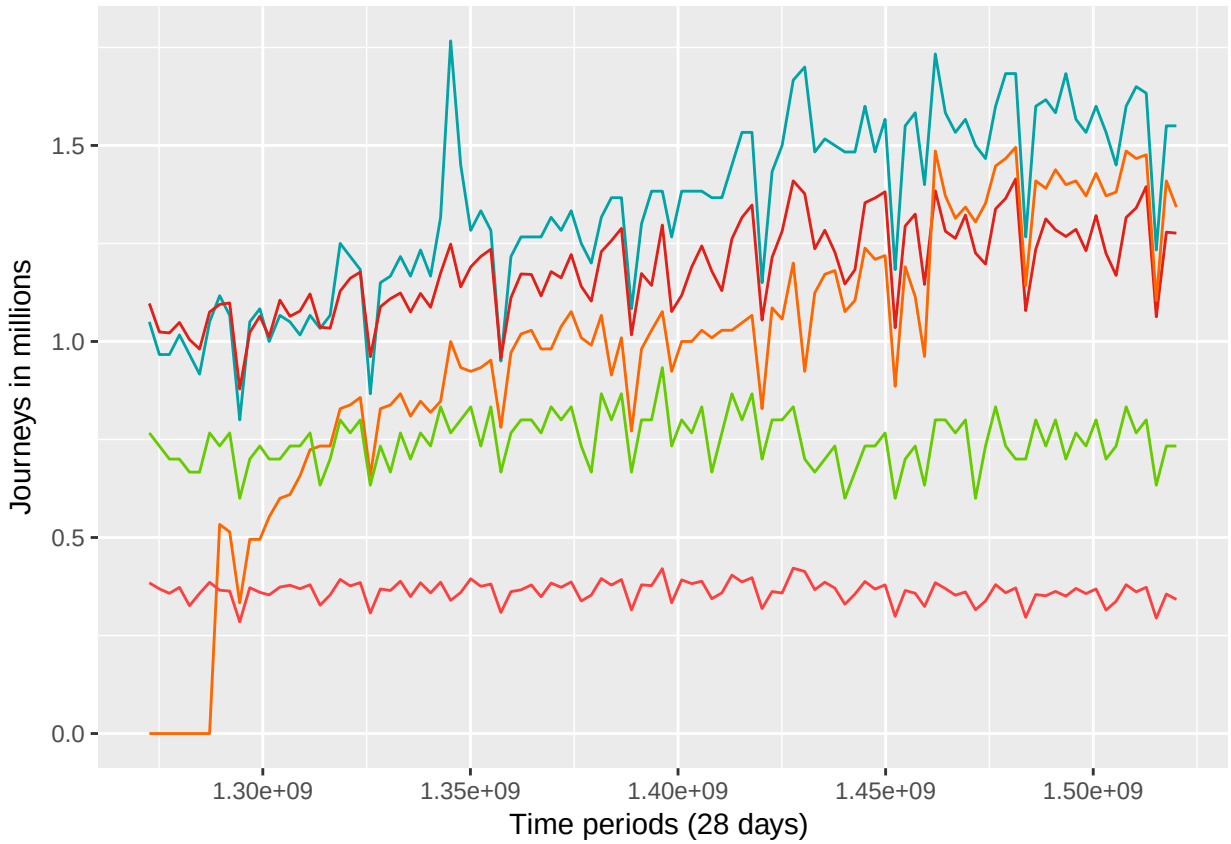
transport_mode	km_operated
underground	82.5
overground	10.5
dlr	6.0
tram	3.0
bus	492.0

4.3.2 Number of journeys divided by the amount of kilometers operated annualy.

```
dfn <- data.frame(df)
dfn["underground_journeys"] <-dfn["underground_journeys"] /
                              total_km_operated[ total_km_operated$transport_mode=="underground", "km_ope
dfn["overground_journeys"] <-dfn["overground_journeys"] /
                              total_km_operated[ total_km_operated$transport_mode=="overground", "km_oper
dfn["dlr_journeys"] <-dfn["dlr_journeys"] /
                              total_km_operated[ total_km_operated$transport_mode=="dlr", "km_operated"]
dfn["tram_journeys"] <-dfn["tram_journeys"] /
                              total_km_operated[ total_km_operated$transport_mode=="tram", "km_operated"]
dfn["bus_journeys"] <-dfn["bus_journeys"] /
                              total_km_operated[ total_km_operated$transport_mode=="bus", "km_operated"]
```

After considering the amount of total kilometers operated annualy by each mode of transport the modified, relative numbers of passengers in consecutive years show us fairly different results than the absolute values plotted earlier.

```
ggplot(dfn, aes(x = period_end_timestamp)) +
  geom_line(aes(y = dlr_journeys), colour = "#00A4A7") +
  geom_line(aes(y = underground_journeys), colour = "#E32017") +
  geom_line(aes(y = overground_journeys), colour = "#FF6600") +
  geom_line(aes(y = tram_journeys), colour = "#66CC00") +
  geom_line(aes(y = bus_journeys), colour = "#FF4040") +
  ylab(label="Journeys in millions") +
  xlab("Time periods (28 days) ")
```


Journeys in millions
1.5
1.0
0.5
0.0
1.30e+09
1.35e+09
1.40e+09
1.45e+09
1.50e+09
Time periods (28 days)