

Efficient Exploratory Data Analysis

using `data.table`, `ggplot2`, `lattice`

Install and load R packages

```
# install R packages
pkgs <- c("data.table", "ggplot2", "lattice", "bikeshare14")
install.packages(pkgs)

# check version of packages
packageVersion("data.table")
sapply(pkgs, packageVersion, simplify = FALSE)

# load packages
require("data.table")
sapply(pkgs, require, character.only = TRUE)
```

Investigate dataset

```
head(batrips)
```

```
?batrips
```

```
str(batrips)
```

```
summary(batrips)
```

```
boxplot(batrips$duration/60, log="y", ylab="duration (min) -  
log scale")
```

```
# proceed with batrips as data.frame/data.table objects
```

```
DF <- batrips
```

```
DT <- as.data.table(batrips)
```

Cleaning data

```
# investigate trips longer than 12 hours
# data.frame way
DF[DF$duration > 60*60*12, ]
# data.table way
DT[duration > 60*60*12]

# exclude trips longer than 12h
DF <- DF[DF$duration <= 60*60*12, ]
DT <- DT[duration <= 60*60*12]
```

DATA TABLES



- think in terms of basic units — rows, columns and groups
- data.table syntax provides *placeholder* for each of them

General form: DT[i, j, by]

On which rows

What to do?

Grouped by
what?

Trips by subscription_type

```
# count trips
table(DF$subscription_type)
aggregate(duration ~ subscription_type, data=DF, FUN=length)
DT[, length(duration), subscription_type]
DT[, .N, subscription_type]
# mean trips duration
aggregate(duration ~ subscription_type, data=DF, FUN=mean)
DT[, mean(duration), subscription_type]

# count, mean and sum
DT[, .(count=.N, mean_s=mean(duration),
sum_d=sum(duration)/(60*60*24)), subscription_type]
```

Plotting using ggplot2 and lattice

ggplot2 package implements "The Grammar of Graphics" designed by [Leland Wilkinson](#). ggplot2 uses plus sign "+" to combine new elements to existing plot. It is also capable to store plot in a variable and render graphic when print method is invoked on a variable.

lattice package provide powerful multi-panel plots "trellis" using R formula interface commonly used in R: $y \sim x \mid z$

Trips by subscription_type and hour of the day

```
# count by hour and subscription type
histogram(duration ~ hour(start_date) | subscription_type,
data=DT)

# median duration in minutes
ans <- DT[, .(median_m=median(duration)/60),
.(subscription_type, start_hour=hour(start_date))]

xyplot(median_m ~ start_hour | subscription_type, data=ans,
main="median trip duration")

ggplot(ans, aes(start_hour, median_m)) + geom_point() +
facet_wrap(~ subscription_type)
```


Common trip routes

```
ans <- DT[, .N, .(start_station, end_station)]
# order by count in descending order, take top 10
ans[order(-N)][1:10]
# by subscription_type, top 5
ans <- DT[, .N, .(subscription_type, start_station,
end_station)]
ans[order(-N), head(.SD, 5), subscription_type]
# by weekday, top 2
DT[, weekday := weekdays(start_date)]
ans <- DT[, .N, .(weekday, start_station, end_station)]
ans[order(-N), head(.SD, 2), .(weekday)]
```

Trips by weekday and subscription_type

```
# set order for weekday factor
days_in_week <- DT[order(wday(start_date)), unique(weekday)]
DT[, weekday := factor(weekday, levels=days_in_week)]
histogram(duration ~ weekday | subscription_type, data=DT,
main="trip count")

# median duration
ans <- DT[, .(.N, duration=median(duration)/60),
.(subscription_type, weekday)]
barchart(duration ~ weekday | subscription_type, data=ans,
ylab="duration (min)")
```

Import US holidays data by pasting into `fread`

```
HD <- fread("
date,holiday
2014-01-01,New Year Day
2014-01-20,Martin Luther King Jr. Day
2014-02-17,Presidents Day (Washingtons Birthday)
2014-05-26,Memorial Day
2014-07-04,Independence Day
2014-09-01,Labor Day
2014-10-13,Columbus Day
2014-11-11,Veterans Day
2014-11-27,Thanksgiving Day
2014-12-25,Christmas Day
")
```

Investigate holidays impact - merge datasets

```
head(HD)
str(HD)
str(DT)
DT[, date := as.Date(start_date, tz="America/Los_Angeles")]
HD[, date := as.Date(date, tz="America/Los_Angeles")]
# add holiday column from HD to DT
DT[HD, holiday := i.holiday, on = "date"]
head(DT)
# feature engineering
DT[, holiday := factor(ifelse(!is.na(holiday), "holiday",
"non-holiday"))]
```

Investigate holidays impact

```
# for non-english weekdays set locale
# Sys.setlocale("LC_TIME", "English")
DT[, workday := factor(ifelse(weekday %in%
c("Saturday", "Sunday"), "weekend", "workday"))]
ans <- DT[, .(N_trips=.N, N_days=uniqueN(date)),
.(subscription_type, workday, holiday)]
ans[, trips_day := N_trips/N_days]
ans[order(subscription_type, workday, holiday)]
barchart(~ trips_day | subscription_type, groups=holiday,
data=ans, auto.key=TRUE, subset=workday=="workday",
origin=0, scales=list(x="free"), main="mean trips per
workday (Mon-Fri)")
```

Ultimate goal of EDA: feature engineering

Feature engineering allows data scientist to prepare data to be better processed by machine learning algorithms in automatic manner to build better models.

Adding "holidays" to our dataset we allow machine learning algos to more precisely predict our measures - "trips per day" in last example.

Not having "holiday" flag in our dataset is not just a matter not being able to use holiday information to predict trips count. Holidays not flagged as holidays in the data will actually generate a "noise" due to higher variation. This happens because automated algos will cluster dates into Mon-Fri or Sat-Sun so holidays will be an outliers within those cluster reducing model precision.

Homework: read about and play with

- read data.table homepage r-datatable.com and [introduction vignette](#)
- know when and how to use `.N` and `.SD` special symbols
- read ggplot2 homepage ggplot2.tidyverse.org
- read lattice [introduction vignette](#)
- understand `order` and `sort` functions
- read about "feature engineering"

Questions?

Jan Gorecki: github.com/jangorecki

Contact: **jan61ji@gmail.com**