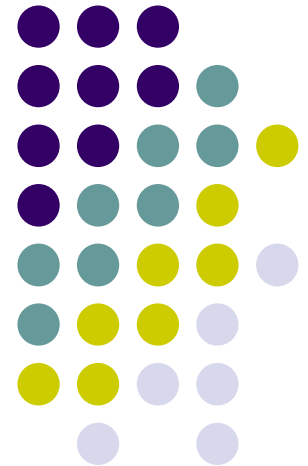


Big Data Analytics

Lecture 1 (Lab part)



Our Course Map



First Step

Define the corpus, acquire & convert documents, choose the unit of analysis

Preprocess

Statistical summaries

Scaling/
Scoring

Supervised
(wordscores)

Unsupervised
(wordfish & co.)

Goal

Second Step

Classification

Known
categories
(supervised)

Automatic tagging
(ontological
dictionaries)

Human
tagging

individual
classification

Classifiers (SVM, Random
Forest, Naive Bayes, etc)

aggregated
estimation

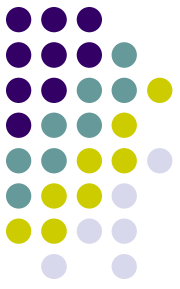
ReadMe - iSA

Unknown
categories
(unsupervised)

Single
Membership
Models

Mixed Membership Models
(LDA, Structural Topic
Model)

Quanteda approach

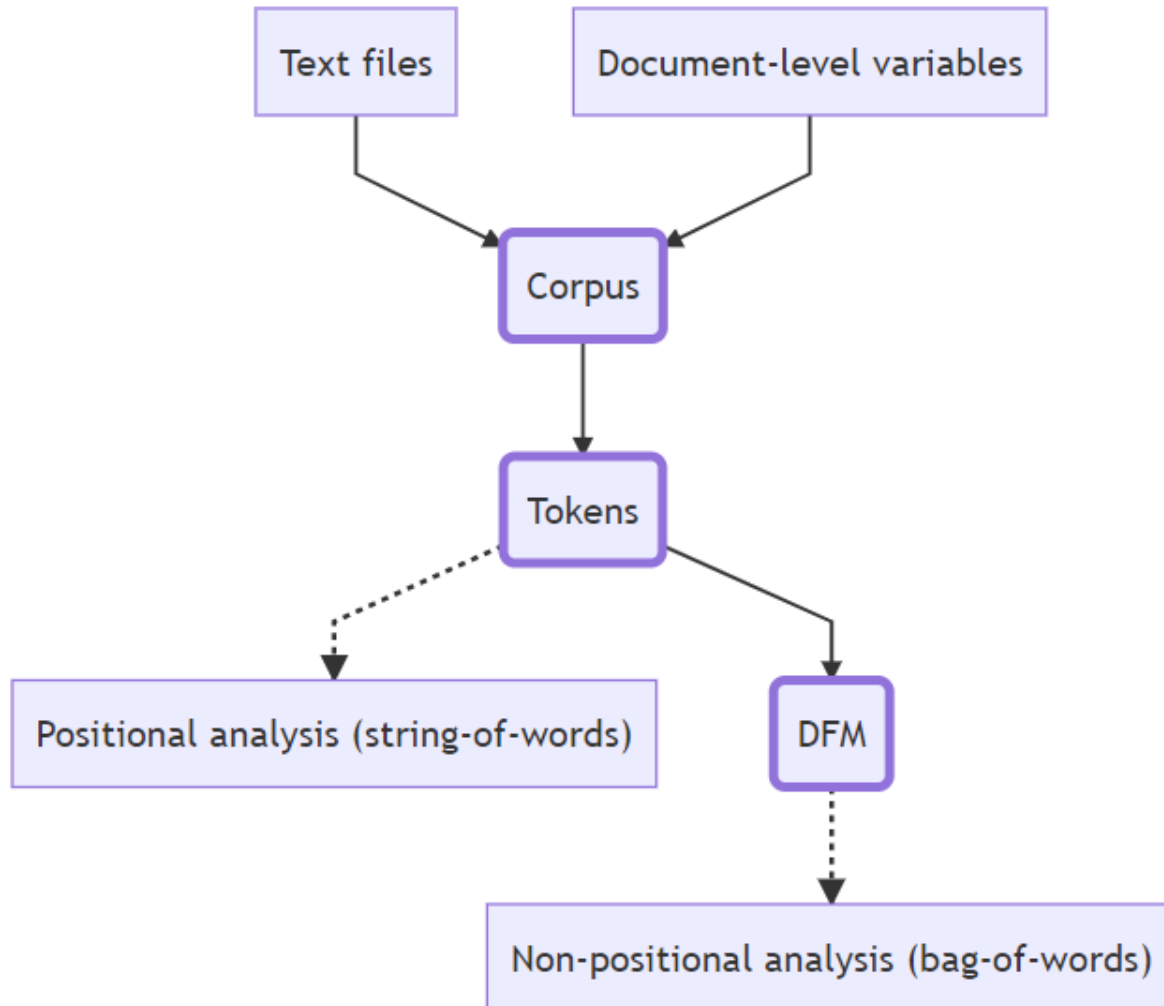
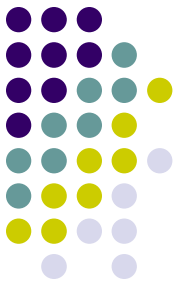


Quanteda package has three basic types of objects:

- *Corpus*: it saves character strings and variables in a data frame, by also combining texts with document-level variables (where available)
- *Tokens*: it stores tokens in a list of vectors (in a more efficient way than character strings), while still preserving positions of words. At this stage, you can apply pre-processing
- *Document-feature matrix (dfm)*: it represents frequencies of features in documents in a matrix. By doing it, it does not preserve information on positions of words within each text

Text analysis with Quanteda via bag-of-words goes always through all those three types of objects either explicitly or implicitly

Quanteda approach



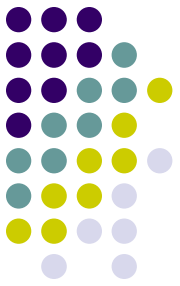
Statistical summaries



Once you have your tokens object and/or DfM, you can run some statistical summaries

Statistical summary methods are essentially quantitative summaries of texts to describe their characteristics on some indicator, and may use (or not) statistical methods based on sampling theory for comparison

Statistical summaries (1)



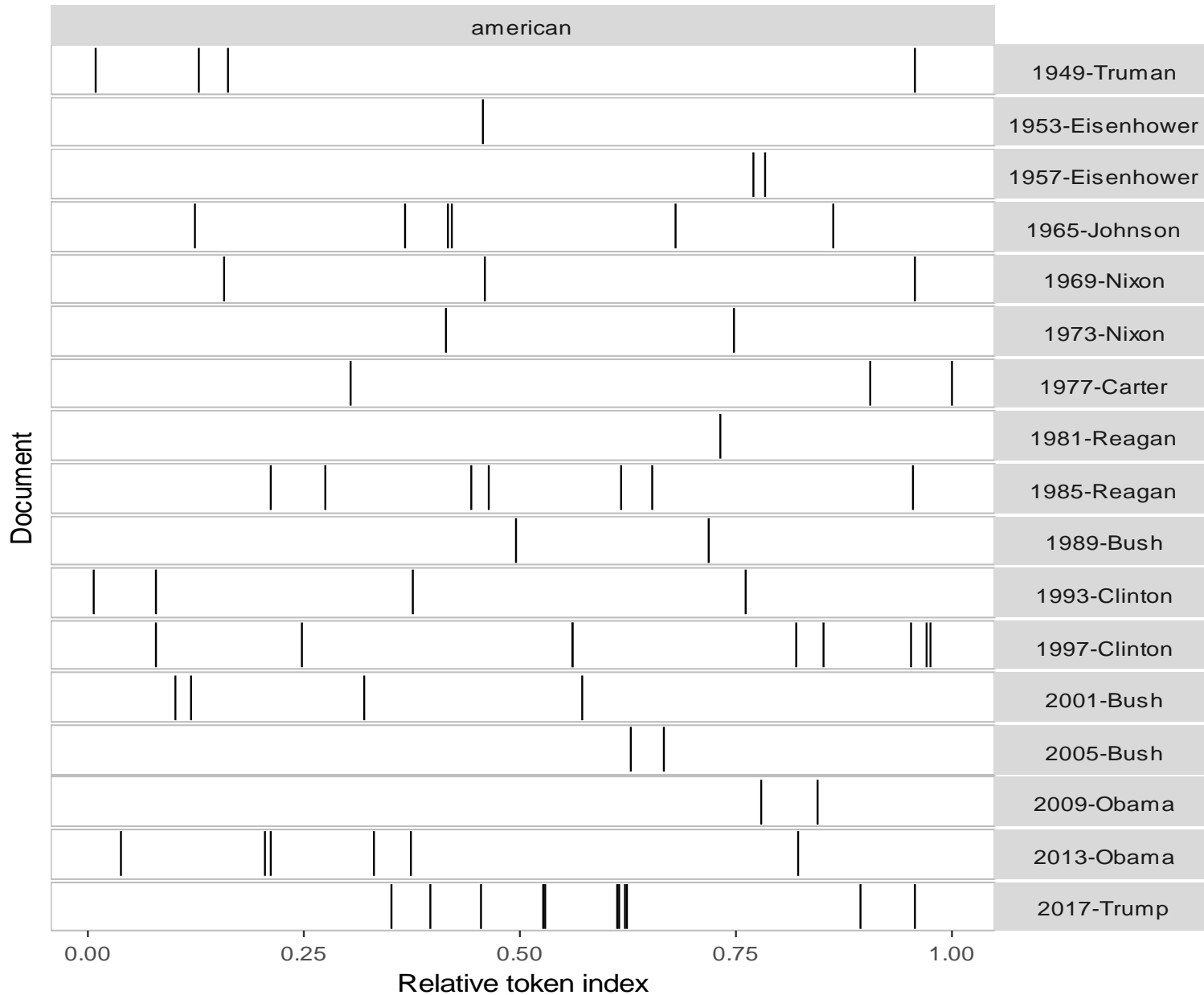
Descriptive statistics can be produced by running a *positional analysis* on the tokens object

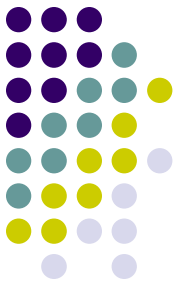
By retaining the original text sequence, for example, you can detect both **the relative frequency of an employed word across documents as well as the “timing” of that word** via a *Lexical dispersion plot*

Inaugural Speeches by US Presidents



Lexical dispersion plot

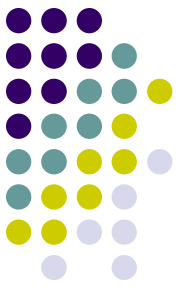




Statistical summaries (1)

Another possible positional analysis is a feature co-occurrence matrix (FCM) that allows you to record the number of co-occurrences of tokens

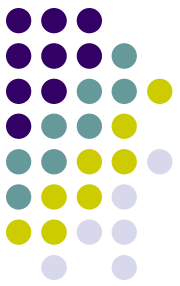
As we will see later on, this is crucial for running a word-embedding approach



Statistical summaries (2)

If we move to a DfM, the simplest *non-positional analysis* allows you to identify the most commonly occurring words, and summarize these as frequency distributions

For example: **tag clouds!** A tag cloud is a visual representation of text data, in which tags are single words whose frequency is shown with different font size (and/or color)



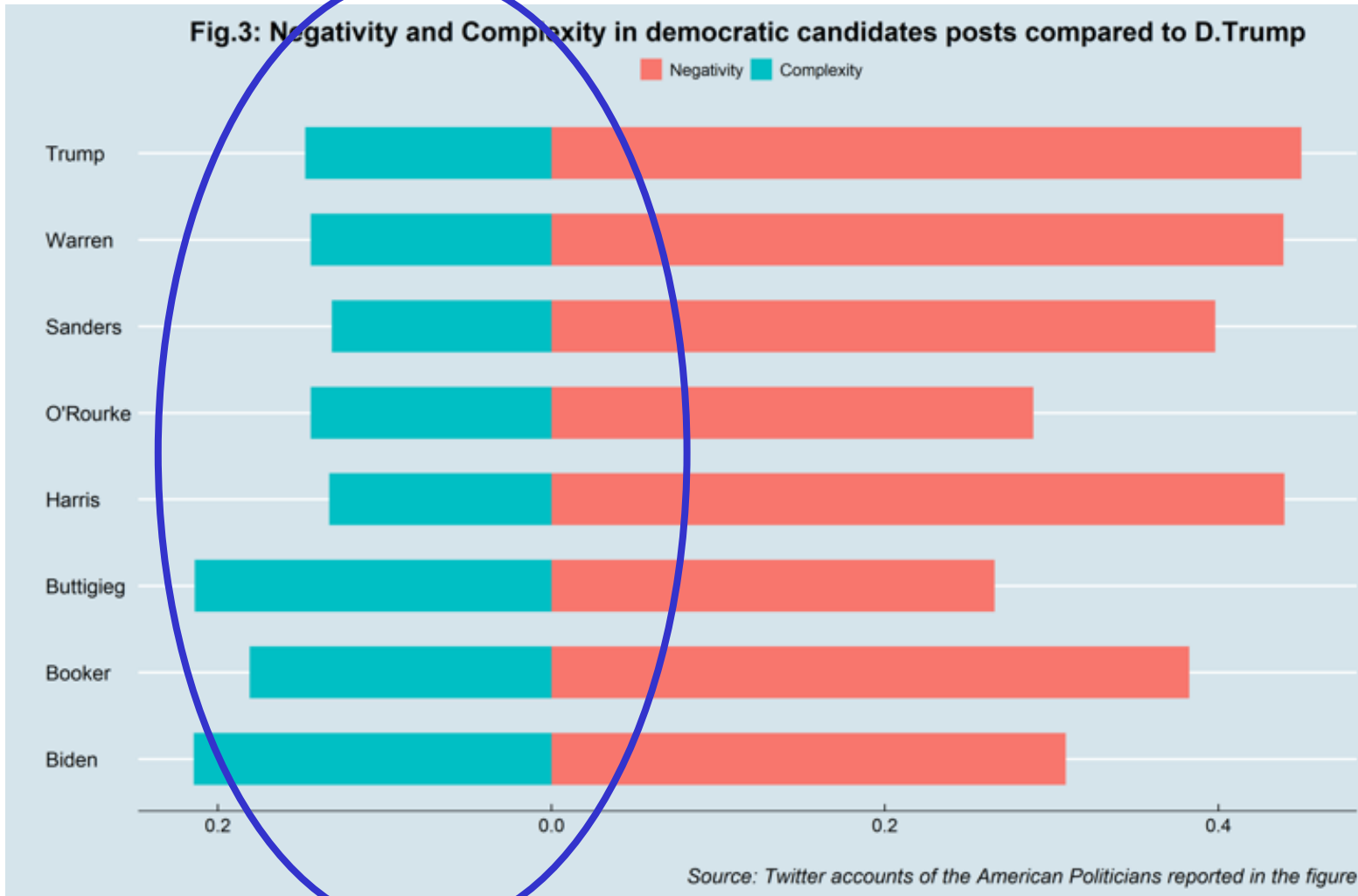
Statistical summaries (3)

Other quantitative summary measures of documents are designed to characterize specific qualities of texts

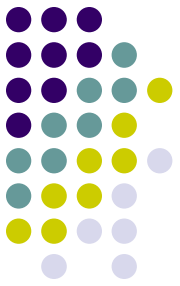
Comparing the **rates of types and tokens** forms the foundation for measures of **lexical diversity** (the rate of vocabulary usage), with most common such measure comparing the number of types to the number of tokens (the “type-token ratio”)

For example, it is argued that populist communication means simplified political discourse (lower diversity), in an attempt to reach the public more easily

So different, yet so alike (to Donald Trump?)

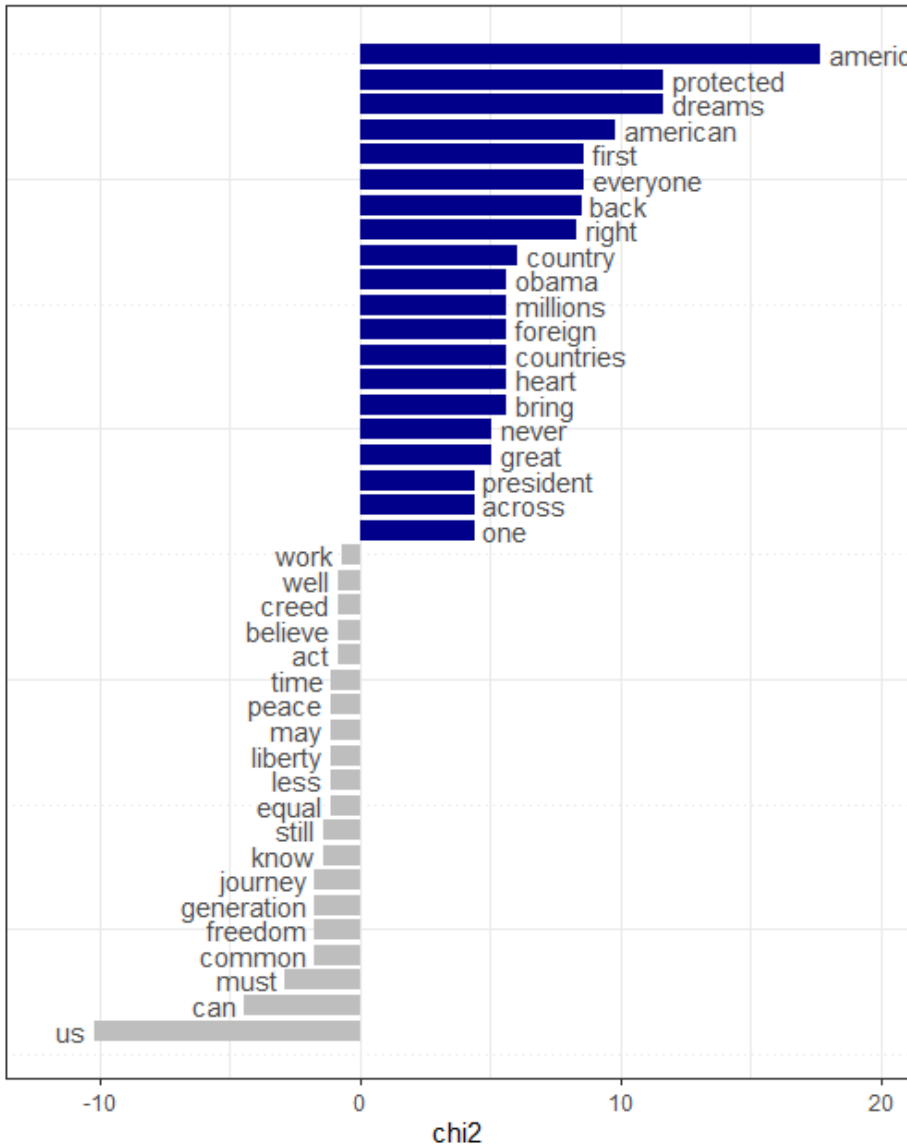
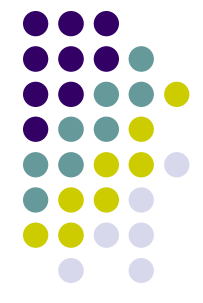


Statistical summaries (4)

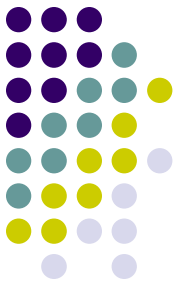


More sophisticated methods **compare the differential occurrences of words across texts** or partitions of a corpus, using statistical association measures, to identify the **words that belong primarily to sub-groups** such as those predominantly associated with male- versus female - authored documents, or Democratic versus Republican speeches

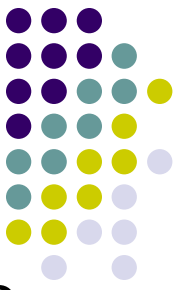
Comparing Obama vs. Trump Presidential inauguration speech



■ 2017-Trump
■ reference



IMPORTANT!!!



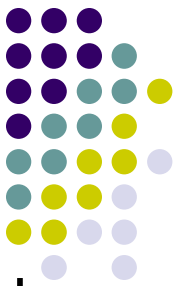
Before using rtweet

We will use since the next week the `rtweet` package: so start to install it!

```
install.packages("rtweet", repos='http://cran.us.r-project.org')
```

```
install.packages("httpuv", repos='http://cran.us.r-project.org')
```

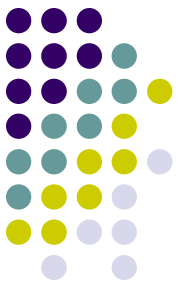
```
install.packages("ggmap", repos='http://cran.us.r-project.org')
```



Before using rtweet

Then open an R session and type the following commands. Plz let me know if you are able (or not) to download the 10 tweets:

```
library(rtweet)
library(httpuv)
rt <- search_tweets( "#rstats", n = 10,
include_rts = FALSE)
print(rt$text[1:10])
```



Optional

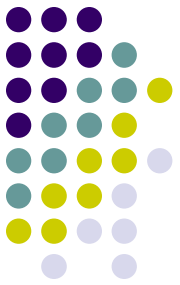
Before we can start geocoding data, we need to obtain an [API key from Google](#). Go to the registration page, and [follow the instructions](#) (select all mapping options)

The **geocoding API** is a free service, but you nevertheless need to associate a credit card with the account.

Please note that the Google Maps API is not a free service. There is a free allowance of 40,000 calls to the geocoding API per month, and beyond that calls are \$0.005 each

This implies that basically you have a monthly free limit of \$200 (more than enough...)

To register you need to have: a) a gmail account; b) a credit card



Optional

After you finish the registration (if everything hopefully works fine!) Google gives you back an API number. Save it!

Then type:

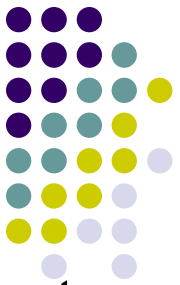
```
library(ggmap)
register_google(key = "NUMBER OF YOUR GOOGLE API!")
geocode(c("White House", "Uluru"))
```

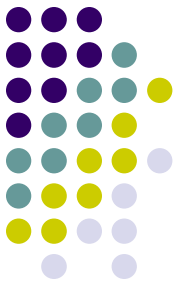
You should get this result back:

```
# A tibble: 2 x 2
  lon   lat
  <dbl> <dbl>
1 -77.0  38.9
2 131.  -25.3
```

Optional

If you are able to get the Google API, but GGMAP does not get any results back, enable the “geocoding app” in your console developer. Check how to enable GOOGLE API [here](#)





**Please check that
everything is ok with
rtweet before **next class!****