BAN432
Applied Textual Data Analysis for Business and Finance

Collecting textual data I:
Application Programming Interfaces (APIs)

Christian Langerfeld and Maximilian Rohrer

September 7, 2022

Packages and files you need in today's lecture

- package `jsonlite`
- package `rtweet` (if you have a Twitter account)
- package `xml2`
- package `rvest`
- package `tidytext`

from the `Files` section on Canvas:

- `data_for_lecture_3_v2022.Rdata`

Structure of the course

1	Introduction to course & basic R	Introduction
2	Introduction to R, specific to textual analysis	
3	Collecting textual data: APIs	Collecting data
4	Collecting textual data: EDGAR	
5	Preprocessing and cleaning, part I	Preprocessing data
6	Preprocessing and cleaning, part II	
7	Guest Lecture: Gisle Andersen (NHH)	
8	Regex-based application, Geography	Analyses
9	Regex-based application, Keyword in Context	
10	Automatic text summarization	
11	Sentiment: Twitter & Critical understanding	
12	Sentiment: Finance application	
13	Doc-Clustering: Cosine similarity & k-means	
14	Doc-clustering: Topic models	
15	Doc-Clustering: Multinominal Inverse Regression	
16	Guest Lecture: Vegard Larsen (Norges Bank)	
17	Contemporaneous papers in Finance	
18	Recap	

Plan for this lecture:

- Text data access
  - pull
  - push
- Text retrieval
  - structured/unstructured data
- APIs
  - Introduction
  - Examples
    - EDGAR
    - Wikipedia
    - Twitter

Text data access – introduction

- the human analyst makes a decision which textual data is used in the analysis
  - preselection: only the relevant data to solve a particular problem are used
  - selecting relevant data is the basic task
- two modes for data retrieval: *pull* and *push*

Text data access – pull and push mode

- pull:
  - the user initiates the access process by *querying* or *browsing*
    - querying: find relevant information using a keyword search in a search engine
    - browsing: traversing through the Internet following hyperlinks
- push:
  - the system initiates the process
  - a service on the Internet sends (pushes) text data to the user as they become available
  - monitor text data
  - examples:
    - academic journals send mail alerts to researchers when a new issue is published
    - media analysis: an organisation receives news articles when it is mentioned in the media (e.g. Retriever in Norway)

Text retrieval

Three kinds of (textual) data:

(1) unstructured data
(2) semi structured data
(3) structured data

Text retrieval – unstructured data (example: Apple 10-K)

- most of the text data is unstructured
- difficult to extract the important information
- where to look in the data set (and what to look for)?

```
PART I

Item 1.

Business
Company Background
The Company designs, manufactures and markets mobile communication and media devices and personal
computers, and sells a variety of related software, services, accessories, networking solutions
and third-party digital content and applications. The Company's products and services include
iPhone®, iPad®, Mac®, Apple Watch®, Apple TV®, a portfolio of consumer and professional software
applications, iOS, macOS®, watchOS® and tvOS operating systems, iCloud®, Apple Pay® and a variety
of accessory, service and support offerings. The Company sells and delivers digital content and
applications through the iTunes Store®, App Store®, Mac App Store, TV App Store, iBooks Store®
and Apple Music® (collectively "Digital Content and Services"). The Company sells its products
worldwide through its retail stores, online stores and direct sales force, as well as through
third-party cellular network carriers, wholesalers, retailers and value-added resellers. In
addition, the Company sells a variety of third-party Apple-compatible products, including
application software and various accessories through its retail and online stores. The Company
sells to consumers, small and mid-sized businesses and education, enterprise and government
customers. The Company's fiscal year is the 52 or 53-week period that ends on the last Saturday
of September. The Company is a California corporation established in 1977.

Business Strategy
The Company is committed to bringing the best user experience to its customers through its
innovative hardware, software and services. The Company's business strategy leverages its unique
ability to design and develop its own operating systems, hardware, application software and [...]
```

Text retrieval – semi-structured data (EDGAR filing)

```
<SEC-DOCUMENT>0000320193-17-000070.txt : 20171103
<SEC-HEADER>0000320193-17-000070.hdr.sgml : 20171103
<ACCEPTANCE-DATETIME>20171103080137
ACCESSION NUMBER:       0000320193-17-000070
CONFORMED SUBMISSION TYPE:  10-K
PUBLIC DOCUMENT COUNT:      97
CONFORMED PERIOD OF REPORT: 20170930
FILED AS OF DATE:       20171103
DATE AS OF CHANGE:      20171103

FILER:

    COMPANY DATA:
        COMPANY CONFORMED NAME:         APPLE INC
        CENTRAL INDEX KEY:          0000320193
        STANDARD INDUSTRIAL CLASSIFICATION: ELECTRONIC COMPUTERS [3571]
        IRS NUMBER:             942404110
        STATE OF INCORPORATION:         CA
        FISCAL YEAR END:            0930

    FILING VALUES:
        FORM TYPE:      10-K
        SEC ACT:        1934 Act
        SEC FILE NUMBER:    001-36743
        FILM NUMBER:        171174673
    [...]
</SEC-HEADER>
<DOCUMENT>
          Here goes the document
</DOCUMENT>
```

link to page

Text retrieval – structured data

```
<response>
  <result>
    <totalrows>1</totalrows>
    <rows>
      <row>
      <rownum>1</rownum>
      <values>
        <value field="cik">0000320193</value>
        <value field="companyname">APPLE INC</value>
        <value field="entityid">2035</value>
        <value field="primaryexchange">Nasdaq Global Market</value>
        <value field="marketoperator">NASDAQ</value>
        <value field="markettier">NASDAQ Global Select Market</value>
        <value field="primarysymbol">AAPL</value>
        <value field="siccode">3571</value>
        <value field="sicdescription">Electronic Computers</value>
      </values>
      </row>
    </rows>
  </result>
</response>
```

API – introduction

- retrieving structured data with APIs
- short for **A**pplication **P**rogramming **I**nterface
- an API allows a program to perform an action on a web service (instead of a human using a Graphical User Interface)
- many services on the Internet provide interfaces which allow the user to let their code interact with a service
- example: Google's APIs

API – introduction (cont.)

- APIs are located at the server side
- the basic procedure for working with APIs in R:
  - (1) inside your code a URL is generated that contains commands to be executed on the server (usually a query)
  - (2) your code sends the URL to the server (`GET` or `POST`)
    - (21) if you use the `GET` method, the server sends back an answer: a file with the results is returned (often .json or .xml)
    - (22) if you use the `POST` method, the server just tells you if the operation was successful
  - (3) if a file is returned, it has to be processed in R
- APIs are designed by humans and differ a lot when it comes to the requirements for the URL-format
- Usually a key is required to interact with an API

API – introduction (cont.)

Task:

- search for `NHH` on Google.
  - How does the URL look like on the results page?
  - Can you detect a structure?

API – introduction (cont.)

- `https://www.google.no/` (URL)
- `search?` (endpoint)
- `q=NHH` (query, contains search string)

. . . and a lot of other parameters

- The basic structure of the Google search-URL is:

`{URL} {endpoint?} {parameters&}`

API – introduction (cont.)

Three APIs we will look at now:

(1) Wikipedia
(2) Edgar
(3) Twitter

1. Wikipedia API

API – Wikipedia

- MediaWiki API is free and you can use it without an authentication key

Sample query that extracts the text of the page:

https://en.wikipedia.org/w/api.php?action=query&titles=Enron%20scandal&prop=extracts&format=json

These are the parts of the URL:

- `https://en.wikipedia.org/w/api.php` (the endpoint)
- `action=query` (the action)
  - `titles=Enron scandal` (action-specific parameter)
  - `prop=extracts` (action-specific parameter)
- `format=json` (the format)
- `?` and `&` (delimiters)

API – Wikipedia (cont.)

- There are (very!) many features in the MediaWiki API. You find the main documentation page here: https://www.mediawiki.org/wiki/API:Main_page
- How can we get the data into *R*?
  (1) Construct a search URL in R
  (2) Tell *R* to use the URL to request and retrieve the results of the query.

API – Wikipedia (cont.)

(1) construct a search URL in *R* to get the contributors to the "Enron scandal" page:

```
# define the parts of the URL
endpoint        <- "https://en.wikipedia.org/w/api.php"
type.of.action  <- "query"
page.title      <- "Enron%20scandal" # space has to be escaped
                                     # with '%20'
property        <- "contributors"    # or try "extracts"
response.format <- "json"

# combine the elements of the URL
search.url <- paste0(endpoint,"?action=",type.of.action,
                     "&titles=",page.title,"&prop=",
                     property,"&format=",response.format)

# inspect
search.url
```

API – Wikipedia (cont.)

(2) *R* has to send a request using the search URL

```
# Send the search url to the Wiki API and process
# the json file that is returned
require(jsonlite)
wiki.data <- fromJSON(search.url)
```

API – Wikipedia (cont.)

```
# inspect the structure of the wiki.data object
str(wiki.data)
```

```
## List of 2
##  $ continue:List of 2
##   ..$ pccontinue: chr "11954274|307"
##   ..$ continue  : chr "||"
##  $ query   :List of 1
##   ..$ pages:List of 1
##   .. ..$ 11954274:List of 5
##   .. .. ..$ pageid          : int 11954274
##   .. .. ..$ ns              : int 0
##   .. .. ..$ title           : chr "Enron scandal"
##   .. .. ..$ anoncontributors: int 474
##   .. .. ..$ contributors    :'data.frame':   10 obs. of  2 variables
##   .. .. .. ..$ userid: int [1:10] 1921264 194203 13791031 22041646 2
##   .. .. .. ..$ name  : chr [1:10] "Swpb" "Graham87" "Frietjes" "Nark
```

API – Wikipedia (cont.)

```
# We get the user names of the contributors in the data frame
# 'contributors' inside wiki.data.
contributors <- wiki.data$query$pages$`11954274`$contributors

contributors
```

```
##       userid                 name
## 1    1921264                 Swpb
## 2     194203              Graham87
## 3   13791031              Frietjes
## 4   22041646          Narky Blert
## 5   26021349                   MB
## 6   27015025 InternetArchiveBot
## 7    7611264            AnomieBOT
## 8    2790592          KylieTastic
## 9     279219              RussBot
## 10 10289486  Trappist the monk
```

2. EDAGR API

API – Edgar

- EDGAR: Electronic Data Gathering, Analysis, and Retrieval system at the U.S. Securities and Exchange Commission
- all companies in the U.S. have to file mandatory documents to EDGAR, like annual reports and news releases
- there are different APIs available to retrieve data from EDGAR
- see the documentation

API – Edgar (cont.)

- we assume that you want to retrieve the submission history of Apple
- from the documentation we see that the submissions API can be accessed under: `https://data.sec.gov/submissions/CIK##########.json`
- `##########` is the entity's 10-digit Central Index Key (CIK), including leading zeros
- Apple's CIK is: 0000320193

API – Edgar

Screen shot from Firefox of the json-file that is returned:

JSON Raw Data Headers

Save Copy Collapse All Expand All (slow) Filter JSON


```
cik:                                "320193"
entityType:                         "operating"
sic:                                "3571"
sicDescription:                     "Electronic Computers"
insiderTransactionForOwnerExists:   0
insiderTransactionForIssuerExists:  1
name:                               "Apple Inc."
tickers:
  0:                                "AAPL"
exchanges:
  0:                                "Nasdaq"
ein:                                "942404110"
description:                        ""
website:                            ""
investorWebsite:                    ""
category:                           "Large accelerated filer"
fiscalYearEnd:                      "0925"
stateOfIncorporation:               "CA"
stateOfIncorporationDescription:    "CA"
addresses:
  mailing:
    street1:                        "ONE APPLE PARK WAY"
    street2:                        null
    city:                           "CUPERTINO"
    stateOrCountry:                 "CA"
    zipCode:                        "95014"
```

4. Twitter API

API – Twitter

- dedicated packages in *R* that communicate with the APIs of popular websites, e.g. `rtweet`
- if you want to run the code on the next slide, you need a Twitter account

API – rtweet package

- the rtweet package is discussed later on in the course
- today just a short presentation
- we use the search_tweets function from the package rtweet to retrieve tweets from the Twitter API containing the string #lufthansa:

```
require(rtweet)

# Set some variables before running the script
query             <- "#lufthansa" # Put your query string here.
number.of.tweets  <- 500          # Max. number is 18000.

# This is the query. Only results from the past 7 days are returned
# (limit in the Twitter-API)
query.results     <- search_tweets(q = query,
                                   n = number.of.tweets)
```

API – rtweet package (cont.)

- the object returned from the `search_tweets` function is a data frame:

```
class(query.results)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

API – rtweet package (cont.)

- the data frame contains a lot of information
- let us have a look at the column names of the data frame

```
colnames(query.results)
```

```
 [1] "created_at"                    "id"
 [3] "id_str"                        "full_text"
 [5] "truncated"                     "display_text_range"
 [7] "entities"                      "metadata"
 [9] "source"                        "in_reply_to_status_id"
[11] "in_reply_to_status_id_str"     "in_reply_to_user_id"
[13] "in_reply_to_user_id_str"       "in_reply_to_screen_name"
[15] "geo"                           "coordinates"
[17] "place"                         "contributors"
[19] "is_quote_status"               "retweet_count"
[21] "favorite_count"                "favorited"
[23] "retweeted"                     "possibly_sensitive"
[25] "lang"                          "retweeted_status"
[27] "quoted_status"                 "text"
[29] "favorited_by"                  "scopes"
[31] "display_text_width"            "quoted_status_id"
[33] "quoted_status_id_str"          "quoted_status_permalink"
[35] "quote_count"                   "timestamp_ms"
[37] "reply_count"                   "filter_level"
[39] "query"                         "withheld_scope"
[41] "withheld_copyright"            "withheld_in_countries"
[43] "possibly_sensitive_appealable"
```

API – rtweet package (cont.)

Task 1: Which are the most frequently used apps for posting the tweets? You find them in the column `source`

Task 2: What is the time range in which the tweets were posted?

Task 3: Get tweets that refer to the Apple stock.

Summary

(1) data access types: ‘push’ and ‘pull’
(2) structured/unstructured data
(3) APIs
    (31) URL-format
(4) Examples:
    (41) Wikipedia: retrieve pages and contributors to the pages
    (42) EDGAR: retrieve data based on a company’s CIK
    (43) Twitter: retrieve data from Twitter with the rtweet package