Introduction

Taylor Arnold[1] Lauren Tilton[2]

04 July 2017

[1] Assistant Professor of Statistics
University of Richmond
@statsmaths

[2] Assistant Professor of Digital Humanities
University of Richmond
@nolauren

Goals

Our goal is to introduce concepts and tools for doing exploratory and predictive modelling from textual sources. We will be focusing on natural language processing.

Outline

- Preliminaries
- Topic I: Tokenising text
- Topic II: The NLP Pipeline
- Topic III: Modelling Textual Data
- Conclusions

About us

Taylor Arnold

- studies the application and computational challenges of using natural language and image processing to analyze large datasets
- intersection of statistics, computer science, linguistics, and the computational humanities

Lauren Tilton

- scholar of 20th century U.S. visual culture
- applies computational techniques to answer humanities questions

Joint Work

- *Humanities Data in R* – (2015) Quantitative Methods in the Humanities and Social Sciences, Springer
- *Photogrammar* – Digital project of historical photographs from the 1930s and 1940s: https://photogrammar.org
- *Photogrammar* – Digital project applying computational techniques to analyze moving image culture at scale: https://www.distanttv.org/

Working through this tutorial

All of the materials we are showing today are available as a GitHub repository. We encourage following along on your laptop as we run through the code snippets, or later at your own pace.

To do so, clone the GitHub repository from here:

https://github.com/statsmaths/useR2017_nlp.

Each of the sections to the talk are presented as individual R markdown files. All of the data is included in the repository and any required packages required can be downloaded from CRAN.

Coding style

Today we will make heavy use of the coding style adapted from

- the piping paradigm of the **magrittr** package
- *table verbs* from the package **dplyr**
- graphics from **ggplot2**

If you are not familiar with these, the code may seem a bit strange at first but we feel that it is the clearest way to present the material and very much simplifies the code.

Pipes

As a way of a very brief tutorial, the pipe operator %>% simply passes the output of one expression to the first element of the next:

```
1:10 %>%
  length()
```

```
## [1] 10
```

Or

```
matrix(1:12, ncol = 4) %>%
  dim()
```

```
## [1] 3 4
```

dplyr verbs: filter

Table verbs always take a data frame as a first argument (and therefore work well with pipes) and generally map into common database operations.

Specifically, `filter` accepts logical conditions and returns only those rows where the result is `TRUE`. For example, here we filter on the party of

```
presidential %>%
  filter(party == "Democratic")
```

```
## # A tibble: 5 x 4
##       name      start        end      party
##      <chr>     <date>     <date>      <chr>
## 1 Kennedy 1961-01-20 1963-11-22 Democratic
## 2 Johnson 1963-11-22 1969-01-20 Democratic
## 3  Carter 1977-01-20 1981-01-20 Democratic
## 4 Clinton 1993-01-20 2001-01-20 Democratic
## 5   Obama 2009-01-20 2017-01-20 Democratic
```

dplyr verbs: mutate and count

Likewise, `mutate` adds new columns to the data frame, and `count` aggregates the number of occurrences of a selected set of variables. For example, to count the parties of the presidents in the `presidents` dataset:

```
presidential %>%
  count(party, sort = TRUE)
```

```
## # A tibble: 2 x 2
##        party     n
##        <chr> <int>
## 1 Republican     6
## 2 Democratic     5
```

cleanNLP

Our main tool for text processing today will be our package **cleanNLP**. It is available on CRAN and is described in the forthcoming paper to be published in The R Journal:

> *Taylor Arnold (2017). A Tidy Data Model for Natural Language Processing using cleanNLP. The R Journal, 9(2), 1-20. URL https://journal.r-project.org/archive/2017/RJ-2017-035/index.html.*

A more formal analysis of the internals of the package will be covered in the talk on Wednesday at 14:06 in Plenary 4.0 as part of the *Text Mining* session.