Lab-12

your name here

2024-04-17

Preface

The goal of this assignment is to help you gain more familiarity with processing text data. As always, please come to office hours and reach out to your teaching staff if you have any questions.

Data

We will work with data on data scientist job postings in the U.S. scraped from popular job boards by JobSpikr.

```r
job_posts <- read_csv("data_scientist_united_states_job_postings.csv") |>
  select(-cursor, -contains("contact"), -uniq_id, -html_job_description) |>
  relocate(crawl_timestamp, url, .after = last_col())
job_posts |>
  head(5)
```

```
# A tibble: 5 x 17
  job_title              category company_name city  state country inferred_city
  <chr>                  <chr>    <chr>        <chr> <chr> <chr>   <chr>
1 Enterprise Data Scien~ Account~ Farmers Ins~ Wood~ CA    Usa     Woodland hil~
2 Data Scientist         <NA>     Luxoft USA ~ Midd~ NJ    Usa     Middletown
3 Data Scientist         <NA>     Cincinnati ~ New ~ NY    Usa     New york
4 Data Scientist, Aladd~ Account~ BlackRock    New ~ NY 1~ Usa     New york
5 Senior Data Scientist  biotech  CyberCoders  Char~ NC    Usa     Charlotte
# i 10 more variables: inferred_state <chr>, inferred_country <chr>,
#   post_date <date>, job_description <chr>, job_type <chr>,
#   salary_offered <chr>, job_board <chr>, geo <chr>, crawl_timestamp <chr>,
#   url <chr>
```

1. Let's start by looking at the job title. We see that all of the job titles for the first few entries include "data scientist." Tokenize `job_title` to bigrams (i.e., n-grams with n = 2). Use a bar chart to show the top ten bigrams that appear in `job_title`, in order from most to least frequent.

What are the most common bigrams? Do they make sense to you?

...

2. Let's look at the category of the jobs in `job_posts`. Tokenize `category` into individual words and remove the stop words contained in `stop_words`. Assign the tokens to the variable `word`, then `count` up the unique instances of `word`, and assign the resulting data frame to the name `category_count`. Then use a bar chart to show the top 10 most common words, in order from most to least frequent.

3. Let's try using a word cloud to visualize the tokens we extracted from `category`. Since this is not something we covered in class, we have provided the code for you below. You just need to remove `eval = FALSE` from the top of the code chunk, so it says `{r out.width = "50%"}` rather than `{r out.width = "50%", eval = FALSE}`.

```r
wordcloud(
  words = category_count |> pull(word), # make sure category_count, word
  freq = category_count |> pull(n),     #   and n are in your data frame
  scale = c(3, 0.5),
  min.freq = 5,
  max.words = 100,
  random.order = FALSE,
  rot.per = 0.30,
  colors = brewer.pal(8, "Dark2")
)
```

Comment on the word cloud: Is it easy to digest?

…

4. Where are these jobs located? Use a bar chart to show the number of job postings of the top 10 cities.

5. What software skills are most commonly required for these jobs? To find out, create logical variables to indicate whether each `job_description` contains the following skill requirements: R, python, tableau, java, and sql. Then calculate the share of postings that require each of these skills, and show them in a bar plot where the y axis ranges from a share of 0 (no listings) to 1 (all listings).

Do your results make sense? If not, can you improve them?

...