CUNY 607 - Project 3

Team 6 Presentation

October 17, 2021

Contents

1 Project Overview

1.1 Team Members

- Donald Butler
- Nick Oliver
- Jeffrey Parks
- Mark Schamfield

1.2 Project Objective

Use data to answer the question, "Which are the most valued data science skills?"

- Identify viable data sources.
- Load data into a relational database with normalized tables.
- Perform tidying, transformaiton and exploratory data analytis in R.
- Analyze data and show findings.

1.3 Project Approach

We considered potential data sources for this project, including:

- Scraping "Data Scientist" job postings from websites like Indeed and LinkedIn.
- Examine Reddit data science-related subreddits for questions related to "skills" or "careers"
- Looking at Python, NPM or R packages that mention specific skills
- Reviewing Youtube videos that mention learning some data science skill
- Interviewing question/interview study guides that are targeted at data scientists
- Finding LinkedIn public profiles of employed data scientists and what skills they list
- Analyzing StackOverflow data science-related pages

After initial discussion, we decided to attempt the first option, scraping 'Data Scientist' job listings.

Our initial idea was to leverage an API endpoint from one of the popular services, such as Indeed.com. However, we found that while Indeed does produce an API, it restricts access to verified Publishers of job postings. In the interests of time, and since we had some team members with experience in web scraping, we pulled job listings directly from the from LinkedIn website using a popular browser automation tool, **Selenium** using Python.

Once we had the data in hand, we identified a multi-step approach to analysis:

- Use a text mining package such **tidytext** to tokenize keywords in the job descriptions
- Manually scan the tokenized keywords for terms describing discrete job skills, and filter out non-skill-related terms.
- Built a relational database with individual tables for Job Listings, Companies and Skills
- Query the database to analyze the counts, frequency and trends.

2 Data Acquisition

Our dataset was based on a search on LinkedIn.com for job listings matching the term “Data Scientist” in the New York, NY Metropolitan region on the evening of October 12, 2021:

https://www.linkedin.com/jobs/search/?geoId=90000070&keywords=data%20scientist&location=New%20York%20City%20Metropolitan%20Area

Results were scraped from an automated browser session managed by the **Selenium** library using **Chromedriver**. The Selenium API received instructions from a Python Jupyter Notebook, and was executed in two sessions.

The first session navigated to the LinkedIn search results, collected the job listing titles and dedicated URLs from the results page, and then automatically paginated to the next page of results and repeated the collection until all search results (approximately 1000) were captured. These results were written to a .csv file, which became the input for the second session. This session took approximately 8-10 minutes to complete.

The second session loaded the .csv file with the dedicated URLs for each search result, and looping through these results collected information from each page. This session resulted in 838 complete profiles and took several hours to complete, exempting search results that were deemed invalid (such as “no longer accepting applications” or other flags.)

Results were saved as a .csv file, **job_listings_final.csv**.

The Jupyter Notebook, Selenium API script and supporting data can be found in: https://github.com/nolivercuny/data607-team-6-project-3/tree/master/data/jobs-scraper

3 Data Transformation

Our first step was to transform the raw job descriptions into tokenized fields for analysis. We decided to create one table with all words (except for "stop words") to analyze overall frequency at a high level, and a second table that would identify common multi-word combinations called "ngrams."

This second approach would correctly return the frequency of phrases such as "Machine Learning", instead of separate results for "Machine" and "Learning". We further filtered this second table manually to identify actual job skills, and not extraneous information (such as company names and other filler text.)

Load the job descriptions dataset

```
urlfile<-"https://raw.githubusercontent.com/nolivercuny/data607-team-6-project-3/master/data/job_listing
jobdat <- read_csv(url(urlfile))
jobdat<-data_frame(jobdat)
```

Create jobdata_words.csv

We created this table to analyze overall term frequency, using **tidytext** to un-nest the job descriptions into indivdual words, and removing "stop words" (the, of, to, etc)

```
jobdat_word<- unnest_tokens(
  jobdat,
  word,
  description,
  token= "words",
  format=c("text"),
  to_lower=TRUE,
  drop=TRUE,
  collapse=NULL,
)

jobdat_word <-jobdat_word %>%
  anti_join(stop_words)

write_csv(jobdat_word,'data/jobdata_words.csv')
```

Create jobdat_ngrams.csv

We created this file to analyze terms and phrases that meet criteria for "hard skills." Using **tidytext**, created ngrams of 1, 2, and 3 words, filtered out the common stop_words, and then did a count and filtered the rows that occurred at least 10 times within the raw dataset. The resulting 6015 rows were exported to a .csv file used to manually determine which terms described actual job skills and which did not.

```
jobdat_1gram <- jobdat %>%
  unnest_tokens(ngram,description,token='ngrams',n=1,format='text',
                drop=TRUE,to_lower=TRUE) %>%
  filter(!ngram %in% stop_words$word) %>%
  count(ngram,sort = TRUE) %>%
  filter(n >= 10)

jobdat_2gram <- jobdat %>%
  unnest_tokens(ngram,description,token='ngrams',n=2,format='text',
                drop=TRUE,to_lower=TRUE) %>%
  separate(ngram,c('word1','word2'),sep = " ") %>%
```

```r
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word) %>%
  unite(ngram, c('word1','word2'), sep = " ") %>%
  count(ngram,sort = TRUE) %>%
  filter(n >= 10)

jobdat_3gram <- jobdat %>%
  unnest_tokens(ngram,description,token='ngrams',n=3,format='text',
                drop=TRUE,to_lower=TRUE) %>%
  separate(ngram,c('word1','word2','word3'),sep = " ") %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word) %>%
  filter(!word3 %in% stop_words$word) %>%
  unite(ngram, c('word1','word2','word3'), sep = " ") %>%
  count(ngram,sort = TRUE) %>%
  filter(n >= 10)

jobdat_ngrams <- jobdat_1gram %>%
  rbind(jobdat_2gram) %>%
  rbind(jobdat_3gram) %>%
  arrange(desc(n))

jobdat_ngrams %>%
  write.table(file = './jobdat_ngrams.csv',quote = FALSE,
              sep = '\t', row.names = FALSE)
```

4 Database Operations

To support the data analysis, we transformed the raw dataset into a relational database using **Sqlite**, a file-based format that is both fast and portable, so that all team members could access the database without having to manage any kind of server.

4.1 Create Database and Tables

```
# For testing - use in-memory DB
con <- dbConnect(RSQLite::SQLite(), dbname = ":memory:")

# Uncomment to manipulate real DB
#con <- dbConnect(RSQLite::SQLite(), dbname = "project3_job_skills.db")
```

```
baseUrl <- "https://raw.githubusercontent.com/nolivercuny/data607-team-6-project-3/master/sql/"
tables <- c("company", "job_listing", "job_listing_skill","job_listing_word")
for (table in tables) {
  fileUrl <- paste(baseUrl, table,".sql",sep = "")
  createTableStatement <- getURL(fileUrl)
  print(paste("Creating ", table, " table"))
  dbSendQuery(con, createTableStatement)
}
```

Load Data From CSV

```
urlfile<-"https://raw.githubusercontent.com/nolivercuny/data607-team-6-project-3/master/data/job_listin
jobdat <- read_csv(url(urlfile))
```

Populate company table

Unique row per company (420)

```
companiesDf <- jobdat %>%
  select(company_name, company_size, industry) %>%
  distinct()

companiesDf$company_name <- companiesDf$company_name %>% replace_na("unknown")
dbWriteTable(con,"company",companiesDf, append=TRUE)
```

column_name	data_type	attrs
id	int	pk
company_name	int	not null
company_size	int	null
industry	int	null

Populate job_listing table

Unique row per job listing (849)

```
# Remove columns that are in the company table
jobListingDf <- jobdat %>% select(-c("company_size", "industry"))

# Fix the one company name that is NA
jobListingDf$company_name <- jobListingDf$company_name %>% replace_na("unknown")

# Read companies to get company ID for joining
companiesWithId <- dbReadTable(con,"company")

# Join job listing with company to populate id, then drop company-specific cols
# rename id column to company id
joined <- left_join(jobListingDf, companiesWithId, by="company_name") %>%
  select(-c("company_size", "industry","company_name")) %>%
  rename(company_id = id)

#write dataframe to job_listing table
dbWriteTable(con, "job_listing", joined, append=TRUE)
```

column_name	data_type	attrs
id	int	pk
search_rank	int	not null
job_title	text	not null
region	text	not null
applicant_count	int	null
salary	text	null
employment_type	text	not null
career_level	text	null
description	text	null
date_queried	text	null
date_posted	text	
company_id	int	fk

Populate job_listing_word table

Unique row per word (18,000)

```
wordsDataUrl <- "https://raw.githubusercontent.com/nolivercuny/data607-team-6-project-3/master/data/job
wordsJobListingData <- read_csv(url(wordsDataUrl))
wordsJobListingData<-data_frame(wordsJobListingData)

wordsDf <- wordsJobListingData %>% select(search_rank, word)
jobListingWithId <- dbReadTable(con,"job_listing")
joinedWords<-left_join(wordsDf, jobListingWithId, by="search_rank") %>%
  select(id, word) %>%
  rename(job_listing_id = id, skill = word)
dbWriteTable(con,"job_listing_word", joinedWords, append=TRUE)
```

column_name	data_type	attrs
id	int	pk
skill	text	not null

Populate job_listing_skill table

Unique row per tokenized skill (53).

We loop though the list of skills and use regex to determine if the skill is listed within the description text. A new attribute is created in the jobs data frame that indicates if the skill is required.

```
skillsDataUrl <- "https://raw.githubusercontent.com/nolivercuny/data607-team-6-project-3/master/JobSkil
skillsData <- read_csv(url(skillsDataUrl))
skillsDf <-data_frame(skillsData)

jobListingWithId <- dbReadTable(con,"job_listing")
skillsTableDf <- data.frame(job_listing_id=integer(), skill=character())
for (i in 1:nrow(jobListingWithId)) {
  listing <- jobListingWithId[i,]
  for (j in 1:nrow(skillsDf)) {
    skill <- skillsDf[j,]
    detected <- str_detect(listing$description,
                           regex(paste('[^A-Z0-9]',skill,'[^A-Z0-9]',sep = ''),
                                 ignore_case = TRUE))
    if(detected==TRUE){
      skillsTableDf <- skillsTableDf %>%
        add_row(job_listing_id = listing$id, skill = skill$JobSkill)
    }
  }
}

dbWriteTable(con,"job_listing_skill", skillsTableDf, append=TRUE)
dbDisconnect(con)
```

column_name	data_type	attrs
id	int	pk
job_listing_id	int	fk
skill	text	not null

5 Exploratory Data Analysis

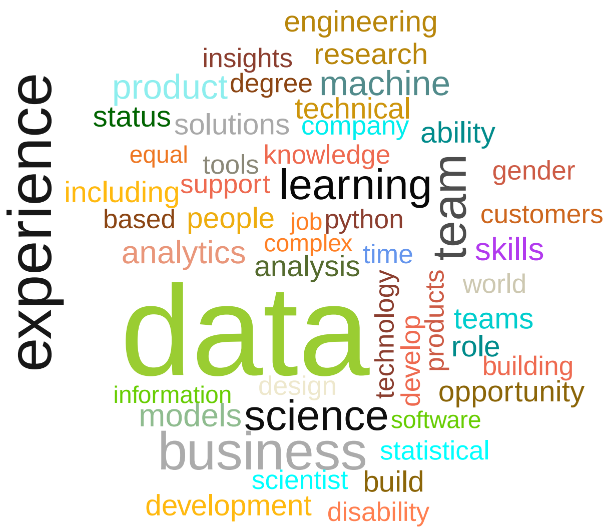
5.1 Load from database / create fact tables

```
# replace with relative or remote path to db
dbLocation <-"/Users/jeff/Development/data607-team-6-project-3/data/project3_job_skills.db"
con <- dbConnect(RSQLite::SQLite(), dbname = dbLocation)

# get tables
tables <- dbListTables(con)
company_df <- dbReadTable(con, 'company')
listing_df <- dbReadTable(con, 'job_listing')
skills_df <- dbReadTable(con, 'job_listing_skill')
words_df <- dbReadTable(con, 'job_listing_word')

# construct fact tables
job_listings <- listing_df %>%
  inner_join(company_df, by = c('company_id' = 'id'))

job_listings_skills <- job_listings %>%
  inner_join(skills_df, by = c('id' = 'job_listing_id'))
```

5.2 Frequency - All Terms

For our initial EDA, we wanted to look at the raw frequency of terms among all 849 job listings, and identify some of the most commonly-used words.

```
words_df %>%
  count(skill) %>%
  with(wordcloud(skill,n,scale=c(4,0.5),max.words=50,
                 random.color = FALSE,rot.per=0.25,colors = colours()))
```

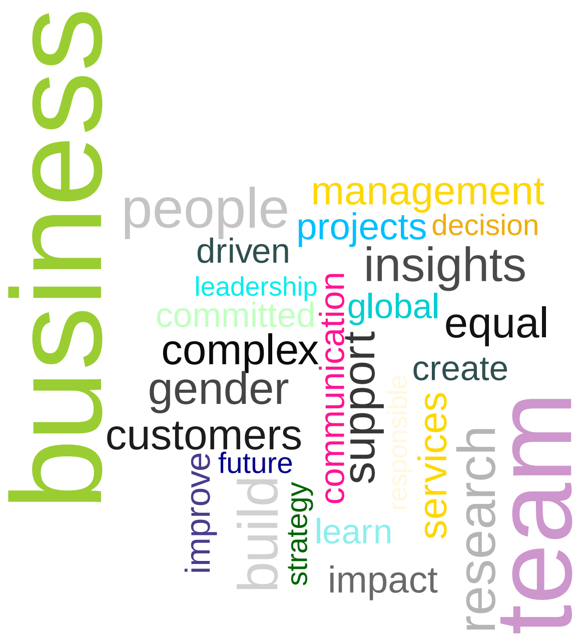

engineering
insights
research
product
degree
machine
status
solutions
technical
company
ability
equal
tools
knowledge
experience
including
support
learning
team
gender
based
people
job
python
customers
analytics
complex
time
skills
analysis
world
data
technology
develop
products
teams
role
building
information
design
opportunity
models
science
software
business
statistical
scientist
build
development
disability

5.3 Skills

5.3.1 Soft Skills

This wordcloud graph of all terms effectively highlights both technical and “soft skill”terms with high frequencies. If we apply a manual filter for some of the common “soft skills” in our dataset, we get a clearer picture of what employers are looking for in this category from Data Scientists:

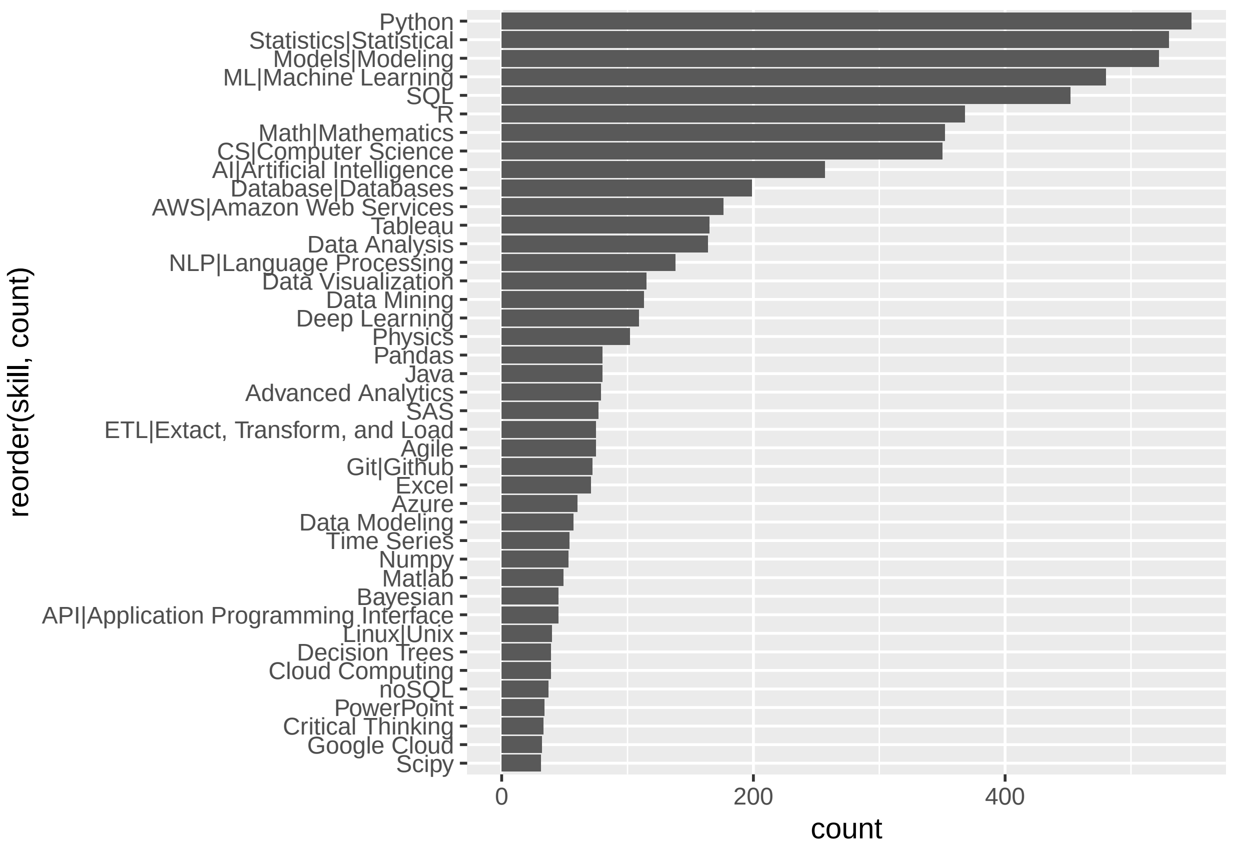
```
soft_skills<-c("business","team","build","people","research","insights","gender","support","communicati

words_df %>%
  filter(skill %in% soft_skills) %>%
  count(skill) %>%
  with(wordcloud(skill,n,scale=c(4,0.5),max.words=50,
                 random.color = FALSE,rot.per=0.25,colors = colours()))
```



5.3.2 Hard Skills

5.3.2.1 Technical Skills for Data Scientist or Data Analyst in New York, NY Our analysis of technical skill frequency suggests that candidates are expected to be versed in Python, R and SQL as foundational skills.

```
job_listings_skills %>%
  filter(grepl('data scientist|data analyst',job_title,ignore.case = TRUE)) %>%
  group_by(skill) %>%
  summarize(count = n()) %>%
  filter(count >= 30) %>%
  ggplot(aes(x = reorder(skill,count), y = count)) + geom_bar(stat = 'identity') + coord_flip()
```

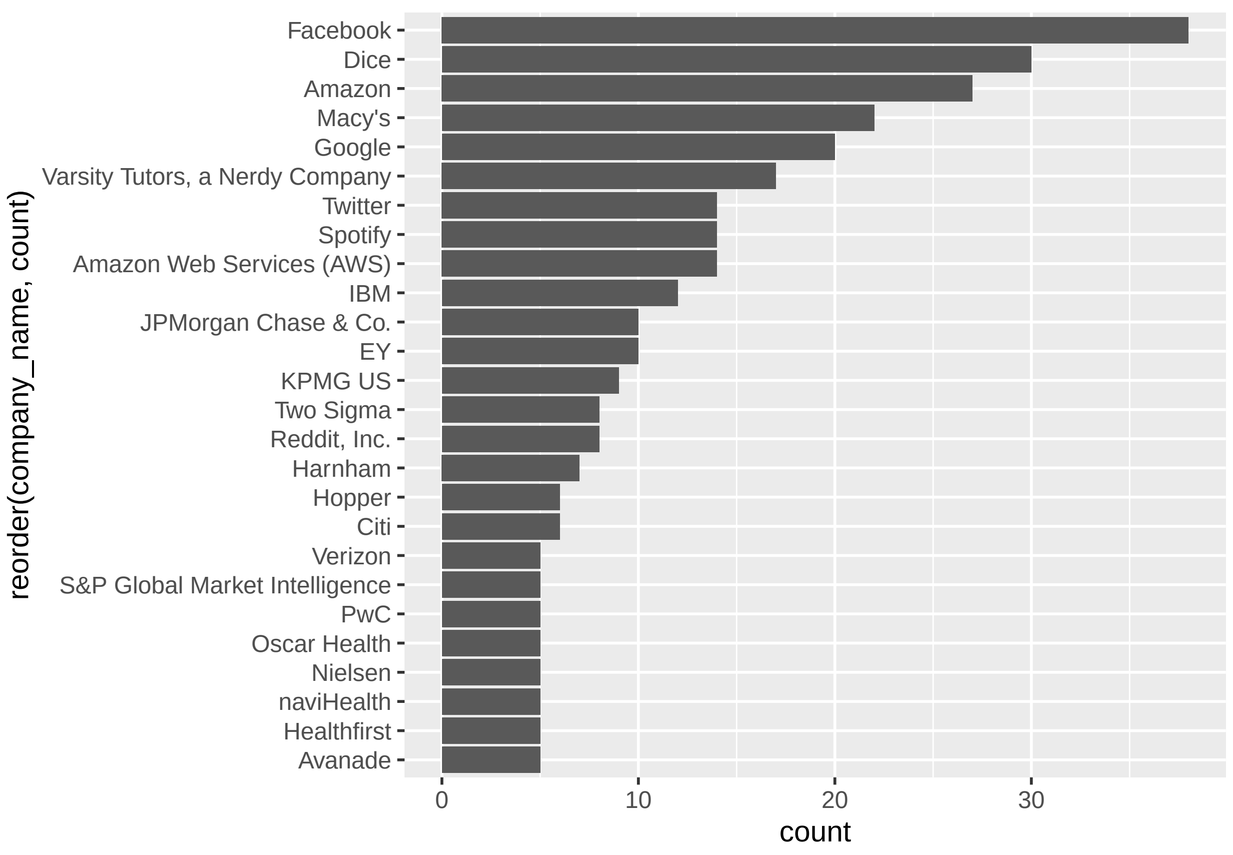


5.4 Companies

5.4.1 Companies posting Data Science jobs in New York, NY

As expected, FAANG companies dominated the listings but there was also strong activity in the Finance, Media and Healthcare sectors.

```
job_listings %>%
  group_by(company_name) %>%
  summarize(count = n()) %>%
  filter(count >= 5) %>%
  ggplot(aes(x = reorder(company_name,count), y = count)) + geom_bar(stat = 'identity') + coord_flip()
```



5.4.2 Companies requiring the most skills

Certain companies were very detailed in their postings, with high average frequency of technical skills outlined in their job description.

```
companySkillsCountDf <- dbGetQuery(con,
  "select c.company_name,
  count(jls.skill) as required_skills
  from job_listing jl
  join job_listing_skill jls on jl.id = jls.job_listing_id
  join company c on c.id = jl.company_id
  group by company_name
  order by required_skills desc;")

companyJobListingsCountDf <- dbGetQuery(con,
  "select c.company_name,
  count(jl.id) as number_job_listings
  from job_listing jl
  join company c on c.id = jl.company_id
  group by company_name
  order by number_job_listings desc;")

joinedDf <- left_join(companyJobListingsCountDf,
                      companySkillsCountDf,
                      by="company_name") %>%
  mutate(mean_skills_per_listing = round(required_skills / number_job_listings))

joinedDf %>%
  select(company_name, mean_skills_per_listing) %>%
  arrange(desc(mean_skills_per_listing)) %>%
  top_n(10, mean_skills_per_listing) %>%
  kable(format='simple')
```

company_name	mean_skills_per_listing
Johnson & Johnson	26
Lexmark	24
Oracle	21
Disney Streaming	21
Nuveen, a TIAA company	20
Zebra Technologies	19
Sumitomo Mitsui Banking Corporation	19
Kubient	19
EVERSANA	19
Evolent Health	18
ViacomCBS	18
Venusgeo Solutions	18
The Mom Project	18
City Experiences	18
BMC Software	18

6 Conclusions

- Top “Hard Skills” included Python, SQL and R as top platforms/languages, with strong emphasis on Statistics, Modeling, Machine Learning and Mathematics.
- Top “Soft Skills” included Teams, Management, Communication, Strategy and Customers.

During our analysis, we also identified additional questions for further investigation, such as:

- How do Hard and Soft Skills vary by Industry and by Company?
- Are certain high-volume companies using basic job description templates? If so, what are the real differntiators from listing to listing?
- What specific mathematical and statistical skillsets are most desired/required?