IDENTIFYING INEQUALITY IN RECOMMENDATIONS: A FRAMEWORK AND EXPERIMENTAL STUDY

An Undergraduate Research Scholars Thesis

by

DIVA KOHLI

Submitted to the LAUNCH: Undergraduate Research office at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by
Faculty Research Advisor: Dr. James Caverlee

May 2022

Major: Computer Science

RESEARCH COMPLIANCE CERTIFICATION

Research activities involving the use of human subjects, vertebrate animals, and/or biohazards must be reviewed and approved by the appropriate Texas A&M University regulatory research committee (i.e., IRB, IACUC, IBC) before the activity can commence. This requirement applies to activities conducted at Texas A&M and to activities conducted at non-Texas A&M facilities or institutions. In both cases, students are responsible for working with the relevant Texas A&M research compliance program to ensure and document that all Texas A&M compliance obligations are met before the study begins.

I, Diva Kohli, certify that all research compliance requirements related to this Undergraduate Research Scholars thesis have been addressed with my Research Faculty Advisor prior to the collection of any data used in this final thesis submission.

This project did not require approval from the Texas A&M University Research Compliance & Biosafety office.

TABLE OF CONTENTS

ABSTRACT

Identifying Inequality in Recommendations: A Framework and Experimental Study

Diva Kohli
Department of Computer Science and Engineering
Texas A&M University

Research Faculty Advisor: Dr. James Caverlee
Department of Computer Science and Engineering
Texas A&M University

Recommender systems have become increasingly prevalent online within the last two decades and are used extensively on social media and networking platforms. We look at one such platform, LinkedIn, which serves the purpose of matching job candidates to open jobs and connecting recruiters to job seekers. More specifically, we look into the approaches used to design recommendation systems and analyze how LinkedIn's search and recommendation algorithms work. However, an issue that arises from the use of recommender systems for job searching is that certain jobs may be shown more often to candidates based on their descriptions, possibly resulting in bias when it comes to networking and potential hiring chances. Because of the impact this could have on both companies and candidates, it is important to ensure that recommendations resulting from these systems are not biased or unfair.

Through this thesis, we propose a framework for the systematic study of inequality that manifests in recommendation results. In particular, we focus on inequality in job recommendations, where we quantify this inequality in terms of the Gini coefficient. In our preliminary experimental studies, we find that some factors may have a greater influence on the recommendations made by the algorithm than others. Notably, we observe that many job recommendations show

high degrees of concentration with respect to companies, and we hope to build upon this work to explore inequality in terms of other factors as well.

DEDICATION

To my family, friends, and instructors who supported me throughout the research process.

ACKNOWLEDGMENTS

Contributors

I would like to thank my faculty advisor, Dr. James Caverlee, and my peers in the Infolab for their guidance and support throughout the course of this research.

Thanks also go to my friends and the Computer Science and Engineering department faculty and staff for making my time at Texas A&M University a great experience.

Finally, thanks to my family for their encouragement and love.

The data collected for this project was from LinkedIn, and the analysis was done by the student and research advisor.

All other work conducted for the thesis was completed by the student independently.

Funding Sources

This work was not funded or sponsored by any organization.

1. INTRODUCTION

For many people on social networks, recommender systems are quite prominent in our daily lives and are meaningful as they can influence both our opinions and some of the decisions we make. For example, research shows that ads shown on social media platforms can be influenced by (or even influence) a user's political views [1]. Additionally, the decisions we make based on the results of recommender systems can range from choosing a video to watch on YouTube to picking a candidate on LinkedIn to interview for a job. Although the former decision may not have a long-lasting impact on the user, the latter may. That is, a platform like LinkedIn could allow certain candidates' profiles to be more visible to recruiters because of keywords in their profile, resulting in bias when it comes to potential hiring chances. Alternatively, certain job openings could be more visible to candidates than others because of certain keywords or commonly found skills listed in the job description. Due to this, it is very important that recommendations based on user or product information are not biased or unfair. Our research aims to provide a framework to look into and address these issues in order to ensure that recommendation algorithms, specifically those used on LinkedIn, are fair and provide unbiased results.

There are several approaches to recommender systems, including collaborative filtering [2] and content-based filtering [3], which we will be looking at in more detail for this project. Essentially, we would like to see how the amount and type of content information provided about an item affects how much fairness or bias there is in how it is recommended. Applying this to social media, we will look at how the amount of information on a person's profile or in a job description could affect its appearance in suggestions to other users. A solution to any issues found during this process could help make networking easier and fairer, and it could potentially help give people more equal professional opportunities.

1.1 Approaches to Recommender Systems

Both the methods mentioned above, collaborative filtering and content-based filtering, are used by many social networks, including LinkedIn.

1.1.1 Collaborative Filtering

Collaborative filtering results in a model that is based on a user's past interactions with a platform as well as the behaviors of other users [3]. This would mean that a user might have recommendations based on what other people like them are getting. For example, on LinkedIn, this approach is used to suggest to users jobs that other users also viewed.

1.1.2 Content-Based Filtering

Content-based filtering uses the characteristics of an item to recommend items that are similar [3]. That is, it interprets information about an item in order to find items similar to it and then recommends those items. In the case of LinkedIn, this approach results in a similar jobs feature, which recommends to the user job postings that are similar to what the user has been browsing recently or has shown interest in.

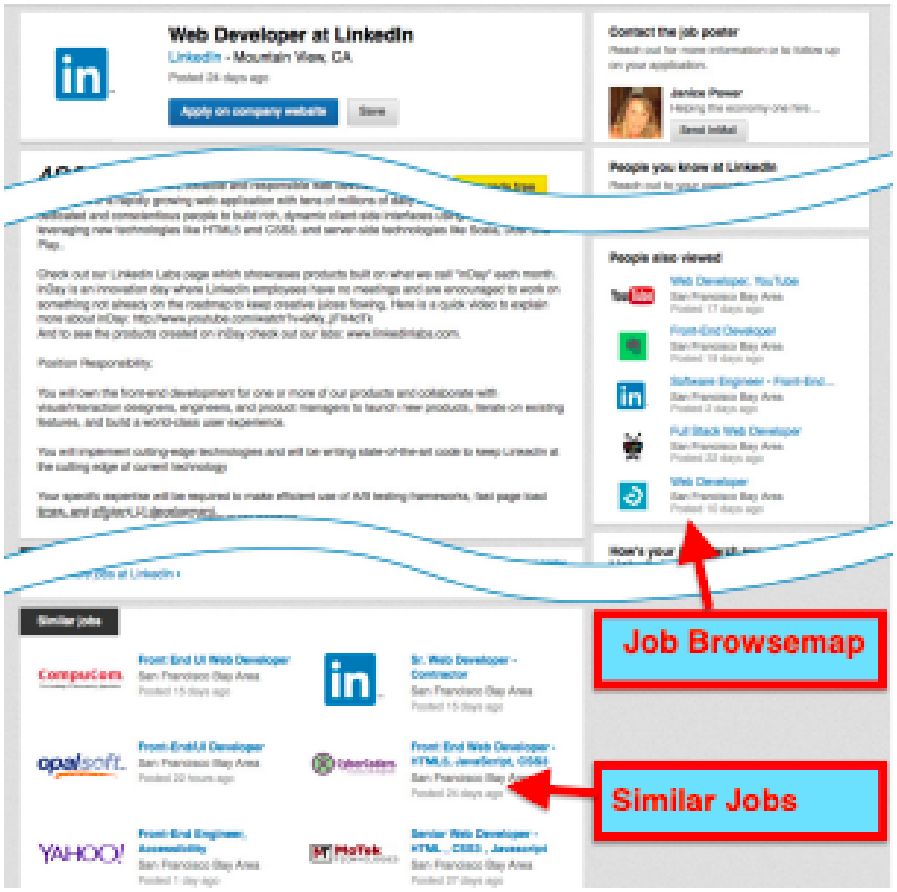


Figure 1.1: Use of both collaborative and content-based filtering on LinkedIn [4]

Although there are some issues with each of these approaches individually, their use in combination helps provide a more robust and personalized experience for users, as seen in the figure above (Figure 1.1). This is because users can browse jobs based on what they have shown to have preferred in the past as well as the preferences of other users similar to them [4].

1.2 Understanding LinkedIn and Talent Search

LinkedIn can be described as a platform that matches job seekers to job openings. One product that helps them do this is LinkedIn Recruiter, a feature that allows employers and recruiters to find talent that is relevant to their open job postings and likely to be a good match for the position. It does so by accepting a search request and then providing a list of candidates that are ranked based on factors such as their work experience, location, and how likely they are to respond [5]. From the LinkedIn Engineering team, there are a few challenges that are associated with coming up with machine learning models that can accurately provide results:

1. Typical recommendation systems are based only on how relevant a search result is to the entered query, but in the case of a job/talent search system, there should be mutual interest between the candidate and recruiter [5]. This means that it is important not only that the candidate matches the search query, but also that the candidate is interested in the job for which they are being recommended to the recruiter.

2. Recruiters could try to search with complex queries, including, for example, a combination of job titles, company names, and skills. Their search method could also be implicit, such as when they provide a job posting to find suitable candidates and would like the recommendation system to provide them with matches based just on the job profile. The model should thus be able to account for these varieties in search methods and be able to return relevant and unbiased results [5].

3. Sometimes, recruiters may not know the best search queries to find their ideal candidates. For example, a job may require a candidate with certain specific technical skills that only a technical recruiter with expertise in the field would be able to accurately describe. Thus, it is important that the model does not rely solely on the search query, but also learns the recruiter's preferences based on their usage data or by their interactions with the platform and with recommended candidates during a session [5].

Article [5] describes some of the approaches LinkedIn has taken to address these issues. These include an offline modeling pipeline that trains the models used to rank candidates based on recruiter usage and interactions with candidates. LinkedIn engineers have summarized the architecture of their talent search and recommendation system in the diagram below (Figure 1.2).

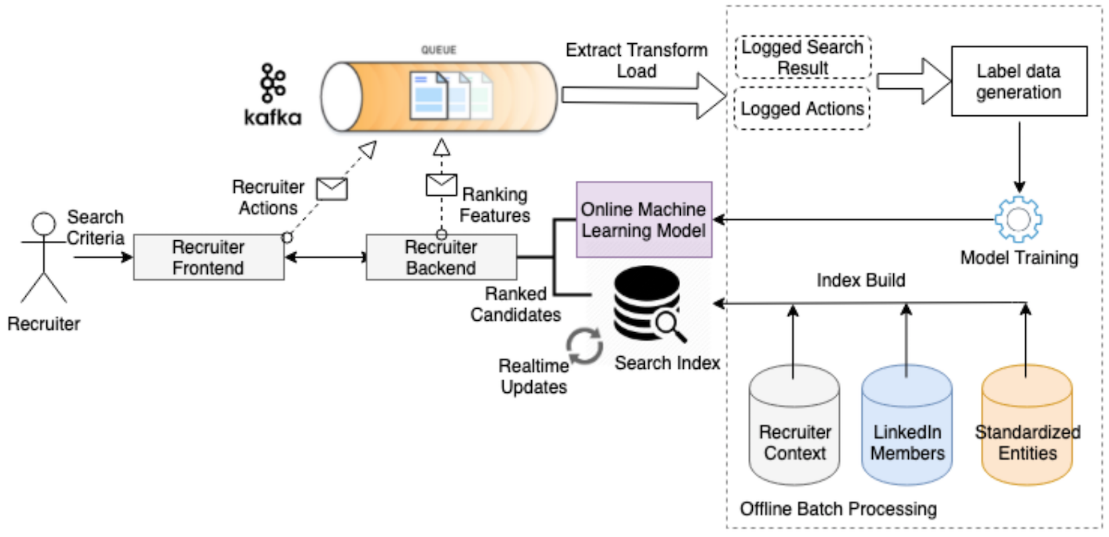


Figure 1.2: LinkedIn’s talent search and recommendation system architecture [5]

The architecture of the online system begins when the recruiter’s search query and the context of the recruiter and session are sent to the LinkedIn search engine. Then, a result set of candidates is retrieved based on the search criteria, and several machine learning models are used to rank them. Both the results and features of the model are recorded and are used later to train models. Lastly, the front-end server generates the results page by compiling the top-ranked candidates, and it also keeps track of the recruiter’s interactions with these candidates for future improvement and personalized recommendations [5].

1.3 Gini Coefficient

Later in our analysis, we use a term known as the Gini coefficient to measure the extent of inequality within a distribution. Developed in 1912 by Italian statistician Corrado Gini, this term is traditionally used in economics to measure the distribution of incomes among a population [6]. We decided to use this as a measure of inequality in our research because it provides a single number that can be used to compare two distributions, which helps us understand how much inequality may exist among them.

The Gini coefficient estimates how much deviation there is from a perfectly equal distribution by measuring the area between the Lorenz curve and the line of absolute equality. Graphically, this curve plots the proportion of the total income/frequency on the y-axis against the proportion of the population on the x-axis. A straight diagonal line represents perfect equality, and the Gini coefficient essentially is the ratio of the area between this line and the Lorenz curve over the total area below the line of equality [6]. This can be calculated using the equation (Eq. 1.1) below, in which G is the Gini coefficient, A is the area between the line of equality and the Lorenz curve, and B is the remaining area below the Lorenz curve.

$$G = A/(A + B) \qquad \text{(Eq. 1.1)}$$

The Gini coefficient is expressed as a percentage of the maximum area under the equality line. So, a Gini coefficient of 0 (0 %) would mean that there is perfect equality (everyone is equally represented) and a coefficient of 1 (100 %) would mean that there is perfect inequality [7]. In other words, the further the Lorenz curve is from the line of equality, the higher the Gini coefficient will be, representing more inequality. A lower Gini coefficient, thus, would be an indicator of a more equal distribution.

2. METHODS

This chapter describes the process we went through and the challenges we encountered while trying to figure out how to run experiments to better understand LinkedIn's algorithm and any biases it may result in.

2.1 Factors to Ensure Unbiased Experimentation

Although we wanted to look into LinkedIn's highly personalized algorithms, we also wanted to make sure that our research was comprehensive and that our experiments could potentially be replicated. A challenge we faced while coming up with experimentation ideas was how we were going to test the algorithm without having any factors of bias, including having a public LinkedIn profile or signing in to LinkedIn before conducting any experiments. We concluded that the best way to conduct experiments and collect data fairly was if we just looked at the information and recommendations we could find while being in an incognito window on our browser (Google Chrome) and not being logged in to LinkedIn. Though this did limit some of the information we could gather, we were able to shift our focus a little bit to the job recommendation algorithm that LinkedIn utilizes as opposed to searching for specific people.

2.2 Initial Experiments Run

Within the general job search feature in LinkedIn, we had the option to search with and filter by a variety of keywords and categories, such as location, company, salary, experience level, and job type. For our initial experiments, we decided to focus on skills required for the jobs as opposed to facts like location or salary. That is, we entered search queries consisting only of skills or related terminology. The two main areas we looked into were:

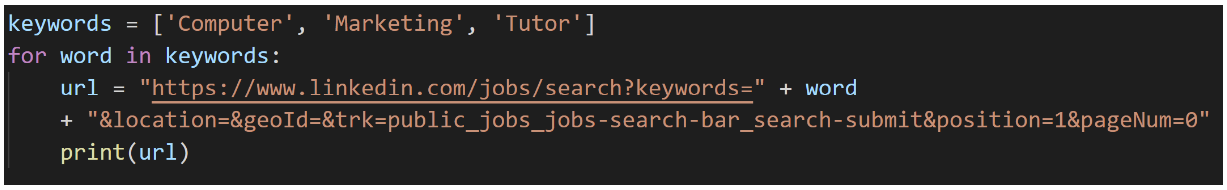
1. Soft Skills (e.g. Communication, Presentation Skills, Teamwork, Management, Leadership Experience, Public Speaking, etc.)

2. Technical Skills (e.g. Computer Engineering, Programming, Translation, Java, Adobe Photoshop, etc.)

2.3 Generalizing Data Collection Framework

Although we found some interesting results from our initial manual experiments (described in the Results chapter below), this method of data collection was quite tedious, and it took a lot of time for us to document the data that we collected. So, we decided that it would be best to automate this process by writing code that would run various search queries quickly and collect the results.

To do this, we decided to use the coding language Python, and its package, Beautiful Soup. The first part of our code included a list of search terms that we wanted the program to go through automatically, as seen below (Figure 2.1).

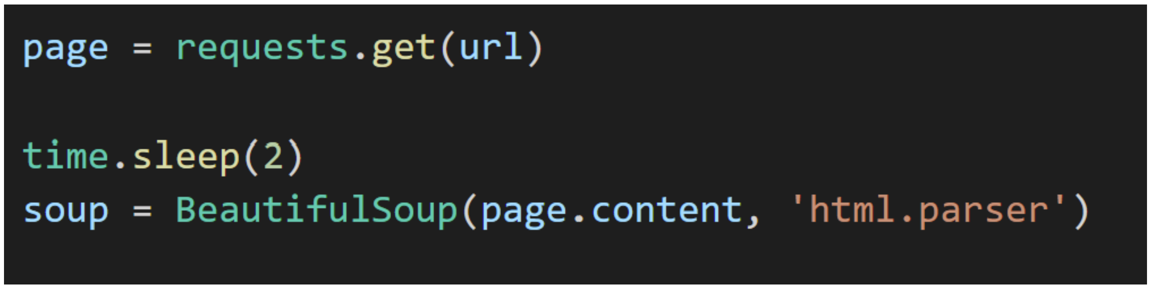


```
keywords = ['Computer', 'Marketing', 'Tutor']
for word in keywords:
    url = "https://www.linkedin.com/jobs/search?keywords=" + word
    + "&location=&geoId=&trk=public_jobs_jobs-search-bar_search-submit&position=1&pageNum=0"
    print(url)
```

Figure 2.1: Creation of custom URL given search queries

We can see above that the keywords array stores a list of the search queries that we wish to run. Although the code snippet above contains only 3 search terms, we expanded this to about 20+ terms when we ran our actual experiments. Next, we have a for loop that goes through each query and puts it in a custom URL that can be used to search for it. To come up with the custom URL link, we inspected the URLs of various search queries on LinkedIn manually and were able to find a pattern, which allowed us to know exactly where the search term would be placed in the URL. We used this to concatenate each query in the keywords list with the other portions of the URL. We then printed the URL to make sure that it was properly formatted.

The next portion of our code generated a page object and a Beautiful Soup object to get the contents of the page.

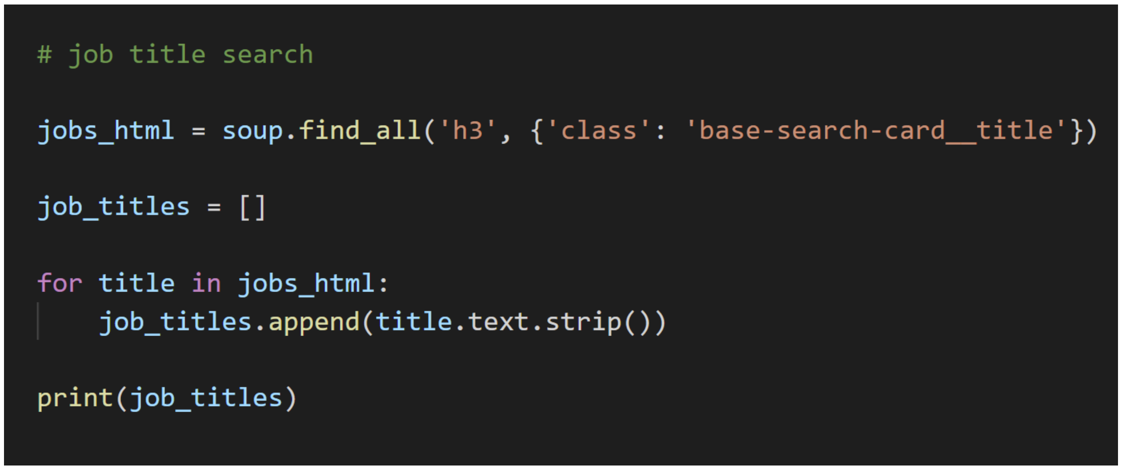


```python
page = requests.get(url)

time.sleep(2)
soup = BeautifulSoup(page.content, 'html.parser')
```

Figure 2.2: Sending a get request to retrieve URL contents

As can be seen in the figure above (Figure 2.2), we first sent an HTTP get request to retrieve the results of our custom URL. Then, we stored the contents of the page in a Beautiful Soup object, which we would later use to extract specific information from the page. We also included a sleep statement so that we would not crowd the server with too many requests.

After this, we had to extract information from the results of each search query, which we did as follows.

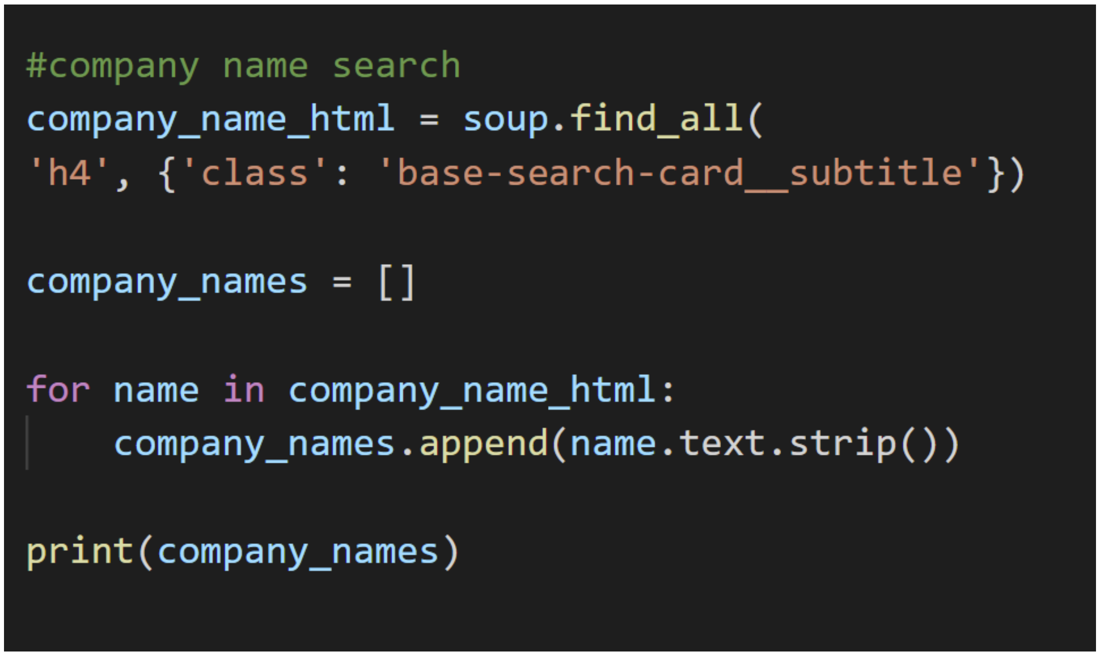


```python
# job title search

jobs_html = soup.find_all('h3', {'class': 'base-search-card__title'})

job_titles = []

for title in jobs_html:
    job_titles.append(title.text.strip())

print(job_titles)
```

Figure 2.3: Finding all the job titles

The code snippet above (Figure 2.3) illustrates how we extracted relevant data from each of the search results. We began by inspecting the HTML contents of the results page on LinkedIn and determining how pages were structured. We found the class that stored job title information and then used a built-in function to find all occurrences of that class in the soup object. We then created an empty array to store the job titles and appended to it each job title that we found on the search results page. Lastly, we printed the array of job titles to make sure that the titles were populated correctly, and we checked this against a manual search query for the keyword as well to ensure that it was accurate.

We repeated this code structure to get results for all the information that we wanted to extract from the page, including job locations, job posting dates, and company names (as seen below).

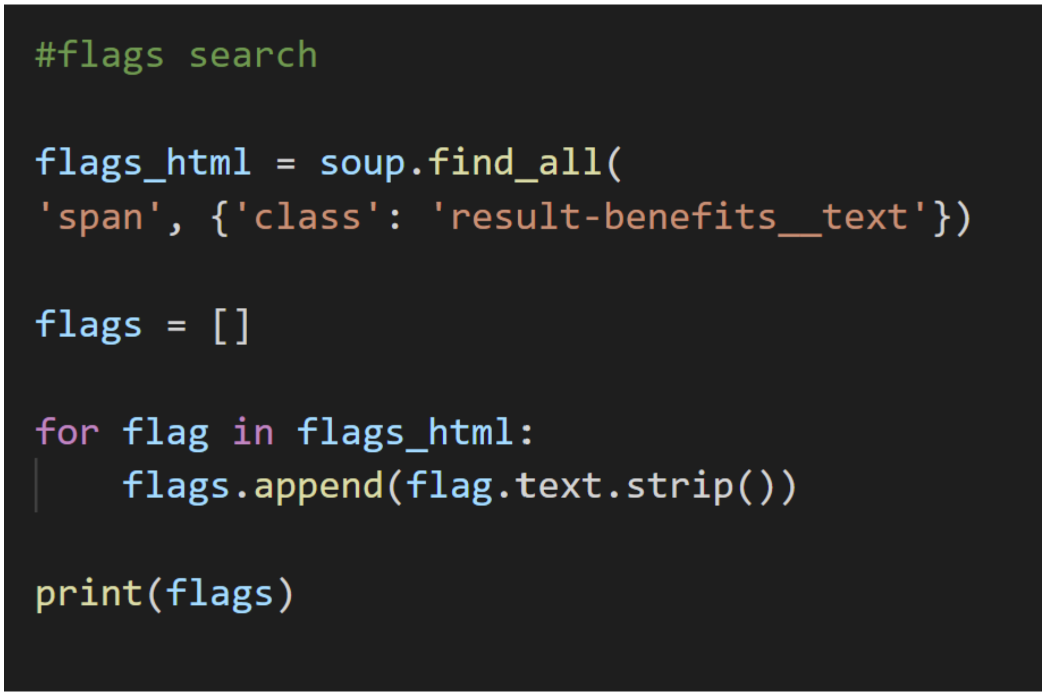


```python
#company name search
company_name_html = soup.find_all(
'h4', {'class': 'base-search-card__subtitle'})

company_names = []

for name in company_name_html:
    company_names.append(name.text.strip())

print(company_names)
```

Figure 2.4: Finding all the company names

The code above (Figure 2.4) shows how we extracted the names of companies who had posted jobs listed in the results. This code was also based on our inspection of the page HTML, which was important as in this case, the class in which company names were stored was called ‘subtitle’ and not ‘company name’ (as would be expected).

Another piece of information that we wanted to extract was any flags that each job posting contained, as these were significant in the results of our manual experiments (shown in the Results chapter below). These flags included whether a user could ‘Be an early applicant’ for a job posting or if the company was ‘Actively Hiring’ for the position. We found by inspecting the page that these were dependent on the time we searched, but any flags were stored in a ‘result-benefits’ class as shown below.



```python
#flags search

flags_html = soup.find_all(
'span', {'class': 'result-benefits__text'})

flags = []

for flag in flags_html:
    flags.append(flag.text.strip())

print(flags)
```

Figure 2.5: Finding flags listed for each job posting

The image above (Figure 2.5) shows how we were able to extract the text of any flags that each job posting contained. The structure of the code is similar to what we used for the other information, putting any flags in an array and then printing it to make sure it was correct.

The next part of our methods included coming up with a way to display all these results so that they were more readable and could be analyzed more efficiently. To do this, we decided to put the search result outputs into a CSV/Excel file, from which we could then go and count the frequency of each company's occurrence. We tweaked our code a little bit to achieve this, having it output to a CSV file instead of to the console as it was doing earlier. Once we had our results in the CSV file, we were able to easily collect statistics on the distributions and use them in our analysis. For example, we were able to, for any given keyword, look at how many times each company appeared on the first page of results, and then analyze all the values for each company to see whether there were any patterns or biases in the way the results were displayed. The frequency of each company's appearance in the results would also help us determine the Gini coefficient for each search query's distribution, as will be shown in the following chapter.

3. RESULTS

In this chapter, we present all the results of our experiments, both the manual ones as well as the automatic data collection. We highlight patterns and significant information we found that helped us better focus the scope of our project.

3.1 Results of Initial Experiments

We recorded the results of our manual experiments in the form of tables in an Excel file. In the sample results below, we noted the top five recommendations given a search term, in addition to the amount of time ago a job was posted, the company name, position name, location, and additional flags given by LinkedIn (EA - be an early applicant, AH - actively hiring). The following table (Table 3.1) shows the results from a search query containing a soft skill.

Table 3.1: Job search results for query 'Presentation Skills'

Posted	**Flags**	**Company**	**Position Name**	**Location**
1 day	AH	Cognizant	Instructional Designer - Remote	Austin, TX
3 hours	EA	LiB.energy	Content Creator	United States
2 weeks	EA	Varsity Tutors	Remote Presentation Skills Expert	United States
4 weeks	EA	New Orleans Pelicans	Game Experience Associate	Metairie, LA
2 hours	AH	Insight Global	Educations Solutions Consultant	United States

As can be seen above, the results have a wide variety, not just in their companies and position names, but also in how long ago the jobs were posted. However, as we will see below, this variety is not necessarily present in the results for all search queries. The following table (Table 3.2) shows the results from a search query containing a technical term.

Table 3.2: Job search results for query ‘Computer’

Posted	Flags	Company	Position Name	Location
4 hours	EA	Opus Consulting Solutions	Entry Level Programmers	Alpharetta, GA
1 week	EA	Seagate Technology	Computer Engineer	Fremont, CA
1 day	EA	Select Source International	Computer Engineer	Altlanta GA
3 hours	EA	Walmart	Data Scientist/ML Eng. (CV)	Dallas, TX
4 days	EA	Pratt And Whitney	Computer Science Specialist	Puerto Rico

The results seen in the table above (Table 3.2) show job titles with a little more relevance to the term entered. However, we can also see that all the top results are flagged with EA - that is, it seems like the system is prioritizing the jobs that have fewer applicants and inviting the user to be an early applicant. This may be beneficial to the job poster but potentially detracts from the quality and relevance of results shown to the candidate. Finally, the next table (Table 3.3) shows the results from a search query containing another soft skill.

Table 3.3: Job search results for query ‘Teamwork’

Posted	Flags	Company	Position Name	Location
51 minutes	EA	Xooma Worldwide	Health and Wellness Coach	US
1 hour	EA	Aston Carter	Continuing Education Coordinator	St.Louis
2 minutes	EA	Solve,Inc Marketing	Management Training, Entry Level	Denver
2 hours	EA	Moncler	Temp. Assistant, Training	NY
42 minutes	EA	Coca-Cola Beverages	Talent and Learning Facilitator	Tampa, FL

These results show a striking preference towards jobs that were posted recently, as all the times are within 2 hours. Moreover, all the top results are again flagged as EA. Both of these observations support the idea that the recommendation system perhaps accounts for time equally or possibly even more than relevance.

Based on the above results and those of the rest of our initial experiments, we were able to better understand how LinkedIn’s recommendation system works for jobs. Overall, we could summarize our main observations as follows:

1. Almost all of the top results were flagged as EA (or AH)

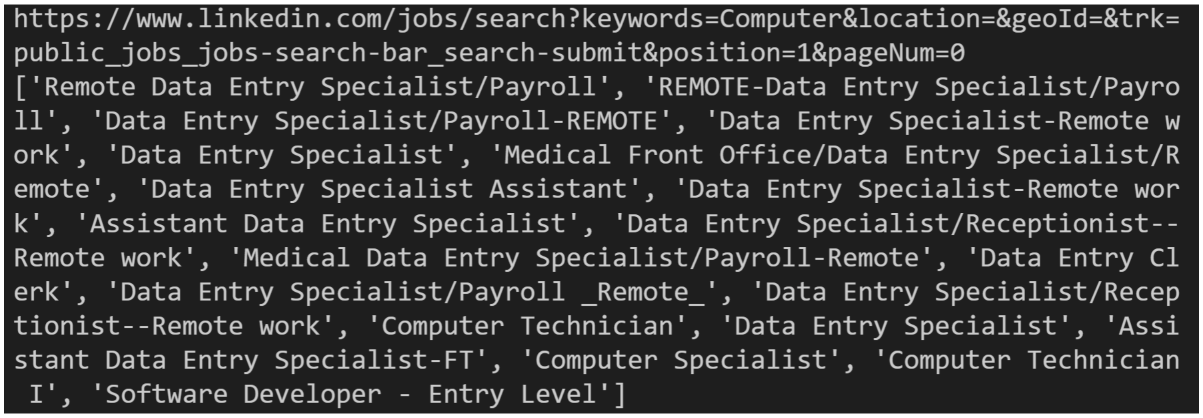
2. Many of the top results were posted quite recently

3. The job titles seemed to match more closely with the search query in the case of technical searches as opposed to searches for soft skills

4. The companies and locations of the jobs were quite varied

3.2 Automatic Data Collection Outputs

As mentioned in the previous chapter, we printed out the contents of our various search results to make sure that our arrays were populating correctly. This also helped us see more patterns in the results that were being returned with each query. The following is an example of the job titles that were printed when we searched with the keyword ‘Computer’.



```
https://www.linkedin.com/jobs/search?keywords=Computer&location=&geoId=&trk=
public_jobs_jobs-search-bar_search-submit&position=1&pageNum=0
['Remote Data Entry Specialist/Payroll', 'REMOTE-Data Entry Specialist/Payro
ll', 'Data Entry Specialist/Payroll-REMOTE', 'Data Entry Specialist-Remote w
ork', 'Data Entry Specialist', 'Medical Front Office/Data Entry Specialist/R
emote', 'Data Entry Specialist Assistant', 'Data Entry Specialist-Remote wor
k', 'Assistant Data Entry Specialist', 'Data Entry Specialist/Receptionist--
Remote work', 'Medical Data Entry Specialist/Payroll-Remote', 'Data Entry Cl
erk', 'Data Entry Specialist/Payroll _Remote_', 'Data Entry Specialist/Recep
tionist--Remote work', 'Computer Technician', 'Data Entry Specialist', 'Assi
stant Data Entry Specialist-FT', 'Computer Specialist', 'Computer Technician
 I', 'Software Developer - Entry Level']
```

Figure 3.1: Job titles returned when search query was ‘Computer’

In the figure above (Figure 3.1), we can see that the custom URL was printed (as we had instructed it to). Below that, we can see a list of the job titles in the search results of the first page. In these sample results, we noticed that many of the job titles were actually quite similar and related more to data entry than computer programming. The following figure shows the output of our array of company names, which helped us better understand why the job titles were so similar.

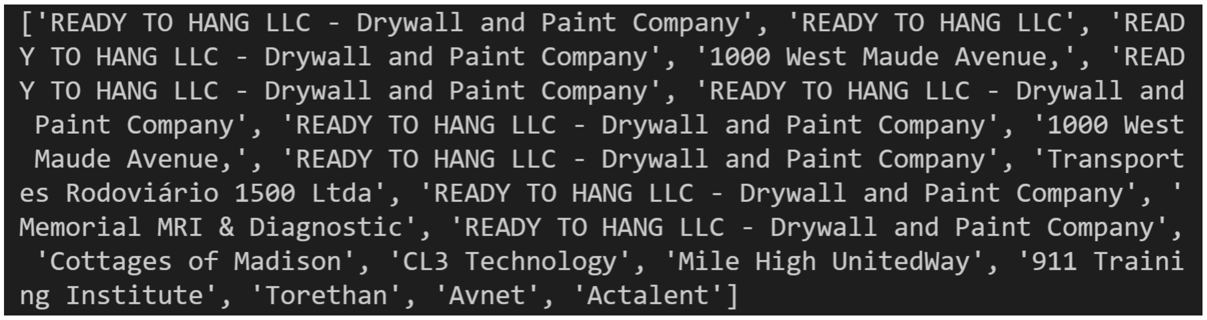
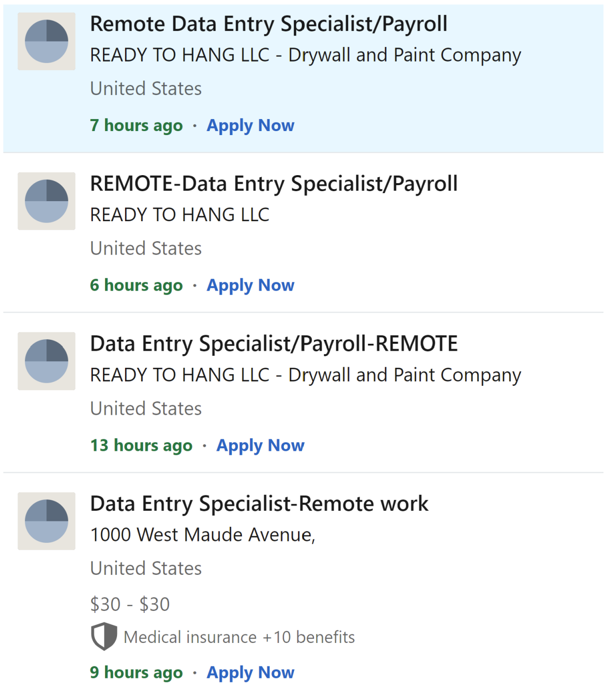

```
['READY TO HANG LLC - Drywall and Paint Company', 'READY TO HANG LLC', 'READ
Y TO HANG LLC - Drywall and Paint Company', '1000 West Maude Avenue,', 'READ
Y TO HANG LLC - Drywall and Paint Company', 'READY TO HANG LLC - Drywall and
 Paint Company', 'READY TO HANG LLC - Drywall and Paint Company', '1000 West
 Maude Avenue,', 'READY TO HANG LLC - Drywall and Paint Company', 'Transport
es Rodoviário 1500 Ltda', 'READY TO HANG LLC - Drywall and Paint Company', '
Memorial MRI & Diagnostic', 'READY TO HANG LLC - Drywall and Paint Company',
 'Cottages of Madison', 'CL3 Technology', 'Mile High UnitedWay', '911 Traini
ng Institute', 'Torethan', 'Avnet', 'Actalent']
```

Figure 3.2: Company names returned when the search query was ‘Computer’

The above image (Figure 3.2) contains the company names associated with each of the job postings seen in the previous one (Figure 3.1). We observed that several of the results were actually jobs posted by the same company, which explained why the titles were so similar. We also noticed, however, that the results were a little further from the types of roles we would have expected to see given this search query. We confirmed that these results were accurate by performing the same search manually, and the figure below shows the results we obtained on LinkedIn.


Remote Data Entry Specialist/Payroll
READY TO HANG LLC - Drywall and Paint Company
United States
7 hours ago · Apply Now

REMOTE-Data Entry Specialist/Payroll
READY TO HANG LLC
United States
6 hours ago · Apply Now

Data Entry Specialist/Payroll-REMOTE
READY TO HANG LLC - Drywall and Paint Company
United States
13 hours ago · Apply Now

Data Entry Specialist-Remote work
1000 West Maude Avenue,
United States
$30 - $30
Medical insurance +10 benefits
9 hours ago · Apply Now

Figure 3.3: LinkedIn results when the search query was ‘Computer’

The picture above (Figure 3.3) helps us see that the results from the code were in line with the results of the query on LinkedIn. We made an important observation from this, which was that many of the top job postings were from the same company and for similar roles. This observation was one of the main things we tried to analyze going forward, as we wanted to look into any inequalities that may exist in the top results presented.

3.3 Results and Analysis of Company Distributions

Following from our observations, we decided to look more closely at the distribution of companies with top job postings. We took results from all the different search queries and analyzed how frequently each company appeared in the top page of results (first 25 results). We did this for queries with both technical and soft skills as keywords.

The following table (Table 3.4) shows the frequency distribution for companies appearing in the top page of results when the search term was ‘Engineer’. The plots below it (Figure 3.4) show a graphical representation of the distribution in the table.

Table 3.4: Company distribution when search term was ‘Engineer’

Company Name	**Frequency**
Harbison-Fischer	1
Boeing	4
Lockheed Martin	1
Vastek Inc	1
Lenovo	1
Smoken Ropes	1
General Dynamics	1
Actalent	10
Modis	1
Rolls-Royce	1
General Motors	1
Moen Incorporated	1
Adient	1

The plot on the left displays the frequencies of each company, while the plot on the right maps these to a Lorenz curve to show the calculation of the Gini coefficient for this distribution [8].

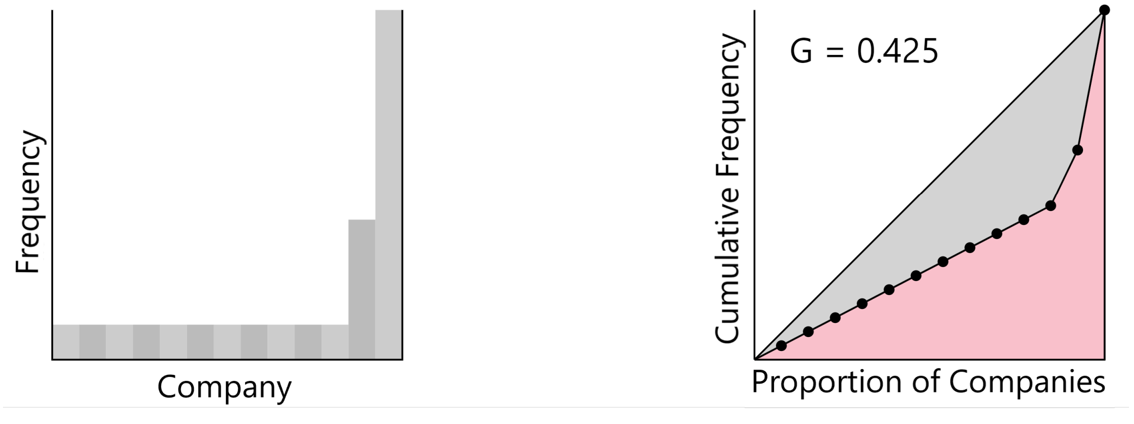


Figure 3.4: Graphs of company distribution when search term was ‘Engineer’

Next, we ran the search for a query that was a mix of technical and soft skills - tutoring.

The following table (Table 3.5) shows the frequency distribution for companies appearing in the top page of results when the search term was ‘Tutor’.

Table 3.5: Company distribution when search term was ‘Tutor’

Company Name	**Frequency**
BookNook	11
Elite Medical tutor	1
Blackmon Tutoring	2
Learning Tree Tutors LLC	1
LogicPrep	1
Inspiring Ivy Educators	1
Acadomia Tutoring US	1
Tutor Doctor of North Jersey and Rockland	1
Varsity Tutors, a Nerdy Company	1
Tutor.com	1
Study Edge	1
Tutor Doctor Goodyear Buckeye	1
The Tutoring Center, Miramar-Pembroke Pines	1
Ballantyne Reading Academy	1

The plots below (Figure 3.5) are a visual representation of the distribution in the table above (Table 3.5). The plot on the left shows how frequently each company appeared and the plot on the right displays this with a Lorenz curve to show the calculation of the Gini coefficient for the distribution.

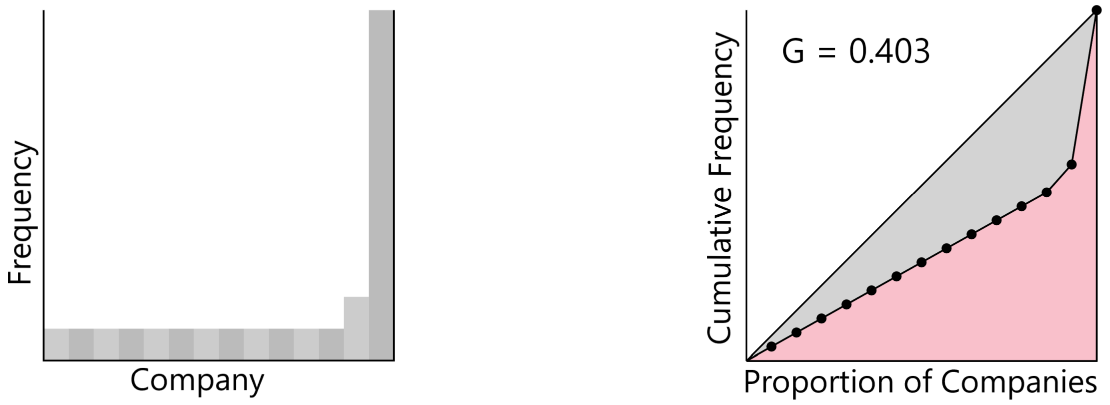


Figure 3.5: Graphs of company distribution when search term was ‘Tutor’

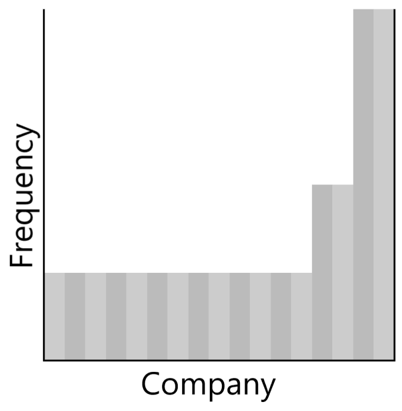
As can be seen, the Gini coefficients for the above two distributions were fairly close (0.425 for Engineer and 0.403 for Tutor). This is also evident looking at the tables of distributions, as both queries resulted in a single company appearing 10-11 times, another company appearing a few times, and all other companies appearing once in the search results. As these coefficients are pretty far from 0, which represents perfect equality, they reflect that there is indeed some inequality in the way companies’ postings appear in the results of a LinkedIn search.

Finally, we searched using a fairly vague and widely required soft skill - responsibility. We noted that LinkedIn tried finding the term in the job titles themselves (instead of in the descriptions) and gave us several results for positions related to corporate and social responsibility. The data in the table below (Table 3.6) shows the frequency distribution for companies appearing in the top page of results when the search term was ‘Responsibility’.

The graphs below the table (Figure 3.6) visually represent the distribution of companies for this search query. The plot on the left displays how frequently each company appears, while the plot on the right maps the frequencies to a Lorenz curve to show how the Gini coefficient is calculated.

Table 3.6: Company distribution when search term was ‘Responsibility’

Company Name	Frequency
CVS Health	4
Commonwealth of Massachusetts	1
Hertz	1
Campbell Soup Company	1
Rakuten Americas	2
Amazon	1
JPMorgan Chase & Co.	4
Orlando Magic NBA Team	2
Major League Soccer	1
Mr. Cooper	1
The Walt Disney Company	1
Amalgamated Bank	1
NBCUniversal	1
CBRE	1
Highmark Inc.	1
BSE Global	1
Target	1



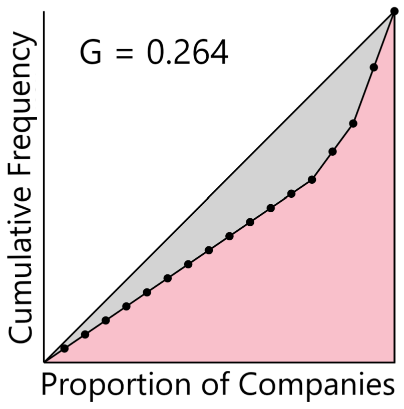


Figure 3.6: Graphs of company distribution when search term was ‘Responsibility’

We note from the above table and figure that this query results in a Gini coefficient of 0.264, which is quite less than those of the other sample queries and relatively closer to 0. This shows that there is some inequality, but the results are more evenly distributed among different companies in this search than in the previous two. The frequencies in the above table (Table 3.6) also reflect this, as they are not too far apart from each other, and there aren't any outliers.

We applied a similar approach to calculate the Gini coefficients for the company distributions resulting from various search queries. We split up our search terms into technical and soft skills to see if there would be any significant differences between the two. The following figure (Figure 3.7) shows the Gini coefficients of various search queries relating to soft skills.

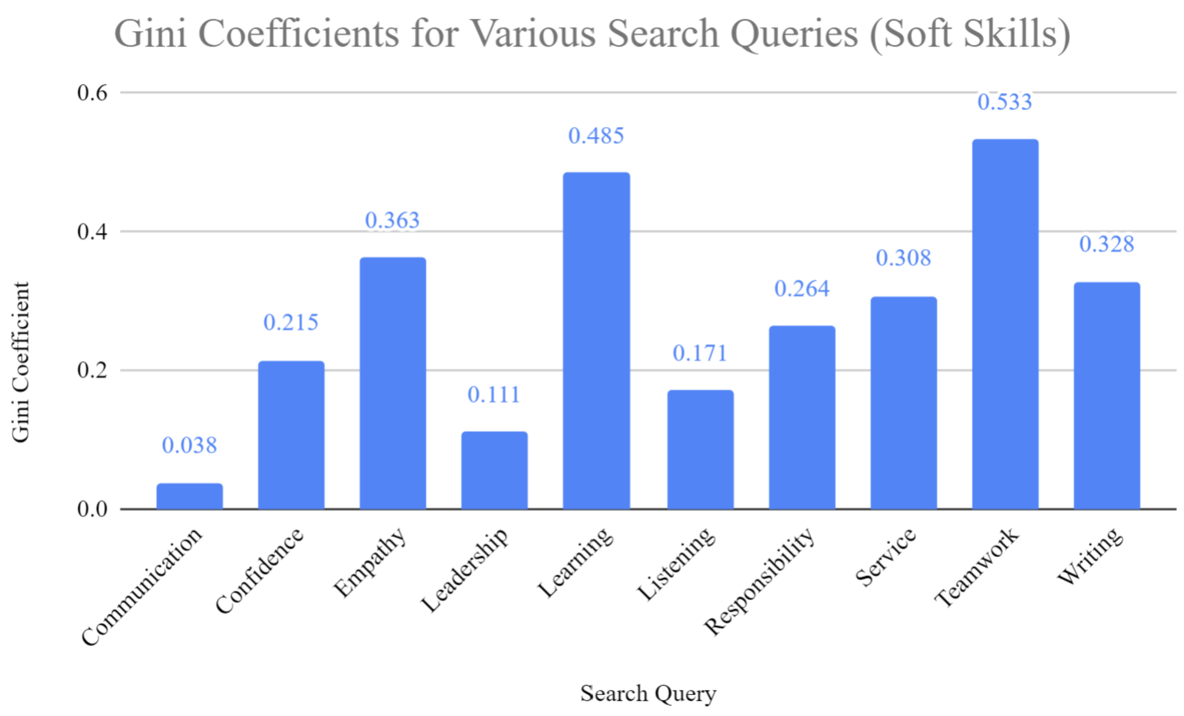


Figure 3.7: Gini coefficients for soft skill searches

We see in the chart above (Figure 3.7) that there is a fairly large variation in the Gini coefficients for different search terms. The lowest coefficient is for the term 'communication,' indicating that the results are almost evenly distributed among companies, with no company having the advantage of significantly more listings in the top 25 results. The highest coefficient can be seen for the term 'teamwork,' which shows that a single company accounted for many of the search results. There could be several reasons behind why some terms appear to have more unequal distributions than others, but a likely explanation is that terms with lower Gini coefficients, such as ‘communication’ or ‘leadership,’ appear more frequently in job descriptions for several companies, resulting in a more even set of results. On the other hand, terms with higher Gini coefficients, such as ‘empathy,’ ‘learning,’ or ‘teamwork,’ may tend to appear less commonly in job descriptions, giving higher priority to companies that do use these terms in their job postings.

We performed our experiments with technical search terms as well, the outcomes of which can be seen in the chart below (Figure 3.8).

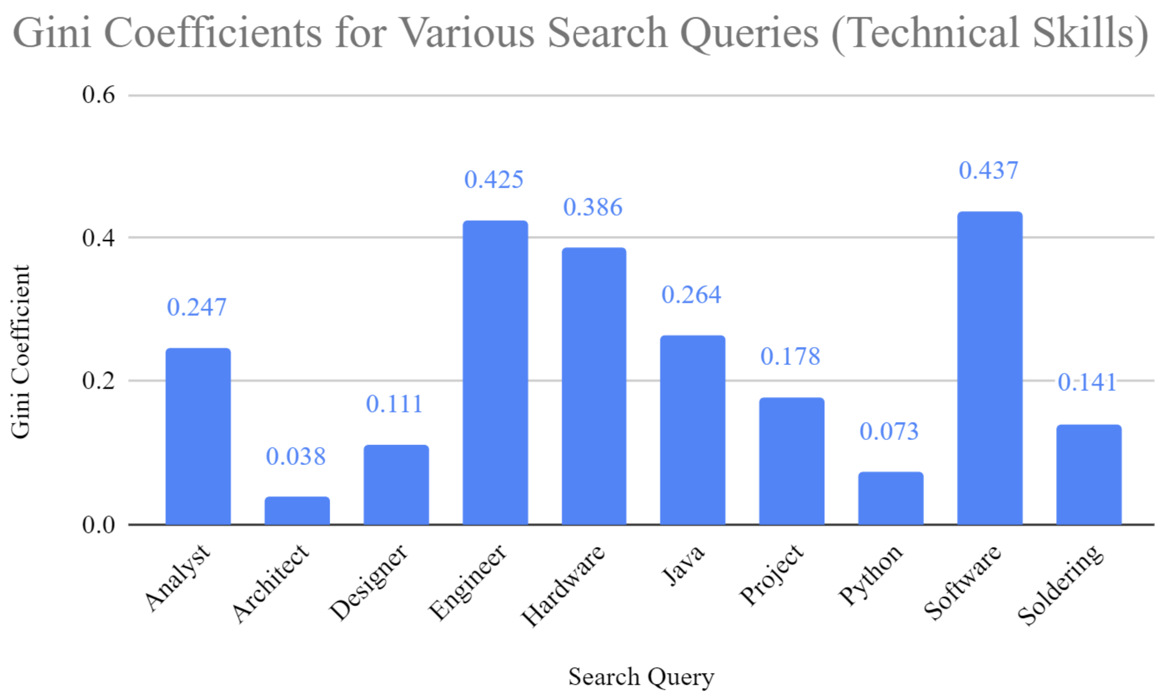


Figure 3.8: Gini coefficients for technical skill searches

As the graph (Figure 3.8) shows, the Gini coefficients for technical search queries also vary, but they are slightly lower than those for soft skills, which indicates a little bit more equality in these results. Terms like ‘architect’ and ‘designer’ have low coefficients, which makes sense as these are pretty broad and can be found in a variety of fields and companies that are reflected in the search results. Contrarily, terms like ‘software’ and ‘engineer’ have higher coefficients, showing that perhaps a single company has several postings seeking people with these skills. Though these terms could be expected to be found in many areas/companies, it seems like the distribution of the results does not reflect this, revealing a potential bias.

4. CONCLUSION

4.1 Significant Findings

In this work, we looked at various types of job search queries on LinkedIn, both manually and automatically. By writing some code to perform web scraping, we were able to come up with a general framework for the collection and analysis of this data. Our framework allowed us to collect various details from each job posting, including but not limited to the job title, company name, location, list date, and benefits. We were then able to analyze what we observed and try to make sense of why the results were displayed the way they were. Specifically, we were able to observe the inequality among job postings from different companies, with some being more heavily represented in LinkedIn's search results than others. This was a metric that we decided to measure more closely by using the Gini coefficient to compare distributions.

Although the focus of our work changed as we made observations, we noticed several interesting patterns in the way that the LinkedIn job search algorithm presented results. Firstly, there was a lot of usage of flags such as 'Actively Hiring' or 'Be an early applicant,' especially in recently posted or top jobs, which could mean that the algorithm was giving preference to those jobs that were either very new or had only a few candidates apply. We also saw that it was not unlikely for the results to be somewhat unrelated to the search query. The locations were varied, and we often noticed that several postings were from the same company, just in different locations. These findings helped us more clearly see some of the issues that may exist in the job recommendation algorithm and pushed us to look into ways we may be able to help reduce them.

4.2 Future Applications

Our framework can be scaled to collect information for more pages of search results, different queries, and even more data from each result. This could hopefully be expanded to gain an even deeper understanding of how LinkedIn's search and recommendation algorithms work. Our observations and analysis could also be used to potentially better the way that the job search algorithm displays results on LinkedIn. For example, the recommendation system could account for how many postings a company has and distribute the presentation of results more evenly. This way, job seekers would be able to easily see a wider variety of job postings, not just repeated ones from the same company. This would help make the platform more useful for job candidates. An even distribution would also allow job seekers to explore different roles related to their search terms, as well as various locations and companies.

Overall, our work is significant as social networking is becoming more and more widespread. It is important to understand how networking and recommendation platforms work so that we can make them even better and more useful going forward. Furthermore, we found in our experiments that many job recommendations show high degrees of concentration with respect to companies, and we are interested in building on this work to explore inequality with respect to location, skills, candidate gender, and other important factors. We hope that by better understanding how current recommendation algorithms work and what issues may exist within them, we may be able to help improve the online job search and professional networking experience for people around the world.

REFERENCES

[1] P. Liu, K. Shivaram, A. Culotta, M. A. Shapiro, and M. Bilgic, "The interaction between political typology and filter bubbles in news recommendation algorithms," *Proceedings of the Web Conference 2021*, 2021.

[2] S. Li, "How did we build book recommender systems in an hour part 2 - k nearest neighbors and matrix." Web, September 2017.

[3] K. Liao, "Prototyping a recommender system step by step part 1: Knn item-based collaborative filtering." Web, November 2018.

[4] L. Wu, S. Shah, S. Choi, M. Tiwari, and C. Posse, "The browsemaps: Collaborative filtering at linkedin," October 2014.

[5] S. C. Geyik, Q. Guo, B. Hu, C. Ozcaglar, K. Thakkar, X. Wu, and K. Kenthapadi, "Talent search and recommendation systems at linkedin: Practical challenges and lessons learned," September 2018.

[6] J. Ramzai, "Clearly explained: Gini coefficient and lorenz curve." Web, April 2020.

[7] OECD, "Gini index." Web, August 2002.

[8] B. Shlegeris, "Gini coefficient calculator." Web.