University of Colorado **Boulder**

Job Recommendation System Using Content-Based Filtering

By

Jagadeesh Nadimpalli
Saketh Reddy Dodda
Navya Chalamalasetty

ABSTRACT

"In the realm of technological innovation, harnessing the power of data to connect individuals with opportunities is not just a task, but a transformative journey." This statement encapsulates the essence of our project to revolutionize the job search experience through a job recommendation system. Our approach, rooted in the principles of content-based filtering, represents a leap towards a more intuitive and user-centric job matching process. This project aims to meet this requirement through the creation of an advanced job recommendation system that utilizes sophisticated Natural Language Processing (NLP) techniques and the revolutionary Word2Vec model. Our technology employs a distinctive approach known as content-based filtering, in which we carefully extract and analyze talents from user resumes. This method guarantees a job matching procedure that is tailored and pertinent to each individual. The foundation of our system relies on the use of the Word2Vec model, which proficiently transforms textual data into vector format. This shift enables a more subtle comprehension of user talents and job prerequisites, permitting a very precise matching procedure. The cosine similarity algorithm is the key component of our recommendation engine. This approach allows us to objectively evaluate the degree of compatibility between users' abilities and job criteria. Moreover, we are pioneering an additional feature aimed at users keen on professional growth. Recognizing the dynamic nature of career paths, our system also suggests job opportunities that align with users' aspirations by acquiring new skills. Our project stands as a testament to the innovative application of NLP and machine learning techniques in addressing real-world challenges. It exemplifies how technological advancements can be harnessed to create systems that are not only efficient but also adaptive to the ever-changing landscape of employment and skill development.

Table of Contents

List of Figures

List of Figures

1. Introduction

As the world gradually recovers from the impacts of a global pandemic, the landscape of employment and business is undergoing a significant transformation. This period of recovery presents a crucial opportunity for businesses to re-initiate their operations by investing in human resources. The pandemic has led to a paradigm shift in how companies operate, with many transitioning to remote work while others, in a way to reduce costs, had to make the difficult decision of letting go of employees, both permanent and contractual. This upheaval has left numerous individuals in a state of professional uncertainty, eagerly searching for new opportunities to reestablish their career paths. The notion of employment is profoundly embedded in the human psyche.

According to Stillman's findings in 2019, a job serves not just as a source of money but also as a source of meaning and identity for individuals. This perpetuates an unending loop of job hunting, when individuals consistently strive to enhance their professional status, frequently resulting in their shift from one position to another. The continuous movement of this process serves as the foundation of the employment process in the contemporary world. As a result of this continuous pattern of work changes, there has been a significant increase in the number of job boards and employment platforms. These platforms have the objective of simplifying the process of searching for employment by providing a variety of services like resume planning, profile development, and, most importantly, job suggestions. In a market flooded with options, job seekers naturally gravitate towards platforms that not only list job opportunities but also provide a more tailored and insightful job-search experience. Job seekers typically go towards platforms that not only list job possibilities but also give a more targeted and intelligent job-search experience in a market overloaded with options. This preference highlights the need of systems that can intelligently match individuals with work possibilities that match their talents, experiences, and career goals.

In this context, our project stands out as an innovative approach. By establishing a job recommendation system that goes beyond the standard job board approach, we hope to improve the job-seeking experience. Our solution promises a more personalized and effective job matching process by leveraging innovative content-based filtering techniques, precisely fitting with the increasing needs of job seekers in the post-pandemic environment. Also, our system is more than just a technological solution; it is a commitment to creating a future where the process of finding a job is as fulfilling as the job itself. By harnessing the power of content-based filtering and a deep understanding of the job market, we are poised to make a significant impact in the way individuals engage with their professional futures.

1.1 What is a Recommendation System?

A recommendation system is a sophisticated technology that sorts through massive quantities of data to propose items that are most relevant to a user's taste or need. It works by analyzing trends in user data and using algorithms to predict preferences, thereby acting as a personal guide in navigating an overwhelming amount of options. These systems, which are used in e-commerce and media streaming services, improve user experience by personalizing content, such as recommending items, movies, or music. They adopt this approach in the context of job searching to match individuals with suitable work prospects, transforming how professionals engage with potential employers.

1.2 Why do we use a Recommendation System?

Recommender systems have evolved as a critical tool in modern business, with the primary goal of improving customer experience and revenue growth. The old strategy of upselling, in which consumers are given new items based on their present choices, has developed into a more sophisticated, data-driven approach in the digital era. Businesses may develop extremely customized suggestions by analyzing customer interactions and purchase trends, effectively translating big data into a strong tool for customer engagement.

The streaming service business, particularly Netflix, provides a clear illustration of the success of recommender systems. Netflix's recommendation algorithm accounts for a sizable amount of its viewing. According to Business Insider Australia, while direct searches account for just roughly 20% of video views on Netflix, their recommendation algorithm generates an astounding 80% of views. This sharp contrast emphasizes the importance of recommender systems in influencing user behavior and preferences.

1.3 Problem Statement

For this project, our primary dataset originates from a comprehensive compilation of user resumes, representing a diverse range of skills and experiences. This dataset is complemented by job postings data, which was meticulously gathered using the Octa parse tool from LinkedIn. Indeed after initial challenges with web scraping using Beautiful Soup. Our research revolves around the central question: **"Can we create an advanced job recommendation system that not only aligns job seekers with roles matching their skills and job preferences but also effectively tackles the suggestion of potential job opportunities for users looking to acquire new skills ?"**. To answer this research question, below are the objectives that need to be satisfied with going forward.

1.4 Research Aims and Objectives

1. Develop an advanced job recommendation system using content-based filtering.
2. Refine Skill Extraction from Resumes using Spacy’s ‘matcher’ and a ‘PdfReader’
3. Utilize NLP and machine learning techniques including Word2Vec and Cosine Similarity
4. Algorithm for accurate job matching based on skill sets and location preferences.
5. Evaluate System Performance using metrics like precision and test adaptability.

1.5 Recommendation Systems

Recommendation systems are software applications that provide personalized suggestions to users, supporting them in discovering relevant objects or information, such as movies, products, or articles, based on their preferences and behaviors. These systems use a variety of strategies, including collaborative filtering, content-based filtering, and hybrid filtering, to produce accurate and engaging recommendations.

1. **Collaborative Filtering:**
   Collaborative filtering is a recommendation technique that is based on user interactions and behavior. It examines data from user-item interactions, such as ratings, reviews, and purchase history, to detect trends and commonalities among users[3]. It recommends goods that similar users have enjoyed by locating users with similar preferences. Collaborative filtering can be divided into two types: user-based and item-based techniques.

2. **Content-based Filtering:**
   Content-based filtering utilizes the characteristics or content of objects and the user's preferences to suggest items to users. The process involves creating a distinct profile for every user and item, considering factors such as genres, keywords, or metadata. Recommendations are generated by comparing user profiles with item profiles, and proposing items that possess comparable features to those that the user has shown interest in. Content-based filtering is a highly efficient method for recommending items based on their distinct qualities or traits[2].

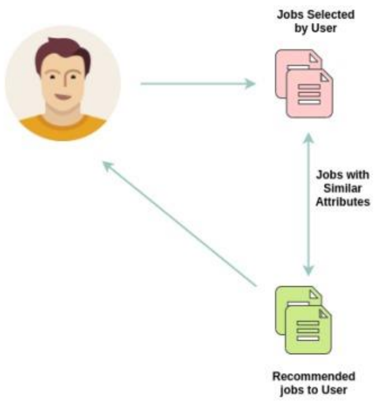

Fig.1 Content-Based Filtering

3. **Hybrid Filtering:**
   Hybrid filtering is a method that integrates many recommendation approaches in order to improve the accuracy and scope of recommendations. It combines collaborative filtering with content-based filtering to overcome their own limitations[1]. Hybrid systems can be implemented by several methods, including the fusion of recommendations from both methodologies, the utilization of one method to enhance the other, or the integration of additional techniques such as matrix factorization or deep learning. Hybrid filtering provides a more extensive and resilient recommendation approach, capable of accepting diverse user preferences and item attributes.

1.6 Natural Language Processing (NLP):

In today's data-driven era, massive amounts of data are generated, including text data, from sources like social media, news, and research papers. The majority of this data is textual, frequently unstructured data. Google manages more than one trillion requests annually, while WhatsApp handles over 30 billion messages on a daily basis. Analyzing unstructured text data to gain valuable information is a difficult task, and the use of Natural Language Processing (NLP) is the key answer for text analysis.

NLP is the computer-driven interpretation of human language that allows applications such as personal assistants, text summarization, and automated captions to be created. NLP is widely used by industry titans such as Google. This technology bridges the divide between industry and academia. In the recruiting area, where user profiles and job listings contain text data, NLP can be used to measure job similarity using techniques such as word embedding and cosine similarity. These methods produce more consistent results than classic techniques such as edit distance or lexical overlap, making NLP an important component in a wide range of industrial applications such as query search and text summarization.

1.7 Cosine Similarity:

The cosine similarity is a mathematical measure of how similar two non-zero vectors are in an area with more than one dimension. It is commonly used in natural language processing (NLP) and information retrieval. It figures out the cosine of the angle between the vectors. A smaller angle means the vectors are more alike. The cosine similarity method is used in NLP to find out how similar the two articles, sentences, or words are when they are shown as vectors in a vector space model, like Word2Vec or TF-IDF. It's great for comparing text data and is used a lot in jobs like document retrieval, recommendation systems, and clustering. The cosine similarity number is between -1 and 1, with 1 meaning that the vectors are exactly the same (perfect similarity), 0 meaning that they are not similar at all, and -1 meaning that they are completely different.

The formula for calculating cosine similarity between two vectors A and B is as follows:

$$\text{Cosine Similarity}(A, B) = (A \cdot B) / (\|A\| * \|B\|)$$

Where:

- $(A \cdot B)$ represents the dot product of vectors A and B.
- $\|A\|$ represents the Euclidean norm (magnitude) of vector A.
- $\|B\|$ represents the Euclidean norm (magnitude) of vector B.

After doing an extensive research on the aforementioned concepts and methods, we collectively decided to omit collaborative filtering from our project, where this decision is in accordance with the project’s objective. Collaborative filtering requires data on user-user interaction, such as user behavioral data, and item-item interaction data, like the number of users applying for particular jobs. Unfortunately, we do not acquire this type of data, specifically because of the absence of job ratings. That’s the reason for choosing content-based filtering for implementation. This approach involves utilizing user’s resumes as input and considering various user attributes, including skills and job preferences, to recommend jobs. Our recommendation process depends on profiles with a high similarity score, which is evaluated using cosine similarity. This allows us to suggest job matches to users that are suitable for them.

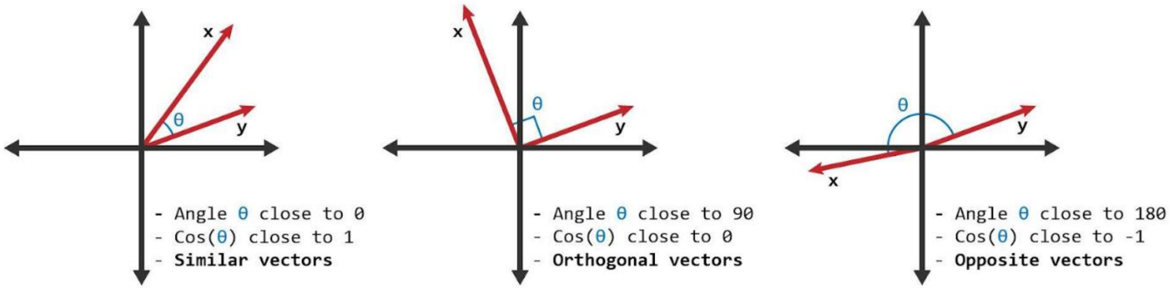

Fig 2. Understanding Cosine Similarity

2. DATA DESCRIPTION

Our project utilizes two distinct datasets: job posting data and user data. The job posting data comprises various job descriptions and requirements, while the user data is derived from the skills extracted from users' resumes. These datasets are fundamental to our job recommendation system, enabling personalized job matching.

Our Job posting dataset consists of 3,684 entries and includes the following characteristics:

job_title: The title of the job listing. Examples include "data scientist bf", "data science intern, analytics", and "assistant data scientist".
job_link: A URL link to the job listing. These links appear to be from LinkedIn.

company: The name of the company offering the job. Examples include "inspyr solutions", "discord", and "fenway group".

company_link: A URL link to the company's profile on LinkedIn.

job_location: The location where the job is based. Examples include "vienna, va", "san francisco, ca", and "southlake, tx".

job_description: A detailed description of the job. This column includes various details like job responsibilities, qualifications, and other relevant information.

seniority_level: The level of seniority required for the job. Examples include "entry level", "internship", "not applicable", and "associate".

employment_type: The type of employment for the job, such as "full-time", "internship", or "contract".

job_function: The primary function of the job. In this dataset, all examples fall under "engineering and information technology".

industries: The industry sector the job belongs to. Examples include "staffing and recruiting", "software development", "IT services and IT consulting", and "insurance".

And, our User skills data consists of skills that are extracted from their respective resumes adn stored in a dataframe.

3. METHODOLOGY

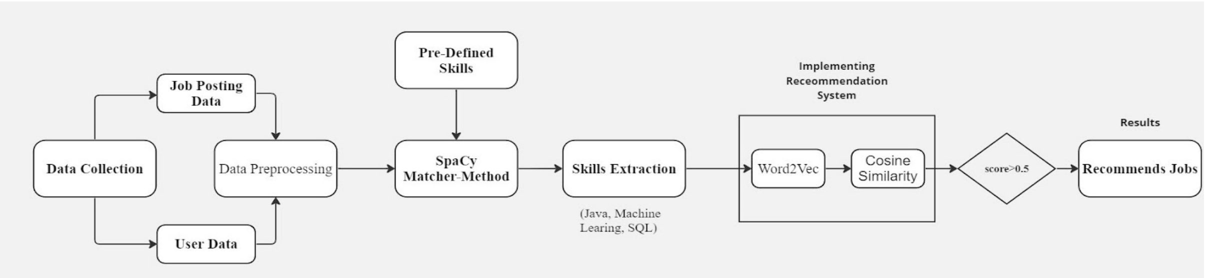

Fig 3. Flowchart of System Architecture

3.1 Data Collection

3.1.1 Web Scraping Job posting Data

During the data collection phase of the project, we faced some initial difficulties with our approach to collecting job posting data. Our original method included extracting job data from LinkedIn using web scraping techniques, namely the Beautiful Soup package. However, due to LinkedIn's data scraping constraints, this methodology was not practicable, prompting us to investigate alternate approaches.

As a more effective approach for our data collecting needs, we chose Octoparse, a powerful web scraping tool. We successfully retrieved a large dataset of job posts using Octoparse, including a wide range of fields and roles. This dataset, which had roughly 3700 entries, served as the basis for our job recommendation system. The Octoparse platform provided a seamless and efficient means of data extraction, allowing us to overcome the initial obstacles and gather the necessary data for our project. We stored this data in a dataframe for the implementation.

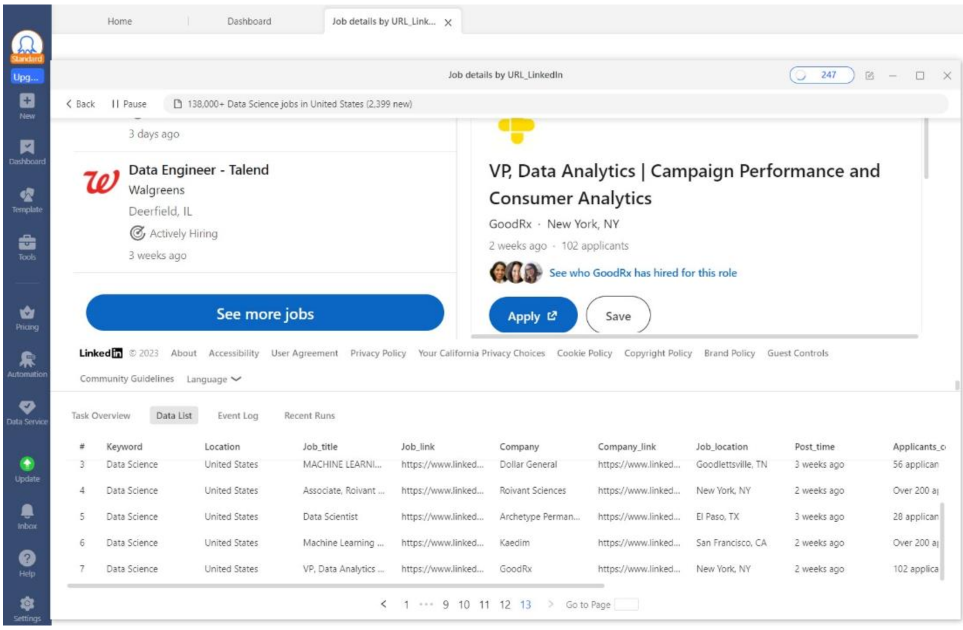

Fig 4. Octoparse tool

This shift in our data collection process was crucial in overcoming the obstacles provided by LinkedIn's scraping limits and ensuring that we have a robust dataset to work with for our recommender system.

3.1.2 User Data collection

The next phase of data collection involved assembling user data, crucial for personalizing the job recommendation system. Initially, we compiled a comprehensive list of hundreds of predefined skills. This list served as a reference for extracting relevant skills from individual user resumes. To achieve this, we employed advanced Natural Language Processing (NLP) techniques. These techniques were applied to analyze the content of users' resumes and accurately identify the predefined skills present in them. This process not only enhanced the accuracy of our recommender system but also ensured that the job recommendations were closely aligned with the users' actual skills and experiences.

3.2 Data preprocessing

The data preprocessing stage of our project involved meticulous organization and refinement of job data. Initially, we faced the challenge of handling datasets with varied fields from Octaparse tool. To streamline this, we merged these diverse datasets into a single Excel file for uniformity and ease of analysis. In the process of consolidation, we also standardized the data by converting all text to lowercase and selectively removed irrelevant columns. This step was critical in ensuring data consistency and clarity, setting the stage for more effective data analysis and subsequent processing in our job recommendation system.

3.3 Implementation

SpaCy

We leveraged the SpaCy library for text extraction of skills from Natural Language Processing. Natural Language Processing (NLP) is a critical component of current computational linguistics and artificial intelligence (AI), and SpaCy is one of the most popular libraries in this field. SpaCy, created by Explosion AI, is intended for practical, real-world applications. It excels at large-scale data extraction tasks, with strong support for NLP tasks including tokenization, part-of-speech tagging, named entity identification, and dependency parsing. SpaCy's very efficient and rapid statistical models for many languages are one of its strengths. SpaCy uses an object-oriented approach, which makes interacting with words annotation simple. It also works well with other Python modules and frameworks, which increases its utility in data science tasks[4].

In our system, we developed a method to extract skills from user resumes using SpaCy, a powerful Natural Language Processing library. We began by creating a universal list of skills applicable across various job roles. Utilizing SpaCy's 'matcher' method, known for its pattern-based token matching capabilities, we efficiently identified skills within the resumes. We converted PDF resumes into text using a 'PdfReader' function, and then processed this text with our custom 'skill_extract' function. This function, built around SpaCy's matcher, identified skills by matching them against our predefined list. The identified skills were then compiled, forming a comprehensive dataset for further analysis in our job recommendation system. The accompanying flowchart illustrates this entire process, providing a visual representation of the sequential steps.

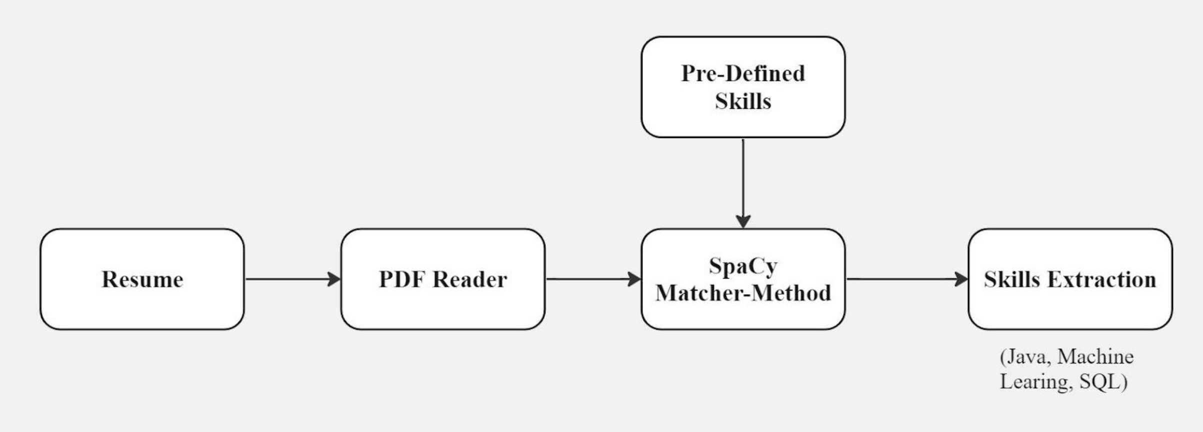

Fig 5. Implementation of SpaCy to extract skills

Later, We expanded the applicability of spaCy's NLP capabilities beyond user resumes to incorporate job descriptions in our technique. First, we used the'skill_extract' method to extract job descriptions from our cleaned dataset. We were able to precisely identify and extract relevant skills from each job. These extracted skills were then meticulously integrated to our dataframe, providing each job item with its own 'skill' column. We also focused on the users' location preferences to improve our job suggestion system's capacity to connect users with acceptable job prospects. We translated the job locations in both the job and user dataframes into vector representations using the OneHotEncoder. When offering employment prospects, this vectorization was critical in allowing our algorithm to properly evaluate both skills and location preferences.

Word2Vec

In our next steps of implementing a Job recommendation system, we utilized the Word2Vec model, a significant advancement in Natural Language Processing (NLP), to transform textual data into a format suitable for our algorithms. Recognizing the necessity for numeric inputs in algorithmic processing, we employed Word2Vec to convert individual skills into vectors. We developed a Word2Vec model with a dimensionality of 100, enabling each skill to be represented as a 100-dimensional vector. This approach effectively encapsulated the semantic meaning of each skill within these vectors.

By invoking "model.wv" returns a 100-dimensional vector reflecting the skill "python." This vector is more than just a numerical translation; it is a representation of the ability in the context of job-related skills. To create a comprehensive profile for each user and job, we computed the mean vector of their respective skill lists. This procedure generated a final vector representation for each entity, encompassing the aggregate of their skills. This vectorized representation of abilities aided the later phases of our job recommendation system, specifically in connecting users with relevant job opportunities based on their skill sets.

Cosine Similarity

In the following stage of our job recommendation system, we concentrated on matching users with job listings by utilising the power of cosine similarity, a metric used to determine the similarity of two vectors. Our method required computing cosine similarity scores for two essential features separately: the skill vectors and the location vectors.

Our matching algorithm was built on skill vectors drawn from the user's skills and the skills necessary for each job. Recognizing the relevance of skill alignment in employment suggestions, we provided the skill similarity score a considerable weight of 0.8. This weighting emphasized our system's emphasis on pairing job seekers with roles that closely match their current skill sets. In addition to skills, we also considered geographical preferences, which are important for many job searchers[5]. The cosine similarity between the user's preferred job location and the location of the job posts was computed and given a weight of 0.2. This weighting indicates the location's secondary prominence, but still significant relevance in job matching. The combined weighted skill and location similarity values were then used to compute the final job recommendation score for each job posting. The below code depicts how we achieved the cosine similarity for the skill vectors and the location vectors.

Final_Similarity_Score = (0.8*Skill_Similarity) + (0.2*Location_Similarity)

```
def similarity_vectors(sl,jl):
  similarity=[]
  my_skills_l = [i.lower().replace(' ','_') for i in sl]
  my_skills_vector=np.array(get_vector(my_skills_l)).reshape(1,-1)
  jl=jl.reshape(1,-1)
  for i in range(len(js)):
    jbs_skill_l=[j.lower().replace(' ','_') for j in js[i]]
    jbs_skills_vector=np.array(get_vector(jbs_skill_l)).reshape(1,-1)
    cs=cosine_similarity(my_skills_vector,jbs_skills_vector)[0][0]
    jc=cosine_similarity(jl,jds[i].reshape(1,-1))[0][0]
    final_similarity = 0.8*cs + 0.2*jc
    similarity.append(final_similarity)
  return similarity
```

Fig 6. Code snippet for cosine similarity score

The final job recommendation score for each job posting was then calculated by combining these weighted skill and location similarity scores. Job postings with a final score exceeding the threshold of 0.5 were deemed suitable and recommended to the users. This threshold was strategically set to ensure a balance between precision and a broad range of options, catering to diverse user preferences while maintaining relevance and quality in the recommendations.

4. Evaluation

In the evaluation phase of our project, we focused on evaluating how effective our recommender system was, which is based on cosine similarity and a user-defined threshold for job recommendations. Precision is the primary metric for evaluation here, which is a measure that shows the accuracy of the recommendations made by our system.

Precision is computed as Precision = (TP) / (TP + FP), where TP (True Positives) are the jobs that the system correctly recommends and that are also in the user's ground truth, signifying the pertinent jobs that are correctly identified. False Positives (FP jobs) are those that the system recommends but are absent from the ground truth; they are irrelevant jobs mistakenly marked as appropriate. Because the occupations the system recommends are our main emphasis, FN (False Negatives) and TN (True Negatives) are less critical in our project.

In our study, we utilized a threshold of 0.5, indicating that jobs with a similarity score over this threshold were suggested to users. For instance, we chose to recommend jobs to 4 users and below are precisions for all the users along with the system's overall precision.

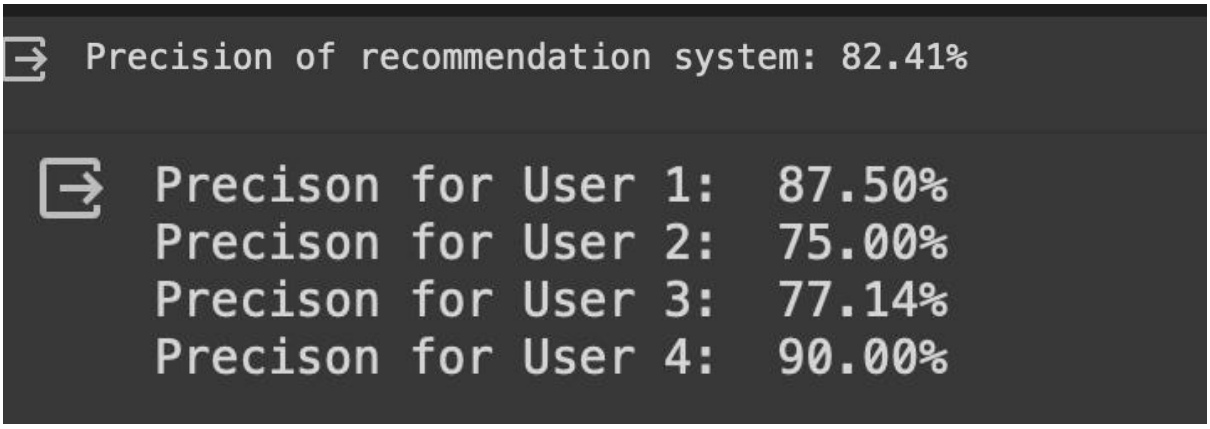

Fig 7. Precision evaluation

5. Results

The performance of our job recommendation model was evaluated by a Precision score of 82%, indicating the strong correlation between job recommendations and user's skill sets and location preferences. This high score demonstrates the model's robustness, especially in matching users with jobs that align closely with their professional profiles. Upon analyzing the results for four different users, it is clear that the model's suggestions are not limited to just the preferred locations, but also take other opportunities into account, as it primarily focuses on aligning skills. For example, User 1 indicated a preference for San Francisco, but the model also found appropriate job opportunities in nearby areas like Aurora and Foster City. The pattern remains consistent among all users, demonstrating the system's capacity to offer a wide range of geographical choices while maintaining high skill matching quality.

The underlying principle of our model is that the congruence between job requirements and the user's skills is more important than location. Therefore, the model remains adaptable in terms of location, allowing the user to have a wider range of job opportunities. This demonstrates its ability to adjust to the complex nature of job searching in today's workforce. The below images describe the recommendations that are given to a particular user based on his professional skills and preferred location.

70 rows × 4 columns

--USER: 4 RECOMMENDATAIONS --

	job_title	company	job_link	job_location
1701	software engineer (university grad)	meta	https://www.linkedin.com/jobs/view/software-en...	austin
63	senior data analyst	roku inc.	https://www.linkedin.com/jobs/view/senior-data...	austin
440	machine learning developer	mindpal	https://www.linkedin.com/jobs/view/machine-lea...	austin
3653	principal machine learning scientist	simplex	https://www.linkedin.com/jobs/view/principal-m...	austin
2155	software engineer - ai	ask consulting	https://www.linkedin.com/jobs/view/software-en...	austin
3568	vp, data science	the knot worldwide	https://www.linkedin.com/jobs/view/vp-data-sci...	austin
1738	software engineer ii (remote)	the home depot	https://www.linkedin.com/jobs/view/software-en...	austin
3631	data analyst	actalent	https://www.linkedin.com/jobs/view/data-analys...	sunnyvale

Fig 8. Job recommendations for one User

--USER: 2 EXTRA RECOMMENDATIONS --

	job_title	company	job_link	job_location
1703	software engineer, intern/co-op	meta	https://www.linkedin.com/jobs/view/software-en...	new york
106	customer facing data scientist	explorium	https://www.linkedin.com/jobs/view/customer-fa...	new york
1895	software engineer (ny remote)	prelim	https://www.linkedin.com/jobs/view/software-en...	new york
1898	software engineer (ny remote)	prelim	https://www.linkedin.com/jobs/view/software-en...	new york
955	devops engineer	tree top staffing llc	https://www.linkedin.com/jobs/view/devops-engi...	new york
80	data scientist	paritas recruitment	https://www.linkedin.com/jobs/view/data-scient...	new york
2569	data software engineer - python & sql	inspyr solutions	https://www.linkedin.com/jobs/view/data-softwa...	new york
2632	software quality engineer intern	s&p global	https://www.linkedin.com/jobs/view/software-qu...	new york
3061	quantitative researcher, cubist portfolio rese...	point72	https://www.linkedin.com/jobs/view/quantitativ...	new york
1327	senior devops engineer	tcgplayer	https://www.linkedin.com/jobs/view/senior-devo...	new york
804	lead devops engineer	j.crew	https://www.linkedin.com/jobs/view/lead-devops...	new york
808	lead devops engineer	madewell	https://www.linkedin.com/jobs/view/lead-devops...	new york
1751	web developer intern	eulerity	https://www.linkedin.com/jobs/view/web-develop...	new york
1356	founding devops/sre engineer	rogo	https://www.linkedin.com/jobs/view/founding-de...	new york
228	machine learning engineer	apollo solutions	https://www.linkedin.com/jobs/view/machine-lea...	new york
3411	market research intern	retrain.ai	https://www.linkedin.com/jobs/view/market-rese...	new york
164	data science engineer	garner health	https://www.linkedin.com/jobs/view/data-scienc...	new york

Fig 9. Job recommendations for another User

We have documented a unique facet of our recommendation system which advocates for professional growth. Acknowledging the aspirational nature of job seekers, our system is calibrated to propose extended job opportunities contingent upon the user's willingness to acquire additional skills. This adaptive recommendation mechanism is designed to not only cater to the present competencies of the users but also to envision and support their potential for skill enhancement. The system is designed in a way that the model asks the user if they are willing to learn some new skills. If yes, it is gonna suggest some skills to them along with the related job to those additional skills. For instance, if a user, such as User 1, expresses an interest in acquiring skills like 'Azure,' 'Containerization,' 'Software Development Life Cycle,' 'Unix,' and more, our system responds by recommending additional job opportunities in locations that align with the user's preferences. As a result, we enhance the user experience by offering job suggestions that are not only skill-matched but also location-relevant. The below image is the output we acquired for suggesting the extra recommendations if they are willing to learn a few more skills.

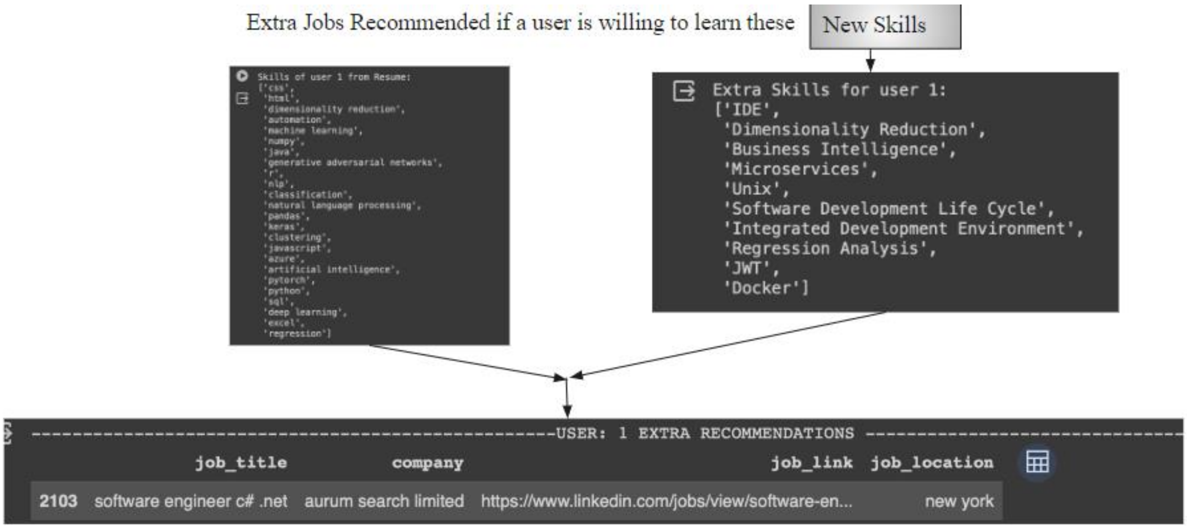

Fig 10. Additional Job Recommendations for User 1

6. Conclusion

Ultimately, this project has successfully created an advanced job recommendation system by utilizing content-based filtering, effectively resolving the issue of insufficient user-job interaction data. The system effectively extracts and analyzes skills from user resumes using a combination of Spacy's 'matcher' method and a proprietary 'PdfReader' function, which converts PDF resumes into text that can be analyzed. Our methodology, which involves gathering, refining, and analyzing data using advanced techniques such as Word2Vec and cosine similarity calculations, resulted in a strong model that can provide job recommendations based on a carefully calibrated combination of skill and location preferences. The technology showcased its adaptability to users who are eager to enhance their skill set by achieving an impressive 82% precision in job matching. This experiment demonstrates the effectiveness of content-based filtering in job recommendation and also paves the way for future improvements in personalized job matching technology.

7. Limitations & Future Work

In the future, integrating collaborative filtering and hybrid methods could be the main focus for improvements to our job recommendation system. By integrating collaborative filtering with the existing content-based filtering, the model's accuracy can be enhanced through the usage of user-job interaction data. This methodology incorporates user behaviors and preferences, providing a highly personalized recommendation interface. An amalgamation of both methodologies in a hybrid model would result in an adaptable system capable of effectively managing different user requirements. Furthermore, it is crucial to enhance the existing Word2Vec skill representation by constructing a collection of words that is specific to the domain. By customizing this set of words to include IT skills, technical terms, and job-related terminology, the system's accuracy in interpreting job descriptions and user profiles would be improved. The development of a highly specialized corpus can enhance the accuracy of recommendations by comprehensively grasping the linguistic complexities specific to the IT industry.

Another important area for future study involves creating a dynamic recommendation feature that adjusts to changing user preferences. By analyzing data that includes previous user-job interactions, the system can suggest new jobs as the user's preferences evolve. In addition to skill and location matching, the system has the capability to include other factors such as cultural integration, company size preferences, and industry-specific requirements. This enhances the job matching process by making it more comprehensive. Integrating a feedback loop that allows users to provide input on recommended jobs would optimize the recommendation algorithm, improving its precision and pertinence. Ensuring optimal performance of the system as it expands is vital, as it must efficiently handle larger datasets while maintaining speed and accuracy. Expanding the system to support multiple languages and adapt to different cultural contexts would enhance its applicability and reach in the global job market, making it a highly versatile tool in the constantly changing field of job recommendations.

Our job recommendation project, although creative, encounters certain constraints. Although content-based filtering is strong, it might ignore the depth of user interaction data, which could impact the modification of recommendations. Relying on the precision of resumes submitted by users may result in inaccurate matches. Additionally, the implementation of a uniform set of skills may fail to account for the particulars of specialized positions, potentially leading to mismatches. The model's static structure may not accurately represent the constantly evolving job market, and difficulties in scaling may arise as the amount of data increases.

References

[1] Bogdan Walek, Vladimir Fojtik (2020). *A hybrid recommender system for recommending relevant movies using an expert system.* https://www.sciencedirect.com/science/article/abs/pii/S0957417420302761

[2] Tanya V. Yadalam; Vaishnavi M. Gowda; Vanditha Shiva Kumar; Disha Girish; Namratha M (2020). *Career Recommendation Systems using Content based Filtering.* https://ieeexplore.ieee.org/document/9137992

[3] Yingya Zhang, Cheng Yang, Zhixiang Niu (2015) *A Research of Job Recommendation System Based on Collaborative Filtering.* https://researchgate.net/publication/281653213_A_Research_of_Job_Recommendation_System_Based_on_Collaborative_Filtering

[4] M Viswa Murali, T G Vishnu, Nancy Victor [2019] - *A Collaborative Filtering based Recommender System for Suggesting New Trends in Any Domain of Research.* - https://ieeexplore.ieee.org/document/8728409

[5] Daniel J. Wang (2022), *Building a Job Recommender for Non-Technical Business Roles via NLP and Machine Learning* https://towardsdatascience.com/building-a-job-recommender-for-non-technical-business-roles-via-nlp-and-machine-learning-626c4039931e

URL for the YouTube video presentation - https://youtu.be/lyJ3enl6f28