Combined Study of Various Techniques for Personalized Job Recommendation

Richa Dwivedi
Dept. of CSE
IIIT-Delhi
Delhi, India
richa20104@iiitd.ac.in

Drishti De
Dept. of CSE
IIIT-Delhi
Delhi, India
drishti20075@iiitd.ac.in

Nidhi Allwani
Dept. of CSE
IIIT-Delhi
Delhi, India
nidhi20034@iiitd.ac.in

***Abstract*—With the problem of unemployment and employee churn advancing at higher rates in our world of today, there is an increase in demand for online resources for efficient job search. Also, different jobs require different skill-sets from their user candidates in order to get hired for the concerned job. In this paper, we perform our analysis on the similarity techniques for Job Recommendation Systems based on the researches done in the field of Job Recommendations. In our implementation, we have used three similarity measures: Tanimoto, Cosine (Orchini), and City Block similarity. These techniques have been tested on a new Job Recommendation Systems Dataset taken from Kaggle. We have also analyzed the performance of other similarity techniques involving other distance measures such as Euclidean distance. The performance of these similarity score-based techniques for generating the highest score-based recommendations is assessed using different evaluation metrics such as Accuracy, Precision, Recall, and F1-score respectively.**

***Index Terms*—Job Recommendation, Tanimoto, Jaccard, City Block, Manhattan, Cosine, Orchini, similarity measures, vectorization, preprocessing, accuracy**

I. Introduction

Job Recommendation Systems are used to provide users with the recommendation of relevant jobs based upon their skills. Job Recommendation Systems aims to help users make the job hunt easier as some users need new jobs or leave their previous jobs due to reasons such as employee churn faced by different organizations. Employee churn in companies happens due to:

- Lack of Growth in the company
- Lack of Recognition
- Lack of proper guidance and feedback
- Feeling Overworked
- Very less Decision-Making Opportunities

Studies regarding the rate of online job search show that roughly one-third of the recent job seekers claim that the internet served to be the essential resource for their most recent employment search, that is, out of the 79% resources used, which were online resources and information, maximum users have reported it to be an essential resource for job searches.

The booming demand for Recommendation systems comes from the creation of tons of journals, research articles, movies, books, music day by day. This upsurge in the creation of different reading and media articles makes it difficult for any person to search for the articles and media of their choice for their relevant works in day-to-day life. Thus, the requirement of an intelligent filter that understands the user's behavior or choices and reduces the list of articles the user needs to choose from by providing relevant articles is in high demand. Furthermore, many reputable and highly successful businesses such as Netflix, LinkedIn, Amazon, etc., incorporate such recommendation systems in their websites for a better user experience.

On the other hand, in the case of Job Recommendation Systems, since the users need to find a better-suited job that reduces the chances of Employee churn and the impact of other impediments of a successful career, we need to provide a more personalized recommendation system by extracting the skills and other essential information from the user's profile in order to enhance the process of recommendation of relevant job results from different companies in the field of the user's interest.

Due to the above-stated reasons, the emphasis of our project is to analyze the performance of job recommendation systems based on the models utilizing different similarity measuring techniques such as Tanimoto or Jaccard similarity, Cosine or Orchini Similarity, City Block or Manhattan distance-based similarity, and Euclidean-based similarity. Upon assessing the performance of these similarity-based job recommendation techniques on the test dataset as a cold-start problem after splitting the dataset into train and test data, we analyze the recommendations based on two types: Top 5 recommendations and Highest similarity score based recommendations. We also assess the performance of our simplistic yet highly efficient job recommendation model by measuring similarity based on the Euclidean distance approach to understand further the concept of distance-based similarity measuring recommendation techniques.

II. Literature Review

Some of the earliest works utilizing different similarity measures concerned to generate a rather simplistic yet efficient job recommender system are as mentioned below.

In the paper, Masahira sato et al. [1] proposed the Manhattan distance as the similarity measure. In this, They used the offline and online model for recommending jobs. There are four features used in this paper. The latitude and longitude distance is used for calculating the Manhattan similarity.

In the paper, Shuo Yang et al. [2] proposed the cosine similarity measure for a job recommendation.They have used the hybrid recommendation, they combined the various collaborative filtering with content based filtering. The similarity was calculated by some scoring heuristic, such as the cosine similarity between the user profile vector and the item feature vector.

Since then, much research done in this field. In one research Vasily Leksin et al. [3] proposed Tanimoto Similarity, Cosine Similarity, in content-based recommendation system. In this, they gave a solution for the RecSys Challenge 2016. They worked on several datasets from a social network for business XING, where they have to predict job postings for a user. Item-based CF, content-based, and hybrid models are used with similarity measures.

In the paper, Xingsheng Guo et al. [4] proposed Jaccard Similarity between two jobs, Cosine Similarity between two jobs, TF-IDF Weighting schemes. In this, they gave different content-based and case-based approaches that consider similarity in features. They also used a hybrid approach by combining the above methods. Data CareerBuilder Dataset is used where the hybrid approach outperforms the other systems.

In the paper, Anika Gupta et al. [5] proposed a Combined weighted sum of cosine similarity and rules weight in collaborative filtering and associate rule mining. They gave a system for job recommendations based on the candidate's profile matching and preserved the candidate's job behavior or preferences.

In the paper, Xiao Xu et al. [6] proposed Cosine similarity in latent topic space to compute the potential interest between a new user and a post. They gave a hierarchical pairwise model with ensemble learning and integration of both content and behavior information utilizing the Hawkes Process.

III. Methodology

A. Dataset

The dataset used in this project has been taken from the Kaggle website. The name of the used dataset is ”Jobs Data for recommender systems,” It consists of a few CSV and JSON files such as ALL_Offers.CSV, offers.csv, users.csv, organisations.csv, jobs.json, and offers.json files. The data file ”all_offers” consists of 8867 records of the user's job, skills, etc., and the Organizations name, address, email, etc. The dataset consists of 252 records of the job, skills, etc., and 201 of Organizations name, address, email, etc. For users data, it contains 7 features: ['user_id', 'status', 'city', 'job_title', 'organization_id', 'contracts', 'description', 'skills'] and for organisations data, it consists of features: ['organization_id', 'company_name', 'email', 'address', 'size', 'sectors']

In total, there are 222 unique jobs acquired by different users. The number of jobs versus the occurrences of those jobs (number of users with the same job) is represented in the form of a bar graph as shown in “Fig. 1”. Also, 155 unique skills overlap in a particular fashion to match different job descriptions for different users. The number of skills overlaps refers to the skill frequency in total for all the user jobs. As seen in “Fig. 2” the number of skills with the highest job overlaps (count = 97) is 1. Therefore, there is one skill that is seen in 97 unique job offers for users.

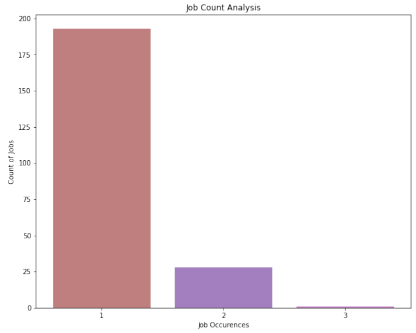

Fig. 1.
Number of Jobs vs Job Occurrences in User Profile

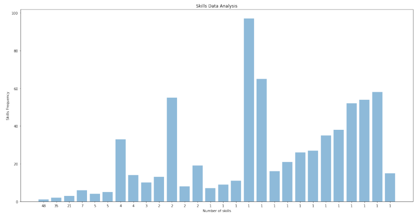

Fig. 2.
Skills Frequency in Jobs versus the number of such skills that overlap

B. Data Preprocessing

For the successful implementation of our project, two datasets were collected from the main Kaggle dataset, namely all_offers and organization data files, were combined after data cleaning and removal of missing/null values and duplicate data to explore the dataset better. After combining the dataset, the essential information regarding jobs and their respective relevant skills were taken. Since the skills for each user job were stored in the form of a dictionary, therefore, in order to extract each skill for different jobs, those dictionaries were converted to strings and then a manual function to remove all special characters such as ”{”,”:”,”,”etc. were used to extract skills relevant for each job. These skills were then used as columns, and as per the availability of each skill in different jobs, the meta-data for our project was created where if the job consisted of the concerned skill, then the value in the data frame was one else in the case of the unavailability of the skill in the job the value was kept as 0. The preprocessing steps are also mentioned in the flowchart in “Fig. 3”.

Also, in the case of cosine, city block similarity, and euclidean measures, each job-relevant skills were appended as whole space-separated strings such that upon performing vectorization on the user input skills query and the job-relevant

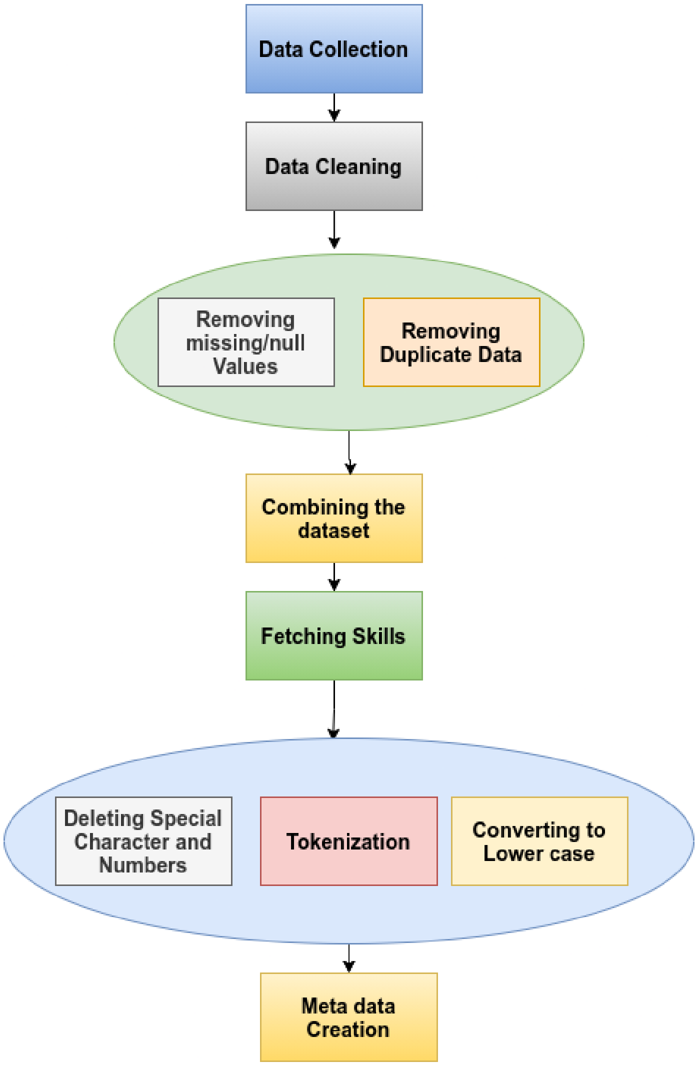

Fig. 3. Preprocessing and Meta-Data creation

skills string, both the generated vectors are compared to give the score as per the concerned similarity metrics.

C. Vectorization

Vectorization refers to the computation and generation of vectors based on the words present in a string. In our project, a function called the text_to_vector() was used to convert the input text string to its corresponding vector such that the two vectors can be compared to understand the similarity between two text strings. This is done by importing the Counter library to count all the words in the given text. Different similarity metrics such as Cosine (Orchini), City block, and Euclidean require vectorization of the skill query input and the job-based skills for searching job options where the user skills are available and then search similar jobs based on the top match.

D. Similarity Measures

The similarity between two data is a numerical measure of the degree to which the two data inputs are alike or similar.

1) Tanimoto (or Jaccard Similarity): Tanimoto similarity is a measure of similarity between two sets of data. The range of 0 to 100% and similarity between data sets is directly proportional to the percentage. In Jaccard, the similarity is calculated where the intersection of two input data is divided by their union. The formula is as shown below:

$$tanimoto(A,B) = \frac{A \bigcup B}{A \bigcap B} \tag{1}$$

$$tanimoto(A,A) = 1 \tag{2}$$

(if the value is 1 the data sets are completely similar)

$$tanimoto(A,B) = 0 \;\; if \;\; A \bigcap B = 0 \tag{3}$$

(if the value is 0 it represents the case of no similarity)

2) Cosine Similarity: It measures the similarity between two non-zero vectors of an inner product space. This similarity is measured by the cosine of the angle between the two vectors and determines whether two vectors are pointing in roughly the same direction. This similarity metric is particularly used in positive space where the outcome is neatly bounded in the range [0,1]. Now, given two vectors A and B, the cosine similarity, $cos(\Theta)$ is represented using a dot product and magnitude as:

$$similarity = cos(\Theta) = \frac{A \cdot B}{||A|| \;\; ||B||} \tag{4}$$

$$similarity = \frac{\sum_{i=1}^{n} A_i \;\; B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{5}$$

where, A_i and B_i are the components of the vectors A and B respectively.

3) City Block (or Manhattan based) similarity: This similarity metrics calculates the distance between two real-valued vectors by representing the distance between points in a city road grid. The idea behind this metric is that lower the value of the city block or Manhattan distance, the more similar the data points are. In a plane with point p_1 at (x_1,y_1) and p_2 at (x_2,y_2), the city block distance is $|x_1 - x_2| + |y_1 - y_2|$. The generalized formula for city block similarity is given as:

$$cityblock(x,y) = \sum_{i=0}^{n} |x_i - y_i| \tag{6}$$

4) Euclidean Distance based similarity: This similarity metric is done apart from the techniques being analyzed to understand the overall performance of any distance-based similarity metrics to recommend relevant and highly similar jobs to users based on their profile skills.

Euclidean distance is the measure of distance between two points p_1 at (x_1,y_1) and p_2 at (x_2,y_2) given as:

$$euclidean(p_1,p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{7}$$

E. Types of User Recommendation

The techniques mentioned earlier and their appropriate preprocessing and vectorization were incorporated as a recommendation model and tested on user test data as a cold-start problem. In this case, the new user skills were extracted and converted to a space-separated string to compare with skills present for each published job to give top recommendations of jobs to users. These top recommendations were given to users in two ways:

1) Top 5 Recommendations: Comparing the user profile skills with the job-relevant skills of each of the published jobs, we find the top 5 jobs with a high similarity (or low distance) score. These jobs may be 99.99% similar to the user skills input or come under the top 5 even with a slightly lesser similarity score than 99.99%.

2) Highest similarity score recommendation: In this case, the jobs with the highest similarity scores (around 99.99%) are recommended to the users based on their skill input extracted from the new user skill column of the test set.

IV. Results and Analysis

Based on the two types of user recommendations mentioned above, we analyze the performance of all the techniques mentioned above. The resultant jobs recommended to each new user are then checked with the job that the user is originally in as per the test dataset. If the original user job is recommended in the model result, then the model appends 1 for yes else, it appends 0 for no. This array of 0s and 1s thus received is then checked for accuracy by computing the count of 1s from the total user predictions. The performance of different techniques as per the two types of user recommendations is shown in two tables: Table 1 and Table 2 below.

TABLE I
Results Obtained from different techniques in top 5 recommendation

Techniques used	Different metrics used	
	Accuracy	***Error rate***
Tanimoto	94%	6%
City block	94%	6%
Cosine	94%	6%
Euclidean	90%	10%

TABLE II
Results Obtained from different techniques in top recommendation(Highest similarity)

Techniques used	Different metrics used				
	Accuracy	***Error rate***	***Precision***	***Recall***	***F1-Score***
Tanimoto	92%	8%	100%	92%	95.83%
City block	92%	8%	100%	92%	95.83%
Cosine	98%	2%	100%	98%	98.98%
Euclidean	92%	8%	100%	90%	94.73%

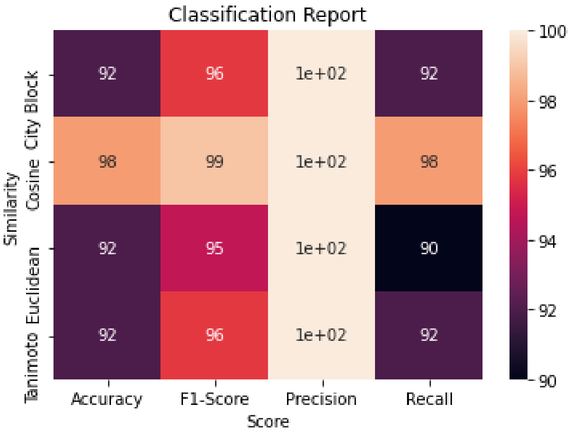

Fig. 4.
Heat map for Classification Report

Along with model analysis, we also provide a way for entering query inputs manually and checking the top recommendations for the skills entered in the query. This enables the users to select their most favoured job among the top recommended ones. ”Table-II” is converted into a Heat map plot (Fig-4) for better visualization purposes. The precision values are recorded as ”1e+02” instead of 100%.

Analyzing the performance of recommendation models based on different core similarity techniques, we observe that the error rate of Cosine similarity in the case of the Highest score-based recommendation is lesser than its counterparts. This is because cosine considers the existence of duplicate terms while computing similarity. Also, computationally, cosine has low complexity and ease over handling spare data vectors since only non-zero dimensions are considered.

Upon critically analyzing the error rate observed in the two tables, we notice the short-comings of some similarity measures upon recommending top 5 and highest-score based job recommendations as even upon achieving high similarity scores, and some users are seen to have different jobs than the ones recommended by the models, thus resulting in 6-10% error rates.

V. Conclusion

This project analyses the outcomes and implications of different similarity measures like Tanimoto (Jaccard) and Cosine. It also emphasizes the models created by measuring job and skill-based similarity from different distance-based similarity techniques. The recommendations from these similarity metrics give 90-94% accuracy but may give 6-10% inaccurate recommendations as seen using the cold-start problem in the user test set. Out of all the similarity metrics, Cosine is observed to outperform all in the highest score-based recommendation as it considers duplicity in terms present in the vectors with low-complexity. Future works in the case of Personalised Job Recommendation Systems are the utilization of the user-preferred location to get job recommendations based on jobs in organizations established in nearby areas. This can be done by extracting the latitudes and longitudes of the user-preferred location and computing the euclidean distances between the latitudes and longitudes of the organization location. This filters out other jobs that fall far from the user-preferred location and gives a more accurate job recommendation.

References

[1] M. Sato, K. Nagatani, and T. Tahara, "Exploring an optimal online model for new job recommendation: Solution for recsys challenge 2017," in *Proceedings of the Recommender Systems Challenge 2017*, 2017, pp. 1–5.

[2] S. Yang, M. Korayem, K. AlJadda, T. Grainger, and S. Natarajan, "Combining content-based and collaborative filtering for job recommendation system: A cost-sensitive statistical relational learning approach," *Knowledge-Based Systems*, vol. 136, pp. 37–45, 2017.

[3] V. Leksin and A. Ostapets, "Job recommendation based on factorization machine and topic modelling," in *Proceedings of the Recommender Systems Challenge*, 2016, pp. 1–4.

[4] X. Guo, H. Jerbi, and M. P. O'Mahony, "An analysis framework for content-based job recommendation," in *22nd International Conference on Case-Based Reasoning (ICCBR), Cork, Ireland, 29 September-01 October 2014*, 2014.

[5] A. Gupta and D. Garg, "Applying data mining techniques in job recommender system for considering candidate job preferences," in *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI).* IEEE, 2014, pp. 1458–1465.

[6] W. Xiao, X. Xu, K. Liang, J. Mao, and J. Wang, "Job recommendation with hawkes process," in *10th ACM Conference on Recommender Systems-RecSys*, vol. 16, 2016.

[7] N. D. Almalis, G. A. Tsihrintzis, N. Karagiannis, and A. D. Strati, "Fodra—a new content-based job recommendation algorithm for job seeking and recruiting," in *2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA).* IEEE, 2015, pp. 1–7.

[8] R. G. Belsare and D. Deshmukh, "Employment recommendation system using matching, collaborative filtering and content based recommendation," *Int. J. Comput. Appl. Technol. Res*, vol. 7, no. 6, pp. 215–220, 2018.

[9] Y. Zhang, C. Yang, and Z. Niu, "A research of job recommendation system based on collaborative filtering," in *2014 Seventh International Symposium on Computational Intelligence and Design*, vol. 1. IEEE, 2014, pp. 533–538.