Analysis of Data Analyst Jobs

A knowledge-based recommender system to predict jobs based on your skills.

Afzal Mukhtar (PES2201800675)
Computer Science and Engineering
PES University
Bengaluru, India
afzalmukhtar985@gmail.com

Aswin A Nair (PES2201800504)
Computer Science and Engineering
PES University
Bengaluru, India
aswinanilnair@gmail.com

Hritika Rahul Mehta (PES2201800024)
Computer Science and Engineering
PES University
Bengaluru, India
hritika2708@gmail.com

Abstract*—The e-recruitment platform is growing and becoming an important part of our lives. Such platforms still have an issue on basing their choice of inputs for recruitment. But overall, these systems are aimed at increasing accuracy and reducing the task of searching through a large number of jobs offered. Many have lost their jobs or unable to find jobs in the COVID-19 pandemic. Although there are numerous job recommender systems, here we focus on recommending jobs to candidates based on their skills. Usually, jobs that are based on experience and skills are ignored, therefore we put a little focus on the skills for jobs as this pandemic can reduce the effective work experience a candidate can have.

***Keywords*—Exploratory Data Analysis, Job Prediction, Recommender Systems, Machine Learning.**

I. INTRODUCTION

One of the most important things in a job search is knowing where you stand in your career and where do you want to go next. The mindset of a job seeker is very much like a product manager, trying to find the best product and a market fit. Smart product managers know what their customers want. So, they carefully find out customers' needs which are expressed through "top qualifications", "what you must have", in the job descriptions. After identifying what is commonly needed in the market, smart product managers will customize their products aka resumes, cover letters, online profiles, portfolios to demonstrate these characteristics and traits as much as possible.

Accurate recruitment of employees is a key element in the business strategy of every company due to its impact on companies' productivity and competitiveness. At present recruitment processes have evolved into tedious tasks involving rigorous evaluations and interviews of candidates, intending to hire the best-suited professionals for each company's needs. With the advent of the Internet and the web, e-Recruitment has become an essential element of all hiring strategies. Many websites, such as CareerBuilder, Monster or even Social Networks, like LinkedIn, help companies and job seekers to find the best possible matches.

This paper documents our approach to building a knowledge-based recommender system to suggest appropriate jobs to individuals based on their skills. The dataset chosen for this project is "Data Analyst Jobs" from Kaggle which has job listings from companies scraped from the Glassdoor website. We first go about a survey of various papers in the domain to get some knowledge on the various techniques and models used for the same. We then perform an extensive data wrangling and exploratory data analysis. We find relation between different aspects of this dataset and add extract details from the dataset to be used for recommendation and any further analysis. Job recommendation is a very extensive task in today's world, so we try to bring forth a simpler approach based on skills of each candidate in this pandemic situation, which can help them apply for the job.

II. LITERATURE REVIEW

A. *Summary of Applying Data Mining Techniques in Job Recommender System for Considering Candidate Job Preferences [1]*

By: Gupta, Anika and Dr. Garg, Deepak

The recommendation system worked on in this paper is about recommending the best jobs based on salary and candidate's skills. Candidates can have similar looking profiles, but their choices may be different based on what position to choose, and what can they trade-off with – distance, salary, company rating. This recommender system takes Candidate information and applies classification rules along with their job preferences to produce a recommended jobs list.

Assumptions:

The authors have assumed that all the text categorization is in place and discrete with well-labelled values, understandable by the system, and that the candidate or the system automatically updates the information dynamically. They have categorized the company into groups instead of using continuous numerical rating. Decision Trees are used for determining the criteria for attribute splitting. A threshold taken for the sample size determined if the value was selected or not, and that was used to fill the job category with 1 or 0, depending on if the rule exists or not.

It is assumed that all the text categorization is already in place and we have discrete and well-labelled values that are easily understandable by the system. It is assumed that either the candidate or the system itself updates the age, experience, education and skills field. It uses TF-IDF value for feature extraction and information gain with a

threshold value for feature addition. Location preference matrix is not considered as it is assumed that the group has applied equally in all 4 regions and hence adding these will not result in any new information.

Conclusion:

This paper recommended jobs and categorize continuous variables into discrete categories which can reduce the number of values by a large extent and thus improve model performance in speed and accuracy. The use of Decision Trees can be used as a classifier on certain aspects of the problem we are trying to solve.

B. A survey of job recommender systems. [2]
By: Shaha T. Al-Otaibi and Mourad Ykhlef

This paper talks about how recommendation systems influence the field of job recruitment. The main focus is on how to implement recommendation systems for identifying suitable employees for a particular job description. The author presents an overview of the various recommendation systems that are in use, in the current day such as Collaborative filtering approach, Content-based filtering approach and Knowledge-based approach, he also goes on to list various challenges associated with them such as performance decrement due to increase in data, limited scalability, a requirement for prior domain knowledge etc.

Assumptions:

The paper also has attached a case study to recommend candidates for a specific job, which was implemented using a content-based filtering approach as it was assumed to be the best approach. However, this led to the matching of candidates that had specific terms related to the job in their CV but did not take into account their level of expertise in those areas. The hybrid approach of using multiple recommender system techniques didn't seem to improve the quality of job recommendation.

Conclusion:

Content-based filtering method requires a sufficient amount of learning period before understanding the user's preferences. With our topic being the analysis of Data Analyst jobs and the paper's prime discussion is on the various methods used for job recommender systems, this paper has given us an insight into how we can use the similar approach for our project.

C. Salary Prediction in the IT Job Market with Few High-Dimensional Samples: A Spanish Case Study [3]
By: Ignacio Martın, Andrea Mauriello, RobertoBattiti, Jose´Alberto Hernandez'

This paper focuses on predicting the salaries of jobs offered on an online portal called "Tecnoempleo" that specializes in IT jobs for people residing in Spain. E-recruitment of candidates is a hot topic of research and there are many different approaches like an automatic evaluation of CV, a ranking of the skills of candidates which are used by companies to aid in the choice of a candidate.

Salary prediction by ranges is the main focus of this paper, several models like – linear models, logistic regression, K-nearest neighbors, artificial neural network, support vector machines, random forest, adaptive boosting with decision trees along with ensembles of the models stated before are considered to formulate a solution.

Assumptions:

Data is collected from an e-Recruitment website with geographical limits to Spain, the posts for jobs not based in Spain were removed. Here the missing values were removed because in most cases there was no default value with which they could be substituted. The unstructured information like the title of the job post, description of the offered post and requested profile were also removed. These comprise some of the assumptions of the paper considered for this report.

Conclusion:

It is very clearly concluded about how salary classification into a range makes job profiling easier and hence a website easier to use for a person hence we could also attempt a classification approach to our problem.

III. PROBLEM STATEMENT

There are numerous jobs in the market, and there are many candidates who do not know what jobs they can get based on their skills. This is a very tedious work to sift through millions of job openings online and apply to each one of them. This can even lead to a candidate not getting the job best suited for them or a job with a satisfactory salary.

In light of the COVID-19 pandemic a vast majority of people have lost their job and are looking to upskill themselves to find a new job. Various companies are trying to sift through candidates who apply to them and offer them salaries according to the skills they possess. This requires a recommender system based on skills of the candidate to help the candidate apply to the companies based on their skill, to have higher chances of selection. This is the problem which we are trying to solve with this project.

IV. OVERVIEW

V. PREPROCESSING:

This dataset has numerous missing values given as strings and numbers of **-1, "Unknown", and "Unknown / Non-Applicable"**. This was converted to Nan to bring a uniformity in the entire dataset before going further with the cleaning. There were a lot of missing values in the dataset, and they are as follows:

Salary Estimate:	0.044%
Rating:	12.073%
Company Name:	0.044%
Headquarters:	7.634%
Size:	9.099%
Founded:	29.294%
Type of ownership:	7.945%
Industry:	15.668%

Sector:	15.668%
Revenue:	34.532%
Competitors:	76.875%
Easy Apply:	96.449%

The dataset for missing values was filled based on their distribution. The factual missing data were dropped. Categorical columns were filled with mode, and numerical were filled based on the distribution of the column. Some columns were categorical, but were converted to numerical for easier exploratory data analysis and visualization.

Salary Estimate column was converted from a string object to a numerical column. Since there was a very small amount of missing values in this column, they were filled using the mode of the column. The values were given in range of low to high salary, which was split into two new columns of low and high salary. These columns were then used to take the average between the low and high and replace the categorical value in the original column. The two new columns on low and high salary was used for further data exploration.

Only one value from **Company Names** was missing, thereby it was dropped. The company names had the ratings of the company appended to it at the end. This was redundant as there was another column specifically for company ratings, thus it was stripped from the company names.

Easy Apply is the column which specifies if the job is open for requirement or not, thereby all null values were filled with False, assuming that they are not currently hiring but will open the recruitment lines soon. The **Headquarters** is a column having the location of the company headquarters, thereby the missing values was related to the Location. Assuming that the headquarters' location was not filled as the company hiring was the headquarter company, so that column was skipped or ignored, we fill the missing values with the value in the Location column of that corresponding row.

Ratings of a company can depend on various aspects, like the sector, size, revenue, or even the salary. Grouping them based on all these columns, we found that the average rating doesn't change for any group, and the data was evenly distributed. Thereby we fill the missing values in the column with the mean value entire Ratings column.

Revenue was filled with the median based on each size of the company. Then the lower bound of the Revenue was extracted from the range and replaced with the original column, this was done as usually it is the lower bound that people are interested in.

Sector and **Industry** columns are related to each other, as an industry is a sub-domain of the sector. On grouping and checking for missing values, we can see that the number of Sectors missing and the number of Industries missing were the same, and they all belonged to the same row. Thus we can fill the entire column of Sector with the mode of the column. Then the Industry column was filled based on the mode of industry of the sector to which it belonged on the sector.

The **Size of the company** can vary on the location, the rating, or even the revenue. On closer look we could find some relation between the Revenue and Size of the company. This showed that lower the size, lower the rating. Though this might not be true in every case, but this seemed the most significant than just filling the data with the mode of the entire column. Thus the missing values were filled based on the mode of the Size of the company, grouped by Revenue column.

Type of ownership was based on the Sector it belonged to, if it is a government or private or health or education. The missing values were groped on Sector and the mode of each Type of ownership was used for that particular Sector, to fill the missing values in the column. **Competitors** and **Founded** columns were dropped as they had a lot of missing values, and couldn't be filled on approximation as they are factual data.

The **Job Title** had a lot of unnecessary textual information and errors in filling the data, like "**Data Analyst**" and "**DataAnalyst**" were different values. Therefore a regular expression was used to scan through the column and replace split the data into different categories, Job Title and Position. Where the Job Title will be the Job which is being offered and the Position being the job position the company is offering. The data was cleaned to have a uniform title and can help in further exploration of data, and makes it easy to read and understand as a final result.

The data from the **Job Description** was extracted to get the list of all possible required skills. These were *categorized and split into different columns, for data exploration*. All these columns were then clubbed under a single **Skills** column which was to be used for building the recommender system.

VI. EXPLORATORY DATA ANALYSIS:

The exploratory data analysis has revealed numerous information on the dataset and how we can use it to proceed further with design of the model. EDA plays a vital role in understanding the data and in helping decide what problems we can face with a certain approach, and what we can do to improve on the model. Using some of these insights we have build our Recommender System accordingly.

In the fig 4.1 we have used a pie chart to find out how the salary is distributed for every industry present in the dataset. There is a very large number of industries present so we cannot see all of them clearly but we can find out the major industries present. They are IT services, Staffing & Outsourcing, Health Care Services & Hospitals, and Computer Hardware & Software. These industries form a major component of the salary estimate in the order as stated before.

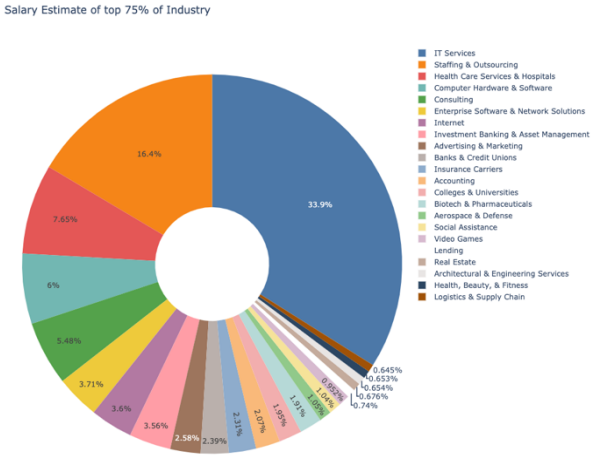

fig 4.1 Salary Estimate of top 75 % of industries. There are a large number of industries so we omitted the lower 75%

We have used a scatter plot matrix in fig 4.2 for a few finding any relation between the numerical columns as a whole. We can see that there is very little to no relation between any of the two attributes. There is ambiguity in the relation between Salary Estimate and the Ratings column. Therefore, these columns cannot be used to build any regression models for recommending the best jobs to the candidates.

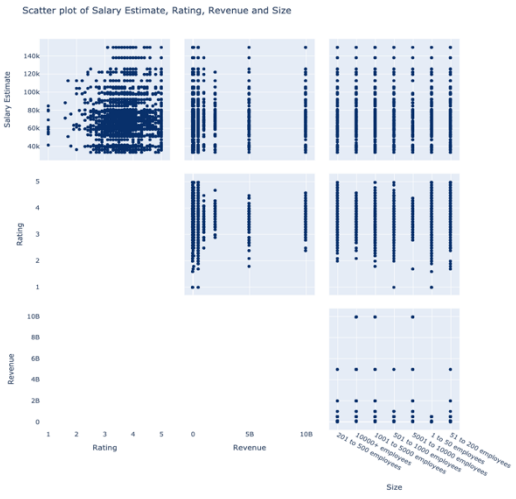

fig 4.2 Scatter Matrix of Salary Estimate, Rating, Revenue and Size, to find any correlation between the data.

In fig 4.3, a distribution plot using histogram and boxplot, we can see the distribution of the salary estimate across the whole dataset. We can find the graph is right skewed which shows us that the higher salaries are less frequently occurring. The boxplot shows a few outliers present in the distribution, which refers shows the extremely high salary provided by the companies.

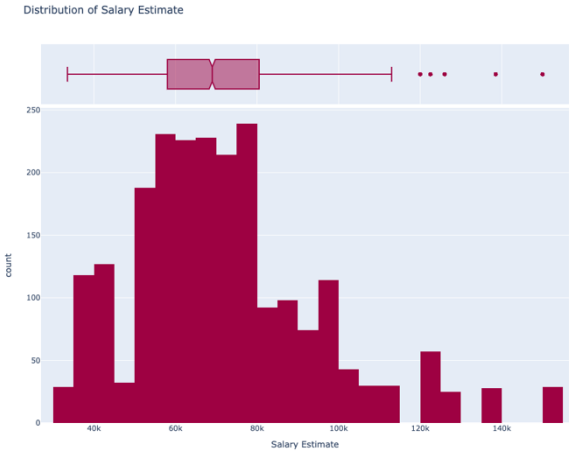

fig 4.3Distribution of Salary Estimate column. It can be seen the data is almost normal with a little right skew.

More on the visualization can be found at our GitHub repository.

VII. BUILDING THE RECOMMENDER SYSTEM:

With the limitations of our dataset, an accurate and good recommender system model cannot be built, so we have decided to build a Knowledge-based recommender system which takes into account a candidate's skill and outputs the best jobs they can apply to, so they have a higher chance of getting selected.

We have made a few assumptions before building the recommender system. They are as follows:

- In the light of the pandemic, the candidate applies for the job at their own location, as travelling is restricted.
- The candidate has no preference in salary, and needs a job to fulfill basic requirements.
- The candidate looking for the job is residing in the United States of America.

After the cleaning of all the data and exploratory data analysis, we created a new feature that will be used by the recommender system. A new feature named **Skills** was created by extracting the list of skills and changing its type to string and stored it in the column for each Job Title. This was the feature which will be used by the recommender system.

A **Jaccard Coefficient** function was created to *calculate the similarity, and based on the user input skills* on what job will be the best for the particular user. Jaccard Coefficient was used as the requirement of the company was based on the provided skills, and not on all possible skills. Thereby the Jaccard Coefficient was taken, rather than Cosine Similarity for our dataset.

$$\boldsymbol{J(A,B)} = \frac{|A \cap B|}{|A \cup B|}$$

$\boldsymbol{J} = Jaccard\ Distance$
$\boldsymbol{A} = Set\ of\ Skills\ required\ by\ company$
$\boldsymbol{B} = Set\ of\ candidate's\ skill$

A list of input was taken from the user about their skills. This is passed through the recommender system, which finds the best similarity from among the jobs in the dataset and it recommends the top 5 jobs based on highest skill requirement provided by the company. Therefore, a candidate seeking for a new job can apply to these companies for a good chance of being selected.

VIII. EVALUATION

Let us consider there are 5 jobs in the listing.

Job Number	Skills
J1	Python, Tableau, R
J2	Python, Excel
J3	C, Python, Java, Excel, French
J4	Database, French, Python
J5	R, Tableau, Python, Scala, Java

table 8.1 List of Jobs in offer

Let there be two candidates with the following skillsets.

Candidate	Skills
C1	Python, Excel, Java, C
C2	Tableau, Python, C, French

table 8.2 List of Candidates and their skills

The input is taken from the user is shown in the table. Each input skill is passed through to the recommender system which calculates the similarity and returns the best jobs that the candidate can apply to. For our evaluation purposes, we will return top 2 jobs for each candidate.

Passing the skills of candidate 1 and 2 through the recommender system, it first calculates the Jaccard Coefficient for each Job listing.

Job Number	Jaccard Coefficient
J1	0.17
J2	0.50
J3	0.80
J4	0.17
J5	0.29

table 8.3 Jaccard Coefficient for Candidate 1

Example calculation of Jaccard Coefficient for Job 1, with Candidate 1:

$$J(J1, C1) = \frac{|\{\text{Python, Tableau, R}\} \cap \{\text{Python, Excel, Java, C}\}|}{|\{\text{Python, Tableau, R}\} \cup \{\text{Python, Excel, Java, C}\}|}$$

$$J(J1, C1) = \frac{|\{\text{Python}\}|}{|\{\text{Python, Tableau, R, Excel, Java, C}\}|}$$

$$J(J1, C1) = \frac{1}{6} = 0.17$$

Similarly, for Candidate 2, we can find the similarity and get the coefficients.

Job Number	Jaccard Coefficient
J1	0.40
J2	0.20
J3	0.50
J4	0.40
J5	0.29

table 8.4 Jaccard Coefficient for Candidate 2

Therefore, sorting and returning the best rated jobs for the candidates will be as follows:

	Job Number	Similarity
C1	J3	0.80
	J2	0.50
C2	J3	0.50
	J1	0.40

table 8.5 Recommended Jobs for each candidate

This way the recommender system works to return the jobs list for a candidate based on their skills. The Jobs with equal similarity are sorted based on their Names/Title.

IX. RESULTS

A knowledge-based recommender system was used due to lack of data that can be taken into account for building a model. A knowledge-based system doesn't depend on historical data, thereby it can give an accurate result of what jobs a candidate can get based on their skills. This helps a user have a clear understanding as to why they lack the capabilities to land a good job, and can thus work to improve their skills before trying for another job.

We have taken a constraint-based system and have designed it based on the constraints provided by the user as their skills. This performs better than a collaborative or content-based recommender systems, as jobs are to be given based on a particular candidate's knowledge and skills, and should not depend on another user's experiences. This type of recommendation system does not need a large amount of data to improve their recommendation capabilities and can work on a small dataset too.

This method can directly be implemented and can still be very apt to any job recommendation or search platforms, without a lot of back-end work. But there is a room for improvement where there can be a lot more inputs from the user, not just on their skills but their domain of working. This can improve the relevance of the jobs to the specific candidate. We do not take those into consideration on the assumption that the candidate wishes to get a temporary job or a graduate student wishes to get experience with any job they can get selected in and has no specific constrains on the domain and location.

X. CONCLUSION

Our project focuses on recommendation of job titles based on skills provided by the user. These skills are matched with the skills of jobs present in the dataset which users can apply to and a similarity rating is obtained. Based on the

rating obtained we recommend the top 5 jobs to the user to choose and apply at their convenience. These job postings are scraped from the website Glassdoor and contain data related to job postings within the geographical limits of USA. There are totally 16 columns and 2253 rows of data present.

The crux of our project lies in the prediction of the job title and this column present in the dataset needed to be cleaned thoroughly to present an accurate recommendation to the user. There were many inconsistencies with the way the job titles had been entered and we used our domain knowledge to modify them and create a cohesive model.

We have converted the range of salary estimate to an average numerical value. Also, the revenue column has been made into numeric form. The other columns have been imputed according to the analysis of missing values present. The job description column is used to extract the required skills for the particular job. We have tokenized the words and extracted the technical terms from it. The spoken or written languages required for the jobs have also been extracted into a separate column.

Since some of the descriptions did not have the technical terms which we had shortlisted we added python and R programming language, and English as the default requirement for all the jobs present in the dataset. The extracted features are converted to strings as they function well with count vectorization. The programming languages and spoken languages are count vectorized and fit on all the records present.

The final recommendation system uses the Jaccard similarity coefficient to determine the best jobs that a user can apply for. The big problem with most recommendation systems is the problem of cold start that is the problem to make recommendations when there is no previous history of the ratings of the user on an item. A simple solution for this is making a knowledge-based recommendation system. A knowledge-based recommendation system is one which makes recommendations not based on the user history but based on the specific queries that are made by the user. It might ask the user to give a series of rules or guidelines as to how the final results should be presented.

Knowledge-based recommender systems are a much-needed entity in a world of ever-expanding information resources. Unlike other recommender systems, they do not depend on large bodies of statistical data about particular rated items or particular users.

Our research has shown that the knowledge component of these recommender systems need not be prohibitively large, since we need only enough knowledge to judge items as similar to each other. Further, knowledge-based recommender systems actually help users explore and thereby understand an information space. Users are an integral part of the knowledge discovery process, elaborating their information needs in the course of interacting with the system. One need only have general knowledge about the set of items and only an informal knowledge of one's needs; the system knows about the tradeoffs, category boundaries, and useful search strategies in the domain.

There are two main types of knowledge-based recommendation systems:

1. **Constraint-based recommender systems:** The users specify their requirements or constraints on the item attributes. Furthermore, domain-specific rules are used to match the user requirements to the item attributes.
2. **Case-based recommender systems:** Specific cases are specified by the user as targets. With similarity metrics being defined in a domain specific way.

In our approach we have used a *constraint-based recommendation system*. There is a conversational approach to obtain input from the user. The user enters the skills which they possess and these act as constraints for the recommendations which we can provide. Similarity is calculated between the skills of the jobs present in the dataset and the skills provided and the top 5 jobs which have the highest similarity are given as a recommendation to the user.

XI. REFERENCES

[1] A. Gupta and D. Garg, "Summary of Applying Data Mining Techniques in Job Recommender System for Considering Candidate Job Preferences," 2014. [Online]. Available: https://gdeepak.com/pubs/Applying_Data_Mining_Anika.pdf. [Accessed 22 September 2020].

[2] S. T. Al-Otaibi and M. Ykhlef, "A Survey of Job Recommender Systems," 26 July 2012. [Online]. Available: https://academicjournals.org/journal/IJPS/article-full-text-pdf/B19DCA416592.pdf. [Accessed 22 September 2020].

[3] I. Martín, A. Mariello, R. Battiti and J. A. Hernández, "Atlantis Press: International Journal of Computational Intelligence Systems," 2018. [Online]. Available: https://www.atlantis-press.com/journals/ijcis/25899235/view. [Accessed 22 September 2020].

XII. CONTRIBUTIONS

Each member of the team has had equal contributions in specific parts of the project. We divided the work equally so no one person had all the work to perform. The contributions are as follows:

- **Data Preprocessing:** Afzal Mukhtar, Hritika Rahul Mehta, and Aswin A Nair.
- **Exploratory Data Analysis:** Hritika Rahul Mehta and Afzal Mukhtar.
- **Tokenizing and creating features:** Aswin A Nair.
- **Building recommender system:** Hritika Rahul Mehta and Afzal Mukhtar.