Algorithm and Optimization for Big Data Final Exam

Name: Yash Kotadia, Roll No: 1401114
School of Engineering and Applied Science
Ahmedabad University

***Abstract*—On the Internet, where the number of choices is overwhelming, there is need to filter, prioritize and efficiently deliver relevant information in order to alleviate the problem of selecting proper career path. Recommendation systems solve this problem by searching through large volume of dynamically generated information to provide users with personalized content and services. They are commonly used over the Internet to guide customers to find the products or services that best fit with their personal preferences. Many people have chosen their career path without receiving proper advice from suitable professional. This may potentially cause mismatch between academic achievements, personality, interest and abilities of the user. In order to suggest career progression paths to user, it is essential to build a recommendation system that provides direction and guidance how the user should consider next set of skills to be acquired. In this paper we aim to solve this problem in two modules.The first module will suggest a user some skills he/she needs to gain using collaborative filtering approach and In the second module a career path is suggested based on a career goal entered by user. Here mean and fuzzy logic technique is used along with the collaborative filtering**

Keywords: Recommendation System, Linear Regression, Collaborative Filtering, Regularization, Gradient Decent, Mean Normalization, Fuzzy Logic.

I. Introduction

Ever wondered, "what algorithm google uses to maximize its target ads revenue?". What about the e-commerce websites which advocates you through options such as 'people who bought this also bought this'. Or "How does Facebook automatically suggest us to tag friends in pictures"?

The answer is Recommendation Engines. With the growing amount of information on world wide web and with significant rise number of users, it becomes increasingly important for companies to search, map and provide them with the relevant chunk of information according to their preferences and tastes.

Companies nowadays are building smart and intelligent recommendation engines by studying the data their users. Hence providing them recommendations and choices of their interest in terms of "Relevant Job postings", "Movies of Interest", "Suggested Videos", "Facebook friends that you may know" and "People who bought this also bought this" etc.

Recommendation systems are simple algorithms which aim to provide the most relevant and accurate items to the user by filtering useful stuff from of a huge pool of information base. Recommendation engines discovers data patterns in the data set by learning consumers choices and produces the outcomes that co-relates to their needs and interests.

Aim of this paper is to design recommendation system for a company like LinkedIn wants to build a module for suggesting career progression paths to its registered users. When a user logs onto the platform, the platform reads user's profile and based on various parameters of this profile comes up with relevant suggestions on how the user should consider next set of skills to be acquired. Basucally it is divided into two modules , a module that reads user's profile and suggest a career path – in terms of skillset to be acquired and another module in which user enters a career goal and based on this career goal and other related information the platform suggest a career path.

II. Recommendation paradigms

Most recommender systems take either of two basic approaches: collaborative filtering or content-based filtering. Other approaches (such as hybrid approaches) also exist.

1. Collaborative Filtering

Collaborative filtering arrives at a recommendation that's based on a model of prior user behavior. The model can be constructed solely from a single user's behavior or — more effectively — also from the behavior of other users who have similar traits. When it takes other users' behavior into account, collaborative filtering uses group knowledge to form a recommendation based on like users. In essence, recommendations are based on an automatic collaboration of multiple users and filtered on those who exhibit similar preferences or behaviors.

2. Content-based Filtering

Content-based filtering constructs a recommendation on the basis of a user's behavior. For example, this approach might use historical browsing information, such as which blogs the user reads and the characteristics of those blogs. If a user commonly reads articles about Linux or is likely to leave comments on blogs about software engineering, content-based filtering can use this history to identify and recommend similar content (articles on Linux or other blogs about software engineering). This content can be manually defined or automatically extracted based on other similarity methods.

3. Hybrids

Hybrid approaches that combine collaborative and content-based filtering are also increasing the efficiency (and complexity) of recommender systems. The results of collaborative and content-based filtering creates the potential for a more accurate recommendation. The hybrid approach could also be used to address collaborative filtering that starts with sparse data by enabling the results to be weighted initially toward content-based filtering, then shifting the weight toward collaborative filtering as the available user data set matures.

III. Dataset

Dataset available to us a form of user profile data. Data given to us contains CandidateID, Additional info, Skills, Work experience, Location and etc. Data was very rough, it was not structured properly. So,First of all we need to parse JSON files and clear the unwanted data. Then extracted CandidateID, identified different skills and year.After having some operations on dataset, it is cleared and ready to use for our module design

IV. Module 1

collaborative Filtering approach is used to find the skill of any particular candidate. Mainly focus was on two things ,skill sets the user has and what is the profession of user. Now we need to collect the skills of users and it will be stored in the matrix. I have used 0 or 1 notation for users' skills. If user has that skill that it will be denoted by 1 else 0. This matrix is kind of sparse. All users will not have all the skills. So, Data matrix Y exists in the form of a sparse matrix, where rows correspond to skills, columns correspond to user and the matrix entries are either zeros or ones. In that we map X matrix as average experience required of that particular job We have feature vector which is based on the experience the user has in the field and the experience the user has in that particular skill. Now, we evaluate the cost function obtained by the feature vector x and the parameter θ (i.e. experience of each user) and computing θ^Tx. To predict the skills we will apply collaborative filtering algorithm. This is nothing but linear regression that we apply having cost function. This helps in predicting the future values. A part of Machine learning comes into play when we use updating of weights in the cost function by gradient descent to reduce errors. Essentially, we will have learned the appropriate values of user and experience to make accurate predictions on the skills for every user.

Algorithm

1: **initialize**: (i, j): User j has achieved ith skill.
2: Y(i,j) : experience by user j in skill i.
3: if particular person has acquired that skill
4: The value of R(i,j) is 1.
5: else
6: The value of R(i,j) is 1.
7: And Y(i,j) is defined iff R(I,j) is 1.
8: **output:** Skills

V. Module 2

Second module is all about giving suggestions based on users career goal.

A. Approach 1

This approach i similar to above approach. Firstly, we have sparse matrix whose column contains the number of users and row has number of skills. As new user comes he will be assigned the required parameter like X, Theta and Y. Then, we calculate linear regression followed by gradient decent to minimize our cost function. **Algorithm**

1: For every skill i that user u has no preference or some preference
2: For every other user y that has a preference for i
3: Compute a similarity s between u and y
4: Add v's preference for i, weighted by s, to a running average
5: Return the top skills, ranked by weighted average

B. Approach 2

Another approach used in order to build this system is Mean, fuzzy logic and string matching. This system recommends those skills first those been adopted by maximum no of users. In order to help the user while entering the career goal, i have also used auto-complete recommendation. So when user write "software" in search bar, system suggests career goals having "Software" keywords. Eg – Software Developer, Software Engineer, Software Architect

Algorithm

1: Initially parse the JSON
2: Using "forEach" loop map the array of skills corresponding to career goal.
3: Internally sort the skills on the basis of number of times they occur in give role and put them accordingly in order.
4: At the time of search, we have autocomplete of the career goal, that will have the user alot at the time of typing.

Here career goal contains all the element in the form of javaScript object. "forEach" function iterate through every item of object which contains Role (Job Position) and Skills. Role (Job Position) is taken here as a career goal. The role is pushed into array for searching purpose and checking whether the same role is already present in the map. if yes, then append the skill set with given skill and concat in previous skills of the same role. if this is for the first time, simply add the skill corresponding the new role.

Next we remove the duplicate value out of the array of job roles. Thus we have unique career goals. We are doing search route in order to set all the roles for auto-complete. The required skills set according to the job type will be returned by fetching the job string from the route, Here the reduce function is going to map the skills set according to the number

of times the given skill occurs. for example, if someone wants to become Software Developer and in that role JAVA comes 5 times and SQL comes 3 times then first it will suggest JAVA.

VI. Challenges

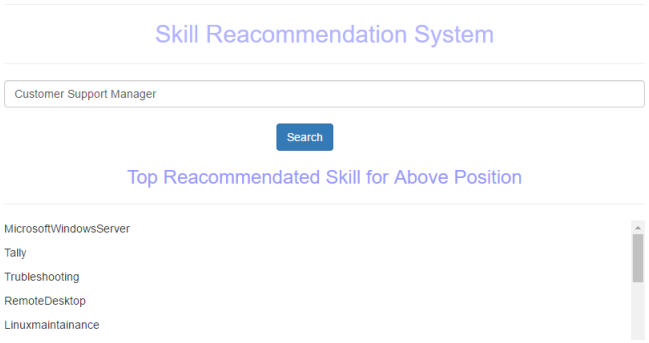
Taking advantage of the ”wisdom of crowds” (with collaborative filtering) has been made simpler with the data-collection opportunities the web affords. But the massive amounts of available data also complicate this opportunity. For example, although some users' behavior can be modeled, other users do not exhibit typical behavior. These users can skew the results of a recommendation system and decrease its efficiency.

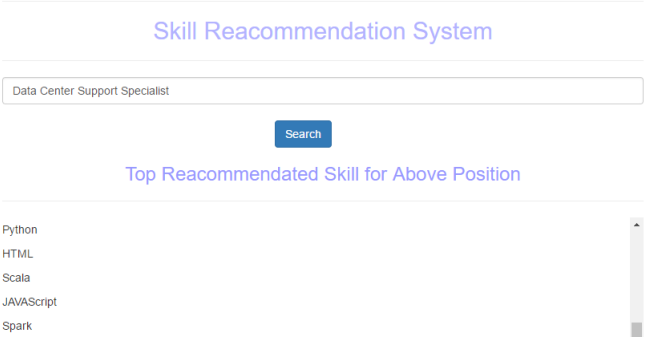
VII. Result

- Approach 1

```
Top recommendations for you:
Predicting rating 8.5 for Skills Python
Predicting rating 5.9 for Skills ESD
Predicting rating 4.9 for Skills Sql
Predicting rating 4.7 for Skills Oracle
Predicting rating 3.5 for Skills C++
Predicting rating 3.5 for Skills Java
Predicting rating 3.2 for Skills C
Predicting rating 1.7 for Skills Db2
Predicting rating 1.2 for Skills Hadoop
Predicting rating -2.2 for Skills Shell
```

Approach 2





References

[1] https://www.ibm.com/developerworks/library/os-recommender2 https://www.ibm.com/developerworks/library/os-recommender1

[2] F.O. Isinkayea, Y.O. Folajimib, B.A. OjokohcRecommendation systems: Principles, methods and evaluation, Published in science direct (2015)

[3] https://www.upwork.com/hiring/data/how-collaborative-filtering-works

[4] www.researchgate.net/publication/227031714 Facial Expression Recog.

[5] A Research of Job Recommendation System Based on Collaborative Filtering, Published in: Computational Intelligence and Design (ISCID), 2014 Seventh International Symposium on (2014)