CS685: Group 22

Prakhar Banga, 60, Y9418, prakban@iitk.ac.in, Dept. of CSE

Romil Gadia, 68, Y9496, romilg@iitk.ac.in, Dept. of CSE

Shashank Sonkar, 73, Y9545, ssonkar@iitk.ac.in, Dept. of CSE

Indian Institute of Technology, Kanpur

Final report
16th Nov, 2012

Abstract

Predicting the jobs people will apply to, based on their previous applications, demographic information, work history and education info.

1 Introduction

We are living in an era where everything is desired on a single click. Web and Internet have started playing a significant role in our day to day activities. As a result, Internet has no dearth of content. But, the problem lies in finding the ”relevant” information among this humongous data that is very often specific to a user. If we are able to properly filter this data according to users' needs, we can greatly enhance user experience on the internet. This led to the development of Recommendation Systems to help users find the movies that match their taste, news that interests them, friends they may want to get connected with and also ads that may want to see. One such field is recommending jobs to a user. The problem here lies in the fact that there is huge variety of jobs available, say from a technical manager to a salesman, or say from a software engineer to a truck driver. Furthermore, even if we are able to group jobs of relevant fields, this doesn't solve the problem completely. This is because all jobs of same field might not be of interest. For instance, a person holding a technical manager's job won't apply for software engineer job profile. Similarly, a person in a very good company A won't opt for recently established company B unless this difference is compensated by a promotion in position. Similarly, demography may be another factor. The user would prefer job openings from nearby places. Thus, incorporating various factors to finally come up with good suggestions is aim of any recommendation system and so of the Job Recommendation System.

1.1 Problem Statement

The aim of this project is to predict the jobs, users would be most interested in, given their past work experience, their graduation level, their locations, details of current job openings and their recent job applications. The problem statement has been inspired from the Job Recommendation Challenge by Kaggle1.

2 Preliminaries

2.1 MAP150

For each applicant, 150 ranked job predictions are to be made. To account for the importance of the higher ranked, measure known as MAP 150 is used. `en.wikipedia.org/wiki/Information_retrieval#Mean_average_precision`

2.2 Collaborative Filtering

To predict which jobs a user X will apply to, we first find the users which have previously applied to the same jobs as X, and then consider the other jobs they have applied to. The idea here is that similar users apply to multiple common jobs.

3 Methodology

3.1 Dataset

The dataset contains information about job applicants: demographic information, degree type, major, graduation date, experience, employment status, application history and about jobs: title, description, application duration, requirements and demographic information.

3.2 Benchmark

150 most popular jobs are proposed to each applicant based on his/her location(city, state). First, the popular jobs from his/her city are used for filling the recommendation followed by that from the state. The MAP150 value obtained for this prediction is used as a benchmark.

3.3 Training and Test Data

The training data was given and test data had been hidden by the challenge. Unfortunately, due to challenge getting over, the evaluation process was closed. So we used the approach of validation sets. We divided the existing training data into training and test datasets. We used the following three divisions:

1. 70:30 division - first 70% into training set and remaining into test set.
2. 50:50 division - first 50% into training set and remaining into test set.
3. 70:30 division - random 70% into training set and remaining 30% into test set

3.4 Approach/Algorithm

We give a score to each job for each user, which reflects the likelihood that the user will apply for that job. So score(A, X) is the likelihood that user A will apply for job X. Based on the scores, we sort jobs and select the top 150 as our predicted jobs. In case the predicted jobs are less than 150, we fill the rest of the slots with jobs predicted based on the benchmark algorithm(geographic proximity)

3.4.1 Collaborative Filtering

If A applies for the set of jobs X = {X1, X2, X3...} and B applies for Y = {Y1, Y2, Y3...} then score(A, Yi) is incremented by size(X $\bigwedge$ Y)

3.4.2 Collaborative Filtering(Jaccard Index)

If A applies for the set of jobs X = {X1, X2, X3...} and B applies for Y = {Y1, Y2, Y3...} then score(A, Yi) is incremented by size(X $\bigwedge$ Y) / size(X $\bigvee$ Y), i.e. the Jaccard Index for A and B.

3.4.3 Collaborative Filtering(till level 2, normalised using Jaccard index)

1. Apply Collaborative Filtering once to get initial scores, say score1
2. If A applies for the set of jobs X and B applies for Y, then score2(A, J) is incremented by score1(B, J) * size(X Y) / size(X Y)

3.4.4 Collaborative Filtering(Using other attributes such as degree types, majors, years of experience)

If A applies for the set of jobs X = {X1, X2, X3...}and B applies for Y = {Y1, Y2, Y3...}then score(A, Yi) is incremented by sim(A, B) * size(X $\bigwedge$ Y) / size(X $\bigvee$ Y), where sim(A, B) is a function of the degree types, majors and years of experience of A and B.

4 Results

The results for different divisions of data are given in Table 1, Table 2 and Table 3.

Approach	MAP150 score
Benchmark (geographic proximity only)	0.0124183360415
Simple Collaborative filtering	0.0230535435623
Collaborative filtering (using Jaccard Index)	0.0271350654504
Collaborative filtering (using Jaccard Index upto level 2)	0.0271101963712
Collaborative filtering (using Jaccard Index, user degree, major, experience upto level 2)	0.0281555041408

Table 1: Results for 70:30 split(first 70% into training, rest into test)

5 Result Analysis

1. Why did collaborative filtering work so well? The idea of collaborative filtering is based on similarity

Approach	MAP150 score
Benchmark (geographic proximity only)	0.0189286804232
Simple Collaborative filtering	0.0340171201336
Collaborative filtering (using Jaccard Index)	0.039737095644
Collaborative filtering (using Jaccard Index, user degree, major, experience upto level 2)	0.0410528786914

Table 2: Results for 50:50 split(first 50% into training, rest into test)

Approach	MAP150 score
Benchmark (geographic proximity only)	0.01358401625564
Simple Collaborative filtering	0.03152891820352
Collaborative filtering (using Jaccard Index)	0.03741029054332
Collaborative filtering (using Jaccard Index, user degree, major, experience upto level 2)	0.0385417758471

Table 3: Averaged results for repeated random subsampling(70% in training, 30% in test) 5 times

of users, i.e., two users are similar if they apply to the same jobs. The similarity of two users is measured by the number of jobs they both applied to. This gives us pretty good results because practically, users with similar job interests are highly likely to apply to the same multiple jobs. This is an extremely effective heuristic which is extensively used in recommendation systems.

2. Why did collaborative filtering with Jaccard Index work better? Normalization with Jaccard index takes into account the total number of jobs the users applied to as well. So the similarity of two users who applied to 5 common jobs and 5 other jobs is higher than that of two users who applied to 5 common and 10 other jobs. As it considers the proportion of common jobs from the total jobs as compared to just the common jobs, we expected it to increase the accuracy, which it does(by about 20

3. Why did the level 2 of collaborative filtering not contribute much? Similarity relations between users are not necessarily transitive, i.e. if A is similar to B and B to C, then A is not necessarily similar to C. In the case that A is actually similar to C, this approach will help and if not, it can worsen the results. Experimentally, we found that the chances of dissimilarity are higher and therefore, going upto depth 2 in collaborative filtering slightly worsened the results.

4. Why did considering degree type, majors, years of experience of both users didnt improve results by much? These values helped our results improve by about 4%. The reason they didnt work better was that the similarity was already very well captured by the Jaccard similarity coefficient and this information had very little to add to it. There are so many diverse fields in majors that extracting the important keywords to find correlation between two users is almost impossible without applying domain knowledge on each and every major title. For instance, Behavioral Sciences and Psychology are related fields, but capturing this correlation is not so trivial.

5. Why didn't we use job description details? We also tried content-based filtering, i.e. looking at the users past job, major etc. and comparing it with the job titles, descriptions and requirements. It performed even worse than when jobs were predicted on the basis of geographic proximity alone. This is because text similarity is a very hard problem. Even having the domain knowledge(i.e. knowledge of english), it is very hard to tell whether a job matches a user profile because of natural language properties such as ambiguity and redundancy.

References

(1) http://www.kaggle.com/c/job-recommendation