SAN JOSÉ STATE
UNIVERSITY

Job Recommendation System

CMPE 256
Summer 2019

Project Report
By

Apoorva Shadaksharappa (013726362)

Instructor
Prof. Shih Yu Chang

Motivation:

Recommendation systems usually involve exploiting the relations among known features and content that describe items (content-based filtering) or the overlap of similar users who interacted with or rated the target item (collaborative filtering). To combine these two filtering approaches, current model-based hybrid recommendation systems typically require extensive feature engineering to construct a user profile. Nowadays with rapid advancement in the technology, most of the companies have Internet-based recruiting platforms as a cardinal recruitment channel, where more and more job seekers release their personal information, whereas enterprises post their job requirements on the internet. With so many different IT-related job types in demand nowadays, it is hard for a fresh graduate or multi-skilled professional to determine where they would fit best. This problem is most apparent in today's data-driven economy, where the job descriptions and roles are closely connected and often overlapping. By using the recommendation technology, for example, content-based recommender and collaborative filtering recommender a job recommender system can be built which can retrieve a list of job positions that satisfy a job seeker's desire, or a list of talent candidates that meet the requirement of a recruiter. The job recommender system, which is the online recruiting system with personalized recommendation, has been proposed to handle the issue for job seekers and enterprises.

Target Customers:

Keeping various needs of users in mind we can build a recommender system that be can be useful for both the job seekers and the recruiters, since the job aspirants search for the job positions whereas the recruiters search for the candidates. Basic audience targeting includes demographic options such as age, gender, etc. However, the platform has much more detailed information on its users that you can use for precise, professional micro-targeting; such as company industry & size, job title, job seniority, job function, years of experience and relevant skill sets. You can go even further and more surgically target individuals based on specific companies & groups, company connections, followers, and that's just for starters.

Business Value:

One can vouch for high quality candidates. Discover candidates best suited for specialized positions. Also, from job seekers perspective, they can find a perfect job that suits their profile. And in case of other ecommerce businesses one can even convert shoppers into customers. Companies can quickly search and filter influential people and other companies and save them in lead lists to formulate organized groups of potential leads to connect with. For almost every service, company or business today, word-of-mouth recommendations are everything. For example, you wouldn't see a movie in theaters if it had a 2 out of 5-star rating on Rotten Tomatoes with horrible reviews.

Dataset:

The first step in any web scraping project is to ensure that the terms of service allow for you to mine their data. Upon reading the terms of service, I found that web scraping Indeed is allowed, so I proceeded with my project. For this project, I have web scraped the data from Indeed, a job aggregator that updates multiple times daily. I conducted my scraping using the "requests" and "Beautiful Soup" libraries in python to gather and parse information from Indeed's pages, once we obtain the information we can use "Pandas" library to assemble the data into data frame for further cleaning and analysis purpose. The first step towards solving the problem was to find the appropriate dataset. This quest was not as simple as one may think, because, even though there is a great abundance of sites that collect and present job-related information, they do not allow mass extraction of their datasets easily. However, after some research, I ended up using data from the online job-listing company indeed.com that provides unlimited and charge-free job-search services all over the world. The service requires simply that a user specify a keyword, and a geographic area and optionally other criteria the engine quickly responds with several job listings referring to openings that match the user's request. Indeed.com follows only a small number of distinct HTML formats to display its listings, and so we managed to develop a web-scraper (aka web-crawler) that handled the job retrieval in an automated fashion while storing the results in a CSV flat file that we later processed for the modelling purposes.

Dependencies Installed for Scraping: bs4 (Beautiful Soup), tqdm, requests, html2text.

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.

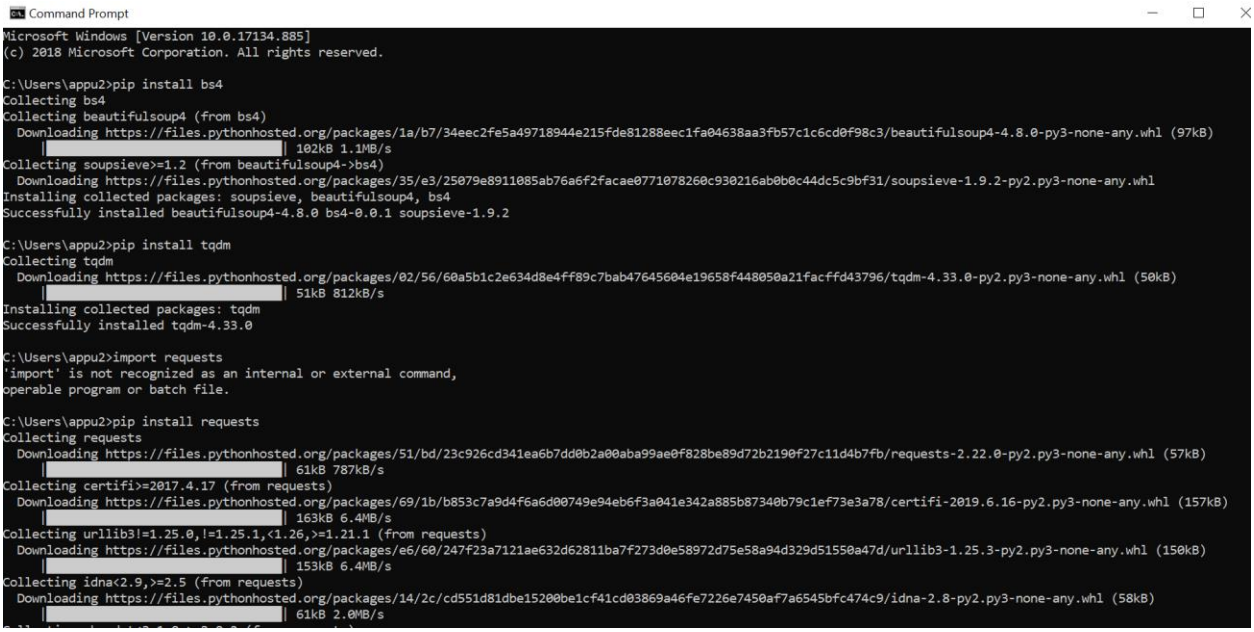
```
from bs4 import BeautifulSoup
soup = BeautifulSoup(html_doc, 'html.parser')
```

Tqdm Instantly make your loops show a smart progress meter - just wrap any iterable with tqdm(iterable)

```
from tqdm import tqdm
for i in tqdm(range (10000)):
```

Request library is the de facto standard for making HTTP requests in Python. It abstracts the complexities of making requests behind a beautiful, simple API so that you can focus on interacting with services and consuming data in your application.

html2text is a Python script that converts a page of HTML into clean, easy-to-read plain ASCII text. Better yet, that ASCII also happens to be valid Markdown (a text-to-HTML format).



Data Attributes:

- ID: a unique number enumerating the jobs extracted
- Job Titles: the titles of the jobs indeed.com returned. At this point we should note that the latter returns to the user results that either match exactly the given job title or they are close to it. For example, if we look for “Data Analyst” jobs we will also get “Business Analyst” jobs.
- Keywords: the terms we have used to find the jobs; in other words, the job titles.
- Job Description: this is the main body of a job offer as it is displayed in indeed.com.

Approach:

A recommendation engine filters the data using different algorithms and recommends the most relevant items to users. It first captures the past behavior of a customer and based on that, recommends jobs which the jobseeker might be interested in based on the keywords (soft and technical).

Pearson’s Correlation is a way to find out similar users. The correlation is a way to represent data sets on graph. Pearson’s correlation is x-y axis graph where we have a straight line known as the best fit as it comes as close to all the items on the chart as possible. If two users rated the books identically then this would result as a straight line (diagonal) and would pass through every book rated by the users. The resultant score is this case is 1. The more the users disagree

from each other the lower their similarity score would be from 1. Pearson's Correlation helps correct grade inflation. Suppose a user 'A' tends to give high scores than user 'B' but both tend to like the book they rated. The correlation could still give perfect score if the differences between their scores are consistent.

Collaborative filtering

Collaborative filtering is the most commonly used technology for human recommendation. This Algorithm is based on users' past preferences and find similarity between the users. Collaborative filtering works on by building a database (user-item matrix) preferences for items by the user. Then it matches users with relevant interests and preferences by calculating similarities between their profiles to make recommendations.

Recommendation systems with collaborative filtering models which assume that people like things that are like the things they like, and things that are liked by people with similar taste. Collaborative Filtering is categorized into two types, i.e. memory based Collaborative Filtering and model based Collaborative Filtering.

A. Model based Collaborative Filtering: Model based on collaborative filtering consists of models that are developed using machine learning algorithms to predict the user's rating of unrated items. This algorithm is based on an offline pre-processing or "model-learning" phase. At run-time, only the learned model is used to make predictions. The models are updated and re-trained periodically. Building the model and updating it regularly can be computationally expensive. This algorithm uses a large variety of techniques. Machine learning algorithms used are Clustering based, Matrix factorization and Deep learning. This type of algorithm is used for huge sparse data. This type of filtering algorithm has advantages such as Scalability, fast prediction speed, and low overfitting. These types of algorithms are inflexible and less quality of prediction.

B. Memory based Collaborative Filtering: Memory based algorithms, also known as the neighbor-based algorithm. Memory based algorithms are widely used in many of the social media platforms and e-commerce sites, such as Amazon, Facebook, Twitter, etc. The Memory-based algorithm approaches is to find a correlation between the users and items in the database to predict the similarity score between them. This predicted score is used whether the user likes the item in the future or not. Using this score, we can recommend the item to a user or not. This algorithm uses an entire database to predict a similar product that the target user might have liked or rated or used. This method consists of two types of algorithms, i.e. item based, and user-based filtering.

C. Item Based Collaborative filtering: Item based collaborative filtering methods calculate the similarities within a group of items that are rated or searched by the user. item based methods are used for recommending similar products based on the products already used or searched by

the user. This model specifies the target user and finds the items rated by the user. It also predicts similar items to the user rated items and recommends them.

D. User based Collaborative filtering: User based methods predict the recommendations based on the similarity between the user. In user-based, users who gave ratings for items in the past will tend to give a similar rating in the future. The algorithm considers the item rating of the user and then predicts the user's rating for the item for which the user has never interacted. Users preferences remain stable and throughout execution. This model specifies the target user and finds the items rated by the user. Then extracts the items that the user has never used and recommends them to the user.

Conclusion:

In this paper, we have introduced a new method for automatically creating datasets for the offline evaluation of job posting similarities. Building dense representations based on full-text job descriptions yields the best results. However, computing representations for novel job postings becomes computational expensive, as the model has to be recomputed, as estimating representations for new documents results in much lower results. Building models from titles, the scores only slightly decrease, however, the computation of new models is much faster. In our experiments, we observe the best performance with a combined model, using the words within the title for weighting words in the description that allows to compute new representations in an online scenario.

References:

1. Omar Abdelwahab and Adel Elmaghraby. 2016. UofL SemEval-2016 task 4: Multi domain word2vec for twitter sentiment classification. In Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval at NAACL-HLT, pages 164–170, San Diego, CA, USA.
2. Fabian Abel, Andras A. Bencz ´ ur, Daniel Kohlsdorf, ´ Martha Larson, and Robert P ´ alovics. 2016. ´ RecSys challenge 2016: Job recommendations. In Proceedings of the 10th ACM Conference on Recommender Systems, pages 425–426, Boston, MA, USA
3. https://pypi.org/project/bs4/
4. https://pypi.org/project/html2text/