Project Report: User-Job Suitability Measurement

XIN LIN
Department of Computer Science
University of Texas at Austin
Austin, TX 78705
jimmylin@utexas.edu

Abstract

Abstract here.

1 Introduction

2 Problem Formulation

if we assume that all job seekers are extremely knowledgeable (understand clearly and completely the profile and requirement of every job) and rational (never apply for the unsuitable jobs), we can directly makes use of the score obtained in the application prediction. However, such assumption receives little support from practical analysis, in the sense that people tend to apply for the job positions with higher salaries and correspongly much more capability seeking.

2.1 Suitability Measurement As Matrix Completion

For startup, we first focus on binary suitability measurement problem. That is, we only investigate the binary association, suitable (1) or not suitable (0), for each pair of user and job. At first glance, this problem can be naively solved by traditional binary classifiers, e.g. logistic regression and support vector machine, or one-class learning solvers (or data description techniques). Nevertheless, more carefull examination reveals obvious disadvantage of such treatment. The main drawback of using traditional binary classifier is that.

2.1.1 Content-based Filtering

2.1.2 Collaborative Filtering

The developing motivation of *Collaborative Filtering* is that the profiles of items or users are not always available or poorly collect in some settings. Instead, the past histories of user-item association, also called *explicit feedback*, are easy to obtain.

Nearest neighbour method and latent factor model are two major approaches for Collaborative Filtering.

This model has an underlying assumption: the real exact rating matrix are low-rank matrix. Stochastic Gradient Descent and Alternating Least Square are two main approaches to numerically achieve matrix completion task of this category.

In netflix prize, the latent factor model, combined with temporal dynamics and bias evaluation, became a single model with the greatest prediction power.

However, collaborative filtering fails to provide supervision of missing values when only rare amount of previous association histories are provided, especailly when the systems are on its early stage. This is commonly referred to as *cold startup* problem.

2.1.3 Features-incorporated Matrix Completion

A recently emerging paper proposed an *Inductive Matrix Completion* method, in which features of items are considered while completing matrices. According to [1], this feature incorporation method can be explained as a way to provide additional support for sparse matrices. From this perspective, we can see it as a new approach that incorporates advantages from both collaborative filtering and content-based filtering. A more recent paper [2] in bioinformatics demonstrated successful application of inductive matrix completion on gene-disease analytics.

One possible enhancement for above inductive matrix completion is to extend our consideration from the linear association between features and latent factors to an version that accepts non-linear association. By this intuition, we can name this approach as *Kernel-based Inductive Matrix Completion*. By taking into account non-linear relations between designed features and hidden topics (shared latent factors), the space of latent factors can be largely expanded and then it would be more likely to automatically detect latent factors with higher quality.

2.2 Suitability Measurement with Prerequisites

1. simulate course recommendaiton (by adita)

3 Experiments: Application Prediction

Due to limitation of acquired data, our first experiment is oriented to problem of application prediction. Specifically, given a set of featured users and featured jobs, the designed system should predict whether one user will apply for one particular job.

3.1 Dataset

The dataset utilized in this experimental project comes from a Job Recommendation Challenge posted on Kaggle.com. The provider of this dataset is CareerBuilder.com, one of the biggest job recommendation service providers. This particular dataset, sized of several Gigabytes, contains a collection of featured users, a collection of characterized jobs and a series of application records that are divided into training and testing split. In this section, we will at first provide the fundamental introduction to these three basic elements: User, Job, Application. And then explanation are illustrated about temporal separation (concept of window) and designed distribution for user, job and application.

As the most essential component of job recommendation system, users are recorded by UserID, WindowID, Split, City, State, Country, ZipCode, DegreeType, Major, GraduationDate, WorkHistoryCount, TotalYearsExperience, CurrentlyEmployed, ManagedOthers, ManagedHowMany. *UserID* refers to the nubmering index of that indicated user. *WindowsID* is about the timing period in which this particular applicaiton about user happened. *City, State, Country, and ZipCode* are related to the living place of that user. What follows are the achievements in school. *DegreeType* shows the highest degree that user has gained from school, *Major* presents the field of his/her speciality and *GraduationDate* reveals when he/she gained the highest degree. Working and management experience comes next. *WorkHistoryCount* represents how many previous jobs one has had and *TotalYearsExperience* represents how many years one has been involved in occupation. *CurrentlyEmployed* and *ManagedOthers* are both binary values, indicating whether one was currently on his/her job and whether he has been in certain management position. *ManagedHowMany* implies his management power and capability, that is, the maximum number of people he/she has managed before.

When it comes to information of every individual job, fields like JobID, WindowID, Title, Description, Requirements, City, State, Country, Zip5, StartDate, EndDate are provided. *JobID* is the identifying number for each particular job. *WindowID* captures the same semantics as it is in a user record – involved timing period of one job. *Title* is the name of position sepcified by corresponding corporation, which can be significantly important since it reflects relative position and power in one company's hierarchy. Description and Requirements are both textual characterization over each particular job. *Description* provides a characteristic overview of one particular job. *Requirements* can be viewed the basic expectation of hiring company towards the job applicants. Similarly to the

record of each individual user, every job also has location information, such as *City, State, Country, ZipCode*. Besides, *StartDate* and *EndDate* shows the timing information of one job, i.e. on which day it starts and ends.

As to each piece of application history record, the dataset contains information about the UserID, WindowID, Split, ApplicationDate, and finally JobID. *UserID* is indexing number of the user who applied for particular job, indexed by *JobID*. *WindowID* implies the timing period of that application event, while *ApplicationDate* presents the specific date of application event. And *Split* labelled in which division (training or testing set) that piece of application was placed.

In outline, the data on users, job postings, and job applications that users have made to job postings is provided. In total, the applications span 13 weeks. All the job applications are split into 7 groups, each group representing a 13-day window. Each 13-day window is split into two parts: The first 9 days are the training period, and the last 4 days are the test period. The graphical representation demonstrating such splits is illustrated below.

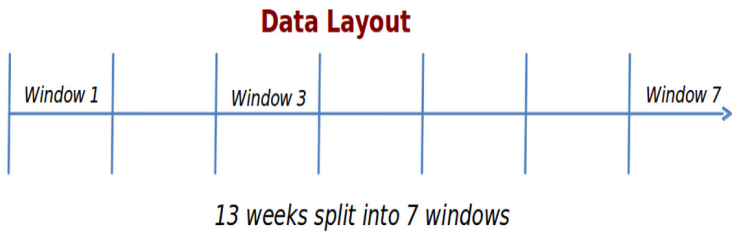

Figure 1: Illustrating Diagram for Temporal Separation of the Dataset

Note that each user and each job posting is randomly assigned to exactly one window.

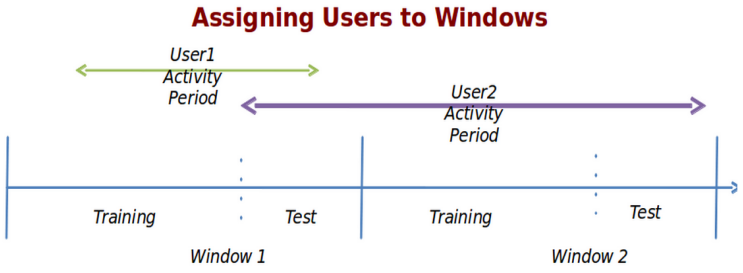

Figure 2:

Each job is assigned to a window with probability proportional to the time it was live on the site in that window. Each user is assigned to a window with probabilty proportional to the number of applications they made to jobs in that window. In the above image, User1 only made submissions to jobs in Window 1, and so was assigned to Window 1 with probability 100%. User2, however, made submissions to jobs in both Window 1 and Window 2, and so may have been assigned to either Window1 or Window2.

In each window, all the job applications that users in that window made to jobs in that window during the 9-day training period. And with each window, users have been split into two groups, *Test group* and *Train group*. The Test users are those who made 5 or more applications in the 4-day test period, and the Train users are those who did not.

For each window, the task of prediction is which jobs in that window the Test users applied for during the window's test period. Note that users may have applied to jobs from other windows as well, but the only thing needed to be predicted is which jobs they applied to in their own windows.

3.2 Preprocessing

Out of computational convenience, only the first part of dataset (WindowsID = 1) are involved in our experiment.

3.3 Traditional binary classifier

feature-based nearest neighbour model. behavior-based nearest neighbour model not available.

3.4 One-class learning solver (SVM)

3.5 Simple matrix factorization

Alternating least square

3.6 Inductive Matrix Completion

4 Experiments: Suitability Evaluation

We now start to divert our focus on investigating Application Potential to Suitability Evaluation.

References

[1] Prateek Jain and Inderjit S Dhillon. Provable inductive matrix completion. *arXiv preprint arXiv:1306.0626*, 2013. 2

[2] Nagarajan Natarajan and Inderjit S Dhillon. Inductive matrix completion for predicting gene–disease associations. *Bioinformatics*, 30(12):i60–i68, 2014. 2