A Preference-Based Restaurant Recommendation System for Individuals and Groups. Team Size: 3

Abstract

Using Yelp data, we built a restaurant recommendation system for individuals and groups. For each of 4.9k individual Yelp users, we create a ranking SVM model with features encompassing users' food preferences and dietary restrictions, such as cuisine type, services offered, ambience, noise level, average rating, etc. Our recommendation system for individual users achieved 72% average prediction accuracy. We propose a metric that maximizes minimum happiness across a group of users, as an alternative to other group recommendation systems where the most commonly recommended restaurant across individual users is selected.

1. Introduction

Inspired by our recent experience searching Yelp for restaurant recommendations, we built a restaurant recommendation system. Recommendation systems provide personalized, relevant recommendations to users and have been used in various domains, such as retail, movie-going, etc.

Currently, Yelp, the leading publisher of reviews of local businesses in the world, does not provide recommendations. Instead, users have to filter, sort, then read reviews to determine whether a business can provide them with what they want. A personalized recommendation system will provide a better user experience by incentivizing users to review and rate more in return for better restaurant recommendations; this in turn gives Yelp more data that can be used to further improve the recommendation system.

2. Problem Definition and Methods

2.1. Task Definition

We use data provided by Yelp [1], described further in the Data section, to build a recommendation system that provides personalized restaurant recommendations to users. Since different people have different food preferences and dietary restrictions, we perform careful feature selection to take advantage of the information reflected in a Yelp user's reviews.

Furthermore, since dining is frequently a communal activity, we generated recommendations not just for an individual user, but also for a group of users. This system needs to consider the information of all individuals in a group and recommend restaurants that satisfies the group of users according to some criterion.

2.2. Algorithms and Methods

We started by building a recommendation system for individual users using a ranking SVM [2], with the label being the user's rating of a restaurant on a scale of 1 to 5. For each user, using features on the restaurants they reviewed, we learnt a hyperplane that reflects their preferences. Hence, how happy the user is about the restaurant corresponds to the distance of the restaurant to their hyperplane. We evaluated our model on the test set by comparing the user's actual rating of the restaurant to the model's prediction.

2.2.1. Group Recommendations

Different users have different preferences and dietary restrictions. We wanted to help a group of users find a restaurant that all will like. We could try several methods: one is finding the restaurant that maximizes average happiness. In this case, the top ranked restaurant is given by $\arg\max_R \{\sum_{i=1}^{n} \mathbf{v}_R.\mathbf{w}_i\}$, where n is the number of people in the group, $\mathbf{w}_i$ is the unit normal vector for the i'th person's hyperplane and $\mathbf{v}_R$ is the normalized feature vector for restaurant R. Another method is to maximize the minimum happiness. In this case, the top ranked restaurant is given by $\arg\max_R \{\arg\min_{1\leq i\leq n} \{\mathbf{v}_R.\mathbf{w}_i\}\}$. The problem with the former approach is that one person in the group might end up very unhappy which is not a pleasant outcome, so we decided to implement the latter.

Both ideas depend on a measure that we called happiness which is a mixture of different features and is

Algorithm 1 Maximize Minimum Happiness

```
**Input:**
Group members $p_1, \dots p_n$
Corresponding hyperplanes: $w_1, \dots, w_n$
Restaurants $r_1, \dots, r_m$
**Output:** Sorted restaurants $R_k$
for $j = 1$ to $m$ do
  $R_j.id \leftarrow r_j.id$
  $R_j.score \leftarrow \infty$
  for $i = 1$ to $n$ do
    $R_j.score \leftarrow \min\{R_j.score, \langle w_i, r_j.features \rangle\}$
  end for
end for
return sort($R$) // Decreasing order on the score
```

different for each user. So we wanted to learn a user's hyperplane in the feature space in such a way that its unit normal vector corresponds to one unit of happiness for that user. Therefore, how happy that user is about the restaurant corresponds to the signed distance of the restaurant to his or her hyperplane. Algorithm 1 is the implementation of this method which runs in $O(nml)$ where n is the number of people in the group, m is the number of restaurants being considered and l is the total number of features. With $14k$ restaurants and 262 features and a manageable group size we can run this algorithm in an online setting. For the case which we have more data we can restrict restaurants to be ranked based on the geographical features.

2.2.2. Breaking Down Dimensions of Reviews

Different users value the various aspects of dining out differently. For instance, a "foodie" user who is able to tolerate subpar ambiance and service would derive much less utility from a 3-star review by a couple on date night compared to a 3-star review by a fellow "foodie". In order to better personalize recommendations to each user based on what the user values, we wanted to break down restaurant reviews along dimensions such as food, service, price, ambiance, convenience, etc.

This is existing work as shown in paper [4]. We hoped to categorize Yelp reviews to refine our features set. To do so, we needed to obtain a training set of reviews broken down into labeled dimensions. We have three options:

- Randomly select a subset of reviews, hand-code their dimensions, and construct a dictionary of words commonly associated with each dimension.
- Search the corpus of NLP research to locate a dictionary of words commonly associated with each dimension.
- Obtain reviews with already-labeled dimensions that [4] "spent 200+ man hours to annotate".

Using this training set, we then would have been able to break down reviews in our testing set along the same dimensions. This gives us more refined ratings for each restaurant – food rating, service rating, etc. – that can be used as additional features in our recommendation system.

Unfortunately, we did not have time to hand-code the dimensions of reviews. We reached out to the authors of [4] to share their hand-coded labels. However, they refused to share their data, so we were unable to complete this part of the study.

3. Experimental Evaluation

3.1. Methodology

3.1.1. Data

We used the dataset from Yelp's Dataset Challenge [1] containing 1,100k reviews of 42k businesses written by 190k users in five cities, namely Phoenix, Las Vegas, Madison, Waterloo in Canada, and Edinburgh in the UK, over 9 years from 2005 to present. The following information on businesses, users, reviews, check-ins, and tips is available:

- **Businesses:** location, hours, categories (e.g. "Italian", "cocktail bar"), attributes (e.g. free wi-fi, noise level, ambience, attire, "good for groups").
- **Users:** friends, fans, compliments received.
- **Check-In:** number of check-ins for each hour and day of week.
- **Review:** rating, review text, votes on whether the review is funny, useful, and/or cool.
- **Tip:** tip text, how many "likes" the tip got.

First we processed the data to link users to reviews and reviews to businesses. Some of the businesses are not restaurants so they were removed from the dataset. We then looked at the reviews to see how much information we can gain for a particular user. Users have a wide range of number of reviews:

- 50% of users only reviewed one restaurant
- 90% of users have less than 10 reviews

- 0.5% of users have more than 1k reviews.

If a user does not have a substantial number of reviews, we simply did not have enough information about them to make a good restaurant recommendation so we removed users with less than 20 restaurant reviews which left us with 700k reviews of 14k restaurants written by 4.9k users. Hence we have 4.9k ranking SVMs, one for each user.

Figure 1 below shows the range of numbers of reviews of all the users that we used our model on.

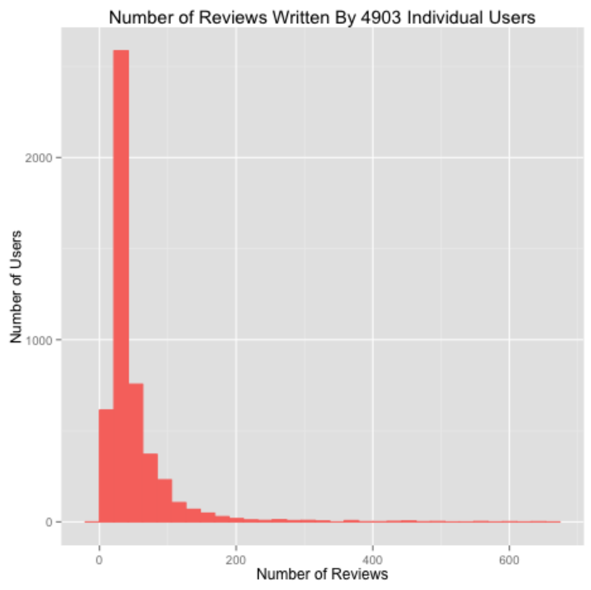

Figure 1. Distribution of reviews over users

3.1.2. Feature Enginering

We built a set of 262 features based on information available about a restaurant.

For each restaurant, we calculated its average rating across all reviews and the number of reviews it receives from all users and normalized these values to use them as features in our SVMs. This indicates the popularity of the restaurant among the whole population.

We also created binary features, one for each restaurant category and attribute. If there are fewer than 10 of 14k restaurants under a category, we did not include the category since the feature would have been very sparsely populated. The net result is 119 categories and 88 attributes. Figure 2 shows some of the most popular categories.

Lastly we used restaurant attributes encompassing cuisine type, services offered, ambience, noise level, etc.

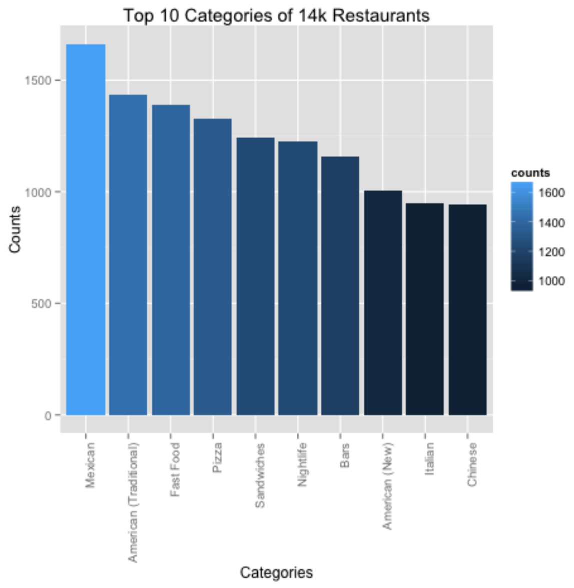

Figure 2. Top 10 Restaurant Types

Examples include free wi-fi, “Good for groups“, ambience, open 24 hours, parking, takes reservations, BYOB, price range, dietary restrictions, alcohol. We imputed missing data on restaurant attributes with 0.5 as the midpoint between having and not having the attribute.

3.2. Evaluation

We sorted the review sets for each user in increasing chronological order and setup a model evaluation pipeline that, for each user, selects the first 80% of reviews for training and retains the remaining 20% for testing. This reflects the way in which a recommendation system would actually be used where training is performed on data available up to a certain time.

Once we have the output for each individual, we can compare it to the true star rating. It is worth mentioning that we are comparing restaurants, so the number of errors are the number of restaurant pairs (i, j) where if s is the score output by the SVM, $s(i) < s(j)$ and the true rating of i by the user is higher that of j. This is basically the number of inversions in the permutation we output and we know the expected number of inversions in a random permutation is $\frac{n*(n-1)}{4}$ so our baseline is 50% since the number of pairs of restaurants is $\frac{n*(n-1)}{2}$.

3.3. Results

Starting with 4.9k users with at least 20 reviews, we obtained prediction accuracy with mean 72% and standard deviation 17%. Figure 3 describes their prediction accuracies. Our mean prediction accuracies are being significantly reduced by the 7% of users with prediction accuracy worst than the 50% baseline. This is shown in Figure 4 where we bucketed users by the number of reviews they wrote.

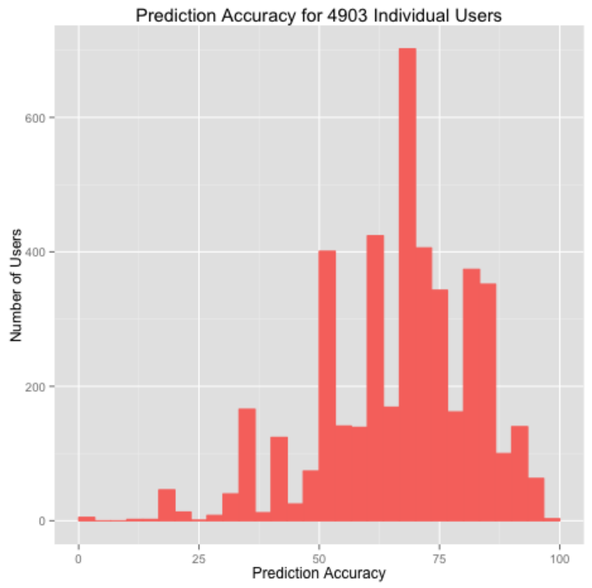

Figure 3. Distribution of prediction accuracy for users with at least 20 reviews

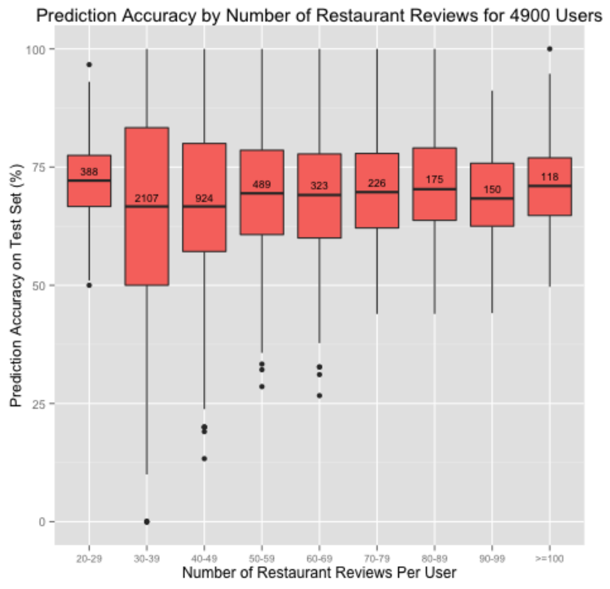

Figure 4. Accuracy for bucketed data

The more reviews a user wrote, the less variance in the prediction accuracy. This supports our suggestion that a restaurant recommendation system could be used to motivate users to write more reviews - with more reviews, the less variance in the predictions. If we look at only users who review at least 50 restaurants, our model returns prediction accuracy with much less variance and no outliers with extremely low prediction accuracies.

Interestingly, in Figure 4 we do not see a significant improvement in median prediction accuracy for users with larger number of reviews. This reflects a limitation on our current set of features, as even with a lot of training data, the model does not reach a prediction rate of 80%. However, this also means if we can improve the feature engineering process, it will result in an overall improvement across all users.

To refine our feature engineering process, we examine the most and least significant features for users with prediction accuracy at least 75%. Figure 6 describes the most significant features, namely features with highest absolute weight in the ranking SVM, and Figure 5 describes the least significant features.

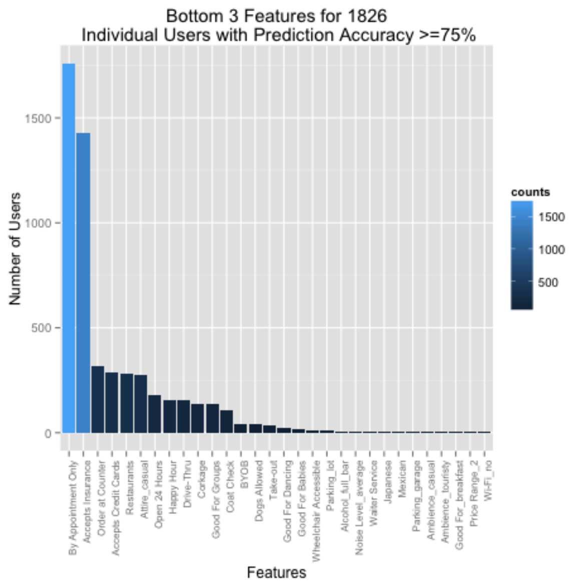

Figure 5. Distribution of bottom 3 features

The least important feature is "Restaurant" which is unsurprising because all of the businesses considered are restaurants. Other unimportant features make sense - it is unsurprising that a user would not care so much about whether a restaurant accepts insurance, has a coat check, or is good for dancing.

The most important feature is average rating which is not unexpected because most Yelp users decide whether to go to a restaurant by looking at its rat-

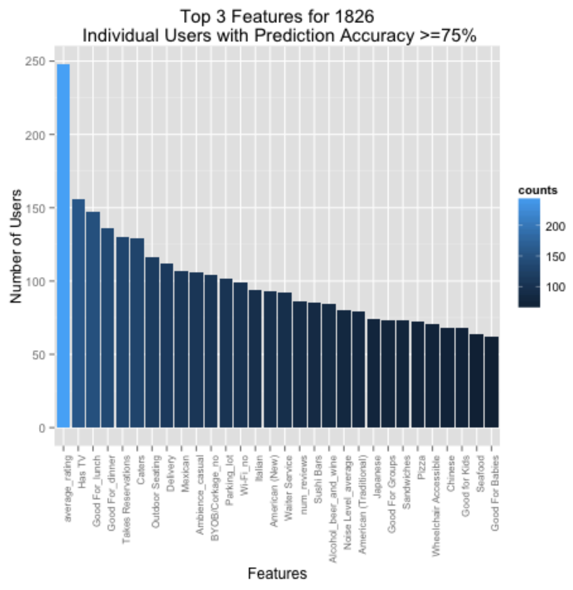

Figure 6. Distribution of top 3 features

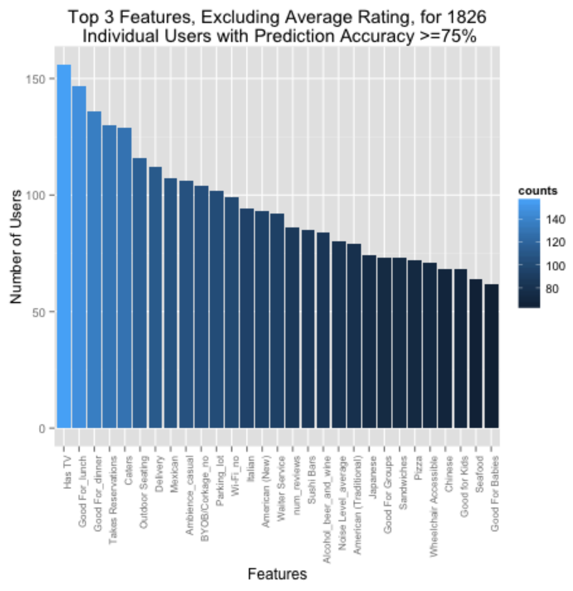

Figure 7. Distribution of top 3 features, excluding average rating

ing. Other important features includes whether the restaurant has delivery, popular cuisine types such as Mexican and Italian, etc.

There are a number of features such as WiFi availability, Wheelchair accessibilities, etc that occurred on both lists which means there are users who strongly consider these features in making their choices and there are others who do not. This implies that a universal recommendation system would not have worked as well as our individualized model, with one ranking SVM per user, because different users value different aspects of a restaurant differently, and have different food preferences and dietary restrictions.

	Initial		Iteration	
# Reviews	Mean	Std Dev.	Mean	Std Dev.
20-29	71.7	7.4	72.9	7.7
30-39	68.5	22.7	68.7	22.4
40-49	68.9	17.4	69.8	17.2
50-59	69.4	14.7	69.2	14.4
60-69	68.8	11.6	69.7	12.5
70-79	69.4	10.9	69.5	11.0
80-89	70.4	10.3	71.2	10.3
90-99	68.6	10.4	68.4	10.1
>=100	70.7	8.6	71.8	8.8

Figure 8. Prediction Accuracies for initial model vs. model with 20 least important features removed

To refine our model, we re-ran the ranking SVMs after removing 20 features of lowest absolute weight. The performance of the model is seen in Table 8.

Table 8 compares the 2 models - the initial model and the model with 20 least important features removed. The removal of those features generally improves the prediction accuracy for most users but does not change the standard deviation in any obvious way.

3.4. Discussion

Our model takes cares of the difference in individuals' preferences in selecting restaurants. It incorporates the most relevant factors in making prediction. However, even with 262 features the prediction accuracy does not reach 80% as expected. The main reason for this is the nature of the problem we are trying to solve and the difficulties encountered in engineering a good set of features. We are trying to predict people's choice of restaurants which is entirely subjective and dependent on a large number of factors that vary widely among different individuals.

By creating individual ranking SVMs for users, we remove the problem of inconsistent standards between users, where one user tends to give really high ratings and another user tends to give low ratings. We also remove the problem of different preferences and dietary restrictions, where one user loves sushi and another user really dislikes it. Hence, our approach allows individual preferences to be taken into account, both at the individual recommendation system stage, and later on at the group recommendation system stage, where our metric accounts for both the user who loves such and really dislikes sushi.

4. Use Case

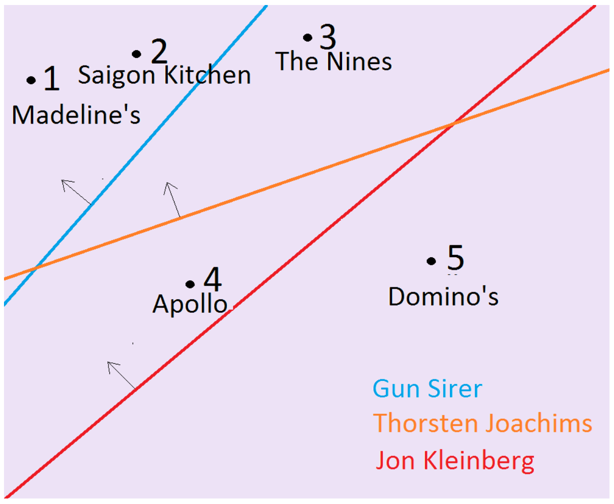

Figure 9. An example of group prediction

Figure 9 demonstrates how the group recommendation system works. Imagine the scenarios where 3 users with 3 different hyperplanes, color-coded as in the picture were to pick one of the five restaurants to go to. In this case, the ranking output by the algorithm is indicated by the number next to the restaurant with 1 being the most recommended and 5 being the least. Ranking of a restaurant is determined based on the minimum distance of the corresponding point to the 3 hyperplanes. For example even though Prof. Joachims prefers Saigon Kitchen to Madeline's which is reflected by the difference in the distance between these 2 points to his hyperplane, the algorithm uses Prof. Sirer's hyperplane to determine the ranking because this hyperplane is closer to the 2 points.

This idea has great potential to be incorporated into Yelp framework. Yelp users can log in with their accounts and form a group. This can be done easily since Yelp already keeps the information about a user's friends list. The calculation then can done online as it is a quick computation given that individuals' hyperplanes can be calculated beforehand. We do not expect a group of Yelp users to be very large so the run-time should not be a problem.

5. Related Work

In [6] there is some work relevant to what we proposed in this project. They used clustering methods to find clusters and use them for recommending restaurants in a cluster which has the highest rating. However they mention that this method does not work for this dataset since we do not have enough reviews to find good clusters. On the contrary, our algorithm works robustly on the data we have and achieves reasonable prediction accuracy.

6. Future Work

Since our data spans 9 years, an approach that can be taken is to put more weight on training examples of newer reviews. More work can be done to refine our feature set to enhance the accuracy of our predictions by studying users' reviews as discussed in [4] which requires manpower and time that we did not have at hand.

We did not take into account the case where users reviewed the same restaurant multiple times. This means they might be rating each of their visit experiences rather than the restaurant itself. If this is the case, we can take the average rating of the same user about a restaurant rather than counting all of the user's reviews separately.

7. Conclusion

Our recommendation system works to factor in Yelp users' different tastes in choosing a restaurant. Using specific information about each user we are able to predict with 72% accuracy their relative rankings of restaurants. With this model we can further investigate how effective different sets of features are at predicting users' choice of restaurants. This allows studies to be done on which factors are relevant and irrelevant to users in making their decisions which in turn reflects the kind of information Yelp should include in restaurant descriptions to enable users better utilize Yelp to figure out if a restaurant will delight them. Our proposal of a metric that maximizes minimum happiness reflects the real-world social situation in which many restaurant goers find themselves in. Such an approach to group recommendation systems provides an alternative to other group recommendation systems in which the suggested restaurant from each individual user's model is aggregated across a group of users, and the top restaurant is selected.

8. References

[1] Yelp, *Yelp Dataset Challenge*, `http://www.yelp.com/dataset_challenge`

[2] T. Joachims, *Optimizing Search Engines Using Clickthrough Data*, Proceedings of the ACM Con-

ference on Knowledge Discovery and Data Mining (KDD), ACM, 2002

[3] D. Cosley, J. A. Konstan, J. Riedl, *PolyLens: A Recommender System for Groups of Users*, Proceedings of the 7th European Conference on Computer Supported Cooperative Work, 2001.

[4] H. Sajnani, V. Saini, K. Kumar, E. Gabrielova, P. Choudary, C.Lopes, *Classifying Yelp reviews into relevant categories*, http://www.ics.uci.edu/~vpsaini/.

[5] T. Joachims, Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, B. Schlkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999. http://svmlight.joachims.org/

[6] A. Ihler et al., Recommender Systems Designed for Yelp.com http://www.math.uci.edu/icamp/summer/research/student_research/recommender_systems_slides.pdf