Recommender Systems for NYC Residential Community

Nicholas Souris
ns3205@nyu.edu

Yanhong Yang
yy1553@nyu.edu

Zeleng Zhuang
zz1135@nyu.edu

Abstract

In this paper we will present the process and algorithms that are used into creating a house recommender system for residences situated in the area of the city of New York. The algorithms used, will be compared and contrasted or even merged (thus creating new hybrid recommender algorithms) in an effort to magnify the accuracy of our system. The resulting data outputs from the usage and application of recommender algorithms, will be cleansed, indexed and filtered in such a way that the feasibility of a final merge, in an effort to converge to a single or a small group of accurate recommendations for a suitable residence, will almost be assured.

1 Introduction

Finding an apartment to rent or buy in New York city can be a daunting task and can very well become a source of deep frustration. Defining a suitable place of residence is a completely subjective matter which mostly has to do with the buyer's or renter's preferences. Living in an area with low crime rate or amongst a local populace with a specific affinity in regards to traditional family values or even in close proximity to high ranking restaurants can be some of the reasons why a person would choose to live in a certain area over another. Coupling the vast amount of information required to choose where to live with the high housing prices and large area diversity of a metropolitan area such as New York city, it is fairly easy to recognize how recommending a specific house to a potential buyer or renter can become a disheartening job.

Our project aims to use historical data regarding housing prices along with several other parameters that might affect the standard of living of residents, in an effort to create a recommender system for NYC residential communities based on zip code.

We will collect data including housing prices, income, safety, infrastructure and other commonly used factors in evaluating a residential community. We will attempt to discover all or most of the necessary techniques which are vital for the practical implementation of a housing recommender system. Given that the timeframe in which we need to implement this system is large enough, a platform will be built for the recommender system for the purpose of collecting data from anonymous volunteer users that are willing to provide information to enrich our database. In this paper, the motivation, research and the thought processes that led to the final design will become transparent and brought to the foreground in an effort to substantiate the accuracy and efficacy of our analytics project.

2 Motivation

Our project is motivated by the real-world problem of finding a living place in a metropolitan city such as New York City. Different people definitely have diversified demands on housing, such as how much they would like to pay, how convenient the place is for dining, shopping, day care, etc.

Our project manages to collect closely related information such as housing price or rent, demographics, crime rate, facilities such as restaurants, stores, preschools and so on. Based on the comprehensive information collected, through a website user interface, our project aims to recommend a list of zip codes such that the corresponding places optimally match the demands from housing-seekers. Our project would interest and benefit home buyers and renters.

3 Related Work

There are three main categories of recommendation methods: content-based, collaborative, and hybrid recommendation approaches. Content-based systems recommend items similar to the ones that a user preferred in the past, collaborative systems recommend items that people with similar tastes and preferences liked in the past, while hybrid recommender systems by combining collaborative and content-based methods can avoid certain limitations of content-based and collaborative systems. [1] has presented an overview of the field of recommender systems.

In content-based recommendation methods, the utility $u(c,s)$ of item s for user c is estimated based on $u(c,s_i)$ where items s_i are "similar" to s and usually defined as:

$$u(c,s) = \text{score}(ContentBasedProfile(c), Content(s)),$$

where $ContentBasedProfile(c)$ is the profile of user c containing tastes and preferences of this user, and $Content(s)$ is an item profile consisting of attributes characterizing item s. In information retrieval-based paradigm of recommending Web pages, Web site URLs or news messages, both $ContentBasedProfile(c)$ and $Content(s)$ can be represented as *term frequency/inverse document frequency* [9] (TF-IDF) vectors $\overrightarrow{\omega}_c$ and $\overrightarrow{\omega}_s$ of keyword weight; and utility function $u(c,s)$ is represented as the cosine similarity measure [2], [9]:

$$u(c,s) = \cos(\overrightarrow{\omega}_c, \overrightarrow{\omega}_s) = \frac{\overrightarrow{\omega}_c \cdot \overrightarrow{\omega}_s}{\|\overrightarrow{\omega}_c\|_2 \times \|\overrightarrow{\omega}_s\|_2}. \qquad (1)$$

In collaborative filtering systems, the utility function $u(c,s)$ of item s for user c is estimated based on $u(c_j,s)$ of item s for users c_j who are "similar" to c. Various algorithms have been developed to make rating predictions, in general grouped into two classes [3]: *memory-based* (or *heuristic-based*) and *model-based*. For memory-based algorithms, the value of an unknown rating $r_{c,s}$ for user c and item s is usually computed as an aggregate function of the ratings $r_{c',s}$ of the most N similar users to c for the same item s. A lot of approaches have been applied to compute the similarity $sim(c,c')$ between two users, most of which are based on their ratings of items that both users have rated; the two most popular approaches are correlation and cosine-based. The Pearson correlation coefficient used to measure similarity is defined as follows [8], [10]:

$$sim(c,c') = \frac{(\gamma_c - \overline{\gamma}_c) \cdot (\gamma_{c'} - \overline{\gamma}_{c'})}{\|\gamma_c - \overline{\gamma}_c\|_2 \times \|\gamma_{c'} - \overline{\gamma}_{c'}\|_2}. \qquad (2)$$

Besides the heuristic rules, statistical and machine learning techniques have also been applied to learn a model. For example, [3] proposes a probabilistic approach calculate the unknown ratings:

$$r_{c,s} = E(r_{c,s}) = \sum_{i=0}^{n} i \times \Pr(r_{c,s} = i \mid r_{c,s'}, s' \in S_c), \qquad (3)$$

where rating values are assumed to be integers between 0 and n and the probability expression is the probability that user c will give a particular rating to item s given user c's ratings of previously rated items S_c. In order to estimate the probability, [3] also proposes two probabilistic models: cluster model and Bayesian networks.

In [1], the methods of combining collaborative and content-based systems are classified into four classes:

1. implementing collaborative and content-based methods separately, and combining their predictions;
2. incorporating content-based characteristics into a collaborative approach;
3. incorporating collaborative characteristics into a content-based approach;
4. constructing a general unifying model that incorporates both content-based and collaborative characteristics.

Moreover, [1] describes various limitations of current recommendation methods and discusses possible extensions that can improve recommendation capabilities.

4 Design

Our project consists of four stages See below the design.

Stage I: Scrap data from websites. This is done mostly by using Scrapy and partly by using Java. We scrap demographic information and housing price from city-data, crime rate from bestplaces.net, data regarding restaurants, stores, clinics and preschools from Yelp.

Stage II: Clean and aggregate data. We apply various Hadoop components such as MapReduce, Hive and Pig. First, we filter seemingly invalid data and remove redundant data. For example, when scrapping restaurant data from Yelp, there could be no address sometimes. In that case, we cannot tell which zip code the corresponding restaurant locates in. Since our recommendation is zip code-based, the address-missing data is useless to us and should be cleared away. Also it happens that the same piece of data can be scrapped multiple times, and the redundancy should be resolved. Second, we apply various metrics to evaluate many kinds of features of a zip code. For example, the statistical demographic data such as the percentage of male/female, the ratio of single/married and so on consist of the demographic vector. Based on the number and users' average reviews of clinics within a zip code, we provide a rating between 0 and 1. Similarly we obtain ratings of a zip code with respect to security, shopping, restaurants and preschools, which consist of the rating vector.

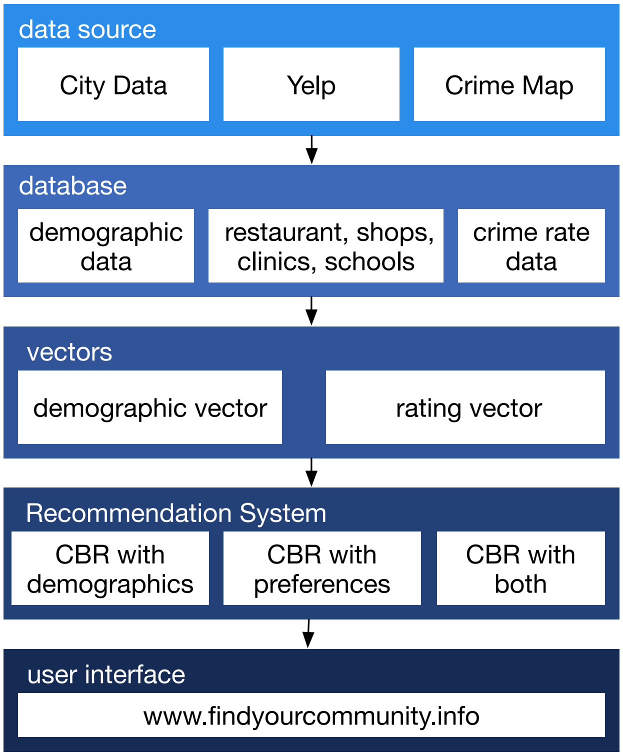

Figure 1: Design of Recommender System

Stage III: Recommend with three approaches:

1. Content-based recommendation with demographics. The information such as gender, marriage status, education level and so on, obtained from the user through user interface, is converted a vector representing that user. We recommend those zip codes whose demographic vector is most close to the user vector when measured by angle.

2. User-based collaborative filtering. We collect a user's preferences regarding security, dining, shopping and so on, which consists of a weight vector. Then we take inner product of the weight vector and the rating vector of each zip code. We recommend those zip codes that maximize the inner product.

3. Hybrid recommendation of the above two. We recommend those zip codes such that the sum of two scores obtained from the above two recommendation systems reaches maximum.

Stage IV: Build a website as user interface. This interface serves two purposes: on one hand, it provides an arbitrary user the opportunity of experiencing and examining our analytics project; on the other hand, with the feedback from user experiences, we can evaluate how precise our analytics project is and may improve it in the end.

5 Results

(Future? In this section, you can describe: Your experimental setup/issues with data/performance/etc. Describe your experiments, describe what you learned. Did you prove or disprove your hypothesis? Were some results unexpected? Why?)

6 Future Work

(Future? Given time, how would you expand your analytic? Could it be applied to other areas? Etc?)

7 Conclusion

(Future? One or two paragraphs about the value/accuracy/goodness of your analytic.)

8 Acknowledgments

We would like to thank Professor Suzanne McIntosh for giving us advice on how to choose and compare different recommendation mechanics. We would also like to thank Amazon Web Services and Cloudera for running our datasets on their platforms. (This section is optional. It can be used to thank the people/companies/organizations who have made data available to you, for example. You can list any HPC people who were particularly helpful, if you used the NYU HPC.)

References

[1] G. Adomavicius and A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. Knowl. Data Eng. 17*, 6 (June 2005), pp. 734–749

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern information Retrieval.* Addison-Wesley, 1999.

[3] J.S. Breese, D. Heckerman and C.Kadie. *Empirical Analysis of Predictive Algorithms for Collaborative Filtering.* In Proc. 14th Conf. Uncertainty in Artificial Intelligence, July 1998.

[4] J. Dean and S. Ghemawat. *MapReduce: Simplified data processing on large clusters.* In proceedings of 6th Symposium on Operating Systems Design and Implemenation, 2004.

[5] A. Gates. *Programming Pig.* O?Reilly Media Inc., Sebastopol, CA, October 2011.

[6] S. Ghemawat, H. Gobioff and S. T. Leung. *The Google File System.* In Proceedings of the nineteenth ACM Symposium on Operating Systems Principles – SOSP '03, 2003.

[7] M. J. Pazzani and D. Billsus. Content-based recommendation systems. IN P. Brusilovsky, A. Kobsa, and W. Nejdl (Eds.): *The adaptive web,* LNCS. 4321, pp. 325–341, 2007.

[8] P. Resnick, N. Iakovou, M. Sushak, P. Bergstrom, and J. Riedl. *GroupLens: An Open Architecture for Collaborative Filtering of Netnews.* In Proc. Computer Supported Cooperative Work Conf., 1994.

[9] G. Salton. *Automatic Text Processing.* Addison-Wesley, 1989.

[10] U. Shardanand and P. Maes. *Social Information Filtering: Algorithms for Automating "Word of Mouth".* In Proc. Conf. Human Factors in Computing Systems, 1995.

[11] T. White. *Hadoop: The Definitive Guide.* O?Reilly Media Inc., Sebastopol, CA, May 2012.