Event Recommendation System

Jun Zhou, Bobi Pu
junzhou@usc.edu, bobi.pu@usc.edu

Abstract

Recommendation system plays an important role in social network services where users create and share events, like facebook, twitter, etc. In this project we are trying to predict which events the users will be interested based on real-world datasets obtained from an online competetion host. We use several classification based models to solve the problem. We eveluate and compare their precision and recall. The results show that in this senario, non-paramatric models like decision trees and random forestes tend to outperform paramatric classifiers like logistic regression and SVM. We also show that user-event relationship based features play an important role in the models, and resample techniques can improve the performance when dataset is highly skewed.

1 Introduction

Users and events are the two most important factors in a social network services. Users create events and post to their profile page or distribute them through the network. They can also invite friends to the event. People see this event can show their preferences by like, maybe going, or dislike.

Help social network users to find the events they are interested in is important to improve the user experience. Based on the user interests and the event information, we might be able to predict whether a user is interested in a event, thus recommend the most relevent events to the user, and help users to enjoy their experiences with the services.

In this project, we are going to build an event recommendation system for a social network service to predict what events users will be interested based on user actions, event metadata, and demographic information. The project is originated from an online competetion[7].

We plan to use several data mining algorithms to build the system. We'll compare the accuracy and performance of those algorithms for this system, and come up with the best solution.

The results will give insights into specifically how social network will affect people's decision. Additionally, the results will also illuminate whether friends' choices or location preference is a more important factor of affecting people making decisions.

2 Datasets

The data we have includes user profiles, friendship relations, event information and event attendees records. There data are in seperate CSV files. There is not too much pre-processing work to do, what's needed is to extract userful information from those CSV files and formulate features for our training algorithms.

The training data contains whether a user is interested in a certain event, whether he is invited or not, and also when did the user see this event information. There are 15398 training examples.

The test data contains the same information as the training data, except for the interested or not information.

User profile data contains demographic data of the user which is identified by an unique ID, includes user gender, birthday, location, when did the user join the website, timezone, locale. There are 38 thousand users.

Event data contains detailed information of events, including time, location as well as the user who creates the event. Each event also have an list of categorized keyword counts, there are 100 categories, which may represent the literal information of the event. There are more than 3 million events in total.

Friendship relation data gives us all the friends of a certain user, which can help us predict the users' interests from this graphic model.

Event attendees records contains information about which users attended various events, including users who indicates that they are going, maybe going, invited to, or not going to the event.

Detailed dataset information is in the following table.

data file	**data fields**
training	userid, eventid, invited, time of user sees the event, interested, not interested
test	userid, eventid, invited, time of user sees the event
user	userid, locale, birtyear, gender, joined time, location, timezone
event	eventid, create userid, start time, address, latitude, longitude, keywords list.
user friends	userid, friends id list
event attendees	eventid, list of users going, maybe going, invited, or not going.

Table 1: Dataset information

3 Related Work

There are several models can be used for this problem. Here we'll discuss collaborative filtering and classification based methods.

3.1 Collaborative filtering

Collaborative filtering is commonly used in recommendation system, like Amazon shopping recommendation [1], Netflix movie recommendation[2] etc. User-based and Item-based collaborative filtering is heavily based on the assumption that an effective similarity measure can be built among user/item groups. In practice, it usually means that for those models to be useful, there must be reasonable overlapping between transactions of different users/items.

Unfortunately, for the datasets we have, this assumption does not hold. Due to the lack of overlapping between user/event pairs, feature matrix will be too sparse to make useful recommendations. Even worse, some users/events in the test set have never appeared in the training set this is usually called anonymus recommendation and almost equivalent to a blind guess.

3.2 Classification

From another point of view, the basic idea of this problem is, given a user-event pair, try to predict whether the user is interested in the event or not, based on all the user information and event information. So this could be modeled as a binary classification problem, and there are many classification based models can be used to handle the problem. we decide to try several classification models on it.

3.3 Compare of classification models

The classifiers we use include:

1. Decision Tree.

2. Logistic Regression.

3. Random Forest.

4. SVM.

Generally speaking, those four models are all widely used, and one might out perform the other in different situations. We decide to try all of them, and find out which one most fits this problem.

Decision tree[5] is Non-parametric, so we don't have to worry about outliers or whether the data is linearly separable. The main disadvantage is that it could easily overfit, especially when there are too many features.

Random Forest[6] operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes output by individual trees. It also has the problem of overfitting.

SVM[4] supports different kernels, like linear kernel, rbf kernel etc. Each kernel reflects an assumption about the seperation of the data. With SVM, there are also many other parameters needs to be tried in order to come up with the best solution.

Logistic Regression[3] can provide class probabilities, and also can choose kernel methods. Compared to SVM, logistic regression classifier is defined by all the data points, while SVM is defined only by the support vectors.

4 Methodology

In this section, we will discuss the tools we used, the data processing procedure, the featrure extraction, and the problems we have encountered along with the solutions.

4.1 Tools used

We use scikit-learn[8] python package to build the model. It provides versatile tools for data mining and analysis in the field of science and engineering, includes almost all the widely used machine learning algorithms.

we also use the geopy[9] package to translate the user/event address into latitude and longitude, and then calculate the distance between the user and event.

4.2 Data pre-processing

Although the data provided by the host is already cleaned and formated, we still need some preprocessing work, since the event information data contains more than 3 million records, and we can't simply load all the data into the main memory of our laptops.

We tried two different ways to solve this problem. First, we try to load the data into a database, then query the data from the database when needed. This slows down the calculation a little bit, but it works well.

Another method is based on the find that only a small portion of the whole events data appear in other datasets, which means most of the events are useless, so we extract those usefull events, which fits into main memory well, and do the following work based on it.

Another part of the pre-processing is to convert the user address into latitude and longitude. We use geopy package to do this.

4.3 Feature Selection

The ultimate goal of the problem is, given a user-event pair, try to predict whether the user is interested in this event. Intuitively, there are three types of features we can make use of. User-based features; event-based features; and user-event relationship based.

User-based features are primarily the user information, includes user age, gender, time when the user joined the website, user location. For event-based, it's similar, the time and location of the event, the creator of the event, and also the keyword list.

4.3.1 user-based and event-based features

For these two types, we have some processing on the original data.

First, for the start time of the event. The original start time is a timestamp which represents the exact time. However, after some tests we found out that the actual start time in hour is a more useful feature. For example, if the event start at 10:00 AM in the morning, most people will be at working that time, so few of them will pariticipate. If the event starts at 8:00 PM, then maybe more will go since it's free time. Also the week day makes different. On weekends more people will be willing to participate in some activities.

Another processing we do is on the keywords list of the event. The original data contains a list of 100 numbers, each number represents the count of one category of keywords which appear in the event discription. Have so much features not only takes more computing resources, but also may cause some overfitting problems. So we use Kmeans to cluster these events based on the lists. After that, each event is assigned with a cluster ID, to represent all the keywords list.

4.3.2 user-event relationship based features

What's more important is the user-event relationship based features. These information will be important to represent the possible relations of the user-event pair. The features we come up with includes the following.

1) The time difference between the start of the event and when the user see the event. This is usefull since if the user only see the event information after the event is finished, he just can not attend.

2) The distance of the user address and the event location. As stated before, we use geopy to first convert the address to latitude and longitude, and then calculate the distance.

4.3.3 overall feature list

At last, we use the following features in our model:

- user age
- user gender
- user join time
- user invited or not
- user is friend of the event creator or not
- event start time in hour
- time difference between the user sees the event and the start of the event
- how many friends will go to the event
- how many people are invited to the event
- how many people will go to the event
- how many people maybe go to the event
- how many people will not go to the event
- distance between location of the user and location of the event
- event clustering categories

4.4 Problems and Solutions

There are two major problems we have encountered.

First, the training dataset is highly skewed, over 80% are not interested. We tried resample to solve the problems. Results show that resample can improve the overall performance about 2

Second, there are much missing data. For example, many users and events don't have address information available. The basic solution is to assign a default value. Here for the missing address, we just assign a big distance value for the user and event pair. This is not a good choice, we plan to try random forest to solve the missing data problem.

5 Experiments

With 12000 training samples, 3397 test samples, we tried several models with different parameters. The results are in Table 2-5.

From the results we can see that, due to the data skew, Logistic regression tends to predict the events as not interested, and the interested tests have very low scores. We have tried both L1 and L2 normalizer, both have similar results.

SVM without class weight adjustment just as poor as logistic regression. With class weight, it just trades precision for recall. Rbf kernel performs a little better than linear kernel. We still need to fine tuning the parameters to find the most suitable ones.

Decision trees and random forest do better job than logistic regression and SVM. Using random forest with 10 estimators, the overfitting is worse than just decision tree, due to the skewed data.

For all the classifiers, resample improves about 2%.

During the tests, we also found other features which improves the performance. For the location calculation between user and events, we first try a basic method, just binary match of the address name. After we use geopy to calculate the distance of the addresses, it improves about 4%.

	No resample			**After resample**			
item	precision	recall	f1-score	precision	recall	f1-score	sample-num
interested	0.81	0.76	0.78	0.81	0.76	0.79	2587
notinterested	0.35	0.42	0.38	0.36	0.44	0.40	810
avg/total	0.70	0.68	0.69	0.70	0.68	0.69	3397

Table 2: Decision Tree

	No resample			**After resample**			
item	precision	recall	f1-score	precision	recall	f1-score	sample-num
notinterested	0.79	0.95	0.87	0.79	0.94	0.86	2587
interested	0.58	0.21	0.31	0.52	0.20	0.29	810
avg/total	0.74	0.77	0.73	0.73	0.77	0.72	3397

Table 3: Random Forest, 10 estimators

	No resample			**After resample**			
item	precision	recall	f1-score	precision	recall	f1-score	sample-num
notinterested	0.76	0.99	0.86	0.80	0.94	0.86	2587
interested	0.38	0.01	0.03	0.55	0.23	0.32	810
avg/total	0.67	0.76	0.66	0.74	0.77	0.73	3397

Table 4: Logistic Regression

	linear kernel			**rbf kernel**			
item	precision	recall	f1-score	precision	recall	f1-score	sample-num
Notinterested	0.80	0.63	0.59	0.80	0.55	0.66	2587
interested	0.27	0.47	0.38	0.29	0.57	0.38	810
avg/total	0.68	0.51	0.54	0.68	0.56	0.59	3397

Table 5: SVM, class weight=1:2.65

5.1 Future work

Right now we just compare different models on the dataset, in the next step, we will try to combine different models together into a hybrid model to improve the results.

Mean while, for each model, we still need to do more cross validation test to find out the most suitable parameters.

On the other side, there are still features we can try. For example, for each event, we can try to cluster the age of the attendees, to check whether certain age period will have a special interests. Same method can be used on the gender information.

What's more, we can try some graph based methods, since in a social network, people tends to cluster to form some kinds of connected sub graphs, each sub graph may have special interestes in different kinds of events.

6 Conclusion

In this project, we use different models to predict which events the users will be interested in a social network service. The results shows that non-paramatric models like decision trees and random forestes tends to outperform paramatric classifiers like logistic regression and SVM. We also shows that user-event relationship based features play an important role in the models, and resample techniques can improve the performance when dataset is highly skewed.

7 References

[1] Linden, Greg, Brent Smith, and Jeremy York. "Amazon. com recommendations: Item-to-item collaborative filtering." Internet Computing, IEEE 7.1 (2003): 76-80.

[2] Zhou, Yunhong, et al. "Large-scale parallel collaborative filtering for the netflix prize." Algorithmic Aspects in Information and Management. Springer Berlin Heidelberg, 2008. 337-348.

[3] Hosmer, David W., and Stanley Lemeshow. Applied logistic regression. Vol. 354. Wiley-Interscience, 2004.

[4] Hearst, Marti A., et al. "Support vector machines." Intelligent Systems and their Applications, IEEE 13.4 (1998): 18-28.

[5] Safavian, S. Rasoul, and David Landgrebe. "A survey of decision tree classifier methodology." Systems, Man and Cybernetics, IEEE Transactions on 21.3 (1991): 660-674.

[6] Liaw, Andy, and Matthew Wiener. "Classification and Regression by randomForest." R news 2.3 (2002): 18-22.

[7] http://www.kaggle.com/

[8] http://scikit-learn.org/stable/

[9] https://code.google.com/p/geopy/

[10] CS599 lecture notes by Prof Yan Liu, USC.