Recommending Accounts to Twitter Users

Siddharth Vijayakrishnan (svijaykr1@gmail.com)

Introduction

Twitter is a service that allows users to post short messages (140 characters) to their profile page. It also allows users to "follow" other users whereby a user can see updates from the users he follows on his twitter page. People usually follow friends (people they know in real life) or accounts whose updates they find interesting. Today, Twitter has more than 100M users and more than 190 M visitors monthly. Therefore, it is not easy to find interesting accounts to follow.The objective of this paper is to recommend accounts to users that they might find interesting

Data Collection

Twitter has an open API that allows anyone to get a list of a user's friends (provided the account is not private) It is therefore easy to create a graph of the network. Since there are more than 100M nodes in this graph with many times that many edges, it requires a lot of computational power to process this entire graph. I therefore propose to focus on a smaller subset. However, recently Twitter has been more circumspect in allowing unfettered access to the entire social graph and tweet stream. It allows this access termed the "fire hose" to a small chosen set of companies only. Through the public API, one can only access a single user's tweet stream and his profile information and also the public timeline of tweets. However, I was fortunate enough to discover a recent data set compiled by *Kwak, Lee and Park[1]* which contains over 41 M user profiles and links back to their real user names. This allowed me to supplement the data set (which contained only followers and following for a user) with other features like the recent stream of tweets and user profile information.

Inferring Edge Strength

In the simplest setting, a user being connected to another user can be used as a preference signal. In recent times, given the explosive growth of twitter, there have emerged a large number of "bot" accounts that seek to follow as many users as possible in the hope that unwitting users will follow them back. Therefore, looking at "followed" accounts yields more information about the account holder's preferences rather than "follower" accounts.

In a traditional item recommendation setting, users rate items on a scale of 1-5 or by an up or down vote. In twitter, there is no explicit rating of accounts by other users. However, we can construct some implicit signals from the user's content stream that are analogous to recommendation. Specifically, I look at three signals that are counted as up votes. First, if a user follows another account, that is considered a positive rating for the account that is followed. Second, if a user retweets (i.e. echoes a tweet to his own tweet stream), that can also considered a positive rating. Thirdly, if a user shares a "hashtag" with another user, that is considered a positive rating for the user who is being followed.

Sharing a hashtag implies that the two tweets are related to the same topic, although they may express two entirely different opinions (for e.g. the recent controversy around wikileaks elicited a storm of either vehement approval or disapproval from twitter users, but they used the same #wikileaks hashtag).

Preparing the Data

The data set contains information on X million profiles. Profile information is limited to the user accounts followed by the user. Since the data set is as of a certain date, the information is not complete as of today. However, for my purposes, the data set contained enough information to create training and test data sets.

Since the data set represents a graph, to construct a utility matrix, I picked 500 random accounts (including my own, to get a sense of whether the predicted accounts make any sense) and recursively pick accounts three steps removed (snowball sampling). In social networking theory, these are known as friends of friends and it has been shown that the degree of influence drops off significantly after three steps. The sample thus collected contained information on over 25,000 profiles. For each of the 500, I divided the follower information into 2 groups. 70% of the followers went into the training data set and the remaining 30% went into the test data set.

Profile Similarity

The first step in creating a user profile is to collect all accounts followed and all accounts followed by those accounts. They are arranged in a matrix as follows

User	U_{11}	U_{12}	....	U_{1N}
U_{11}	1	1	0	1
U_{12}	0	1	1	0

Fig 1

In the figure above, an element A_{ij} = 1 if U_{1i} follows U_{1j} , 0 otherwise

Note that since relationships are not always symmetric, it could be the case that U_{11} follows U_{12} but not vice-versa. In this case, the value of the element representing the interaction is set to 0. As referred to above, there are some "bot" accounts that follow many thousands of users. Since this introduces noise into the data and makes the computation of similarity difficult, I leave out any account that has follows more than 300 users from the graph. 300 is again arbitrarily chosen, but it not unreasonable to impose a limit, since intuitively, a user who follows more than a certain number of other users will find it difficult to maintain strong interactions with all of them.

To the representation above, I add the the number of times a user has replied to another user in his list of followed accounts. Since twitter allows an open communication structure, where any user can directly respond to another user simply by using the @username convention, there are a lot of tweets that are addressed to people not in the immediate network. For some users, this was as high as 50%. I ignore all tweets that are not sent to an account which is not in the extended network. Similarly, if two users share a hashtag, that gets another vote. If a user "retweets" (i.e. echoes a tweet published by another user into his own tweet stream), that gets another vote. It is conceivable that each of these methods of interaction do not count the same. However, to keep the model simple, I simply assign the same score (1) to all of them.

Tweet Similarity

Another method of constructing a profile for the user is to analyze the stream of tweets published by the user. These may include tweets originating from the user or tweets originating from others users (retweets). It is possible to use several IR techniques to analyze the content of the stream and classify them by topic. I however, opted to take the easy way out and relied on the excellent service provided by OpenCalais online. This service takes in arbitrary text (individual tweets in this case) and provides a list of topics that the tweet belongs to. Since the list of topics is quite small, a matrix similar to the one above can be constructed, where each element represents the number of tweets that pertained to that topic

Evaluation Metric

In the past, most recommendation problems have considered using RMSE between the actual and predicted values as an evaluation metric. While the RMSE makes intuitive sense in an item rating scenario, it doesn't make much sense in a binary classification (interesting v/s not interesting setting) such as this. Therefore, I opted to use the top N recommendation metric to evaluate the performance of various algorithms. The metric is simple. For each account that I am making predictions for, I sort of the related (all accounts three steps removed) in order of similarity. Then I choose N and check how many are actually present in the test data set for that user.

The evaluation metric M =

$$\sum_{i=1}^{3} \{1 \, if \; predicted \; user \; is \; present, 0 \, otherwise)$$

I chose the value of N to be 3 somewhat arbitrarily (since that is the number of profiles twitter currently recommends to a user). Under this metric, if M is 0, then it implies that the algorithm failed to provide a single good recommendation. Since it is meaningless to recommend users already being followed to the target user, I exclude the first level of accounts while coming up with the top N list of similar users.

Recommendation Algorithms

I implemented two simple and well known methods of constructing recommendations. The first termed collaborative filtering relies on finding similar items by looking at the number of items on which two users' ratings agree or disagree. In our setting, this translates to the number of user accounts that are followed by both users as a proportion of the total number of users that are followed by the pair. The second approach is to cluster users based on some measure of similarity. I attempted both methods using both profile similarity as well as tweet similarity. I believe this model can be extended to include both similarity measures into a single normalized metric, but I did not experiment with this approach.

Jaccard Distance

Jaccard Similarity S (U_i and U_j) = $$Ui \cap Uj \div \sum_{i=1}^{N} Ui \sum_{j=1}^{N} Uj$$

Jaccard Distance D = 1 - S

Cosine distance

Cosine . Similarity C = U_i . U_j / || U_i ||. || U_j ||

Clustering

I tried to group users into target clusters based on both their profile similarities as well as their tweet similarities. After the target clusters were formed, I picked the closest N (equal to 3) users who were part of the target user cluster, who were already not connected to the target user. I investigated the use of both k-means and hierarchical clustering and decided to use the hierarchical clustering technique since it was not clear *a priori* how many clusters needed to be present. For the similarity measure, I used cosine similarity.

Experiment Results

The average evaluation metric obtained on the different combinations were as follows

Algorithm	Average Prediction Accuracy
C.F using Jaccard Distance/Profile Similarty	0.84
C.F using Cosine Distance/Profile Similarity	0.94
C.F using Jaccard Distance/Tweet Similarity	0.94
C.F using Cosine Distance/Tweet Similarity	0.91
Clustering using Profile Similarity	0.79
Clustering using Tweet Similarity	1.02

As can been seen from the table above, the best performing algorithm on the experimental data set was clustering using tweet similarity. Overall, even though I expected better results from all algorithms, it is heartening to note that only in about 30% of the cases, did the algorithm fail to make even a single correct prediction. Since even a single correct prediction can increase the value of the service to the user, I consider this a positive result.

Further Work

The two approaches represent a rather simple method to calculate good recommendations. There are several other more sophisticated methods that seem to have yielded better results in practice. In particular, I found three methods to be particularly promising. The first is using Singular Value Decomposition as detailed in [2]. As [3] points out, using this method appropriately resulted in a 7% reduction in RMSE on the Netflix prize data set. The second is using Restricted Boltzmann Machines as detailed in [4]. The final prize winning entry in the Netflix challenge used a combination of the two approaches detailed above along with accounting for temporal bias, which is important as a user's tastes evolve. Finally, a recent set of papers [5] and [6] detail a promising approach that models the social network as a directed graph and tries to learn the edge weight using a supervised random walk with restarts approach. The authors

of the paper applied this technique to predicting links in the Facebook social graph and produced good results. This looks particularly promising since the Facebook link graph is similar to the Twitter graph in many ways.

Conclusion

To summarize, I tried to recommend twitter accounts to follow to users of the service by utilizing their existing set of connections. To do this, I created a set of user profiles and tweet profiles for 500 users and their connections 3 steps removed. I then applied two well known recommendation algorithms and found clustering based on the content of a user's tweets to be more effective. I then studied a few more sophisticated algorithms, but due to time considerations did not implement any of them.

References

[1] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon "*What is Twitter, a Social Network or a News Media?* " Data downloaded from the torrent file found on http://an.kaist.ac.kr/traces/WWW2010.html

[2] Sarwar, Badrul and Karypis, George and Konstan, Joseph and Riedl, John "*Analysis of recommendation algorithms for e-commerce*"

[3] Rajaraman, Anand and Ullman Jeffrey D "*Notes on Recommendation Systems*" http://infolab.stanford.edu/~ullman/mining/2008/slides/RecommendationSystems.pdf

[4] Salakhutdinov Ruslan, Mnih Andriy, Hinton Geoffrey "*Restricted Boltzmann Machines for Collaborative Filtering* "

[5] Agarwal, A., Chakrabarti, S., & Aggarwal, S. (2006). "*Learning to rank networked entities.* "

[6] L. Backstrom, J. Leskovec. "*Supervised Random Walks: Predicting and Recommending Links in Social Networks* "