Predicting Tastes from Friend Relationships

Chris Bond and Duncan Findlay

December 12, 2008

1 Introduction

In the last few years, online social networks have become an important part of people's lives. They have made data describing the connections between people more accessible than it has ever been in the past. In light of this new technology, we can now explore the predictive power of social data.

In this paper, we present a method to predict target variables that describe individuals based upon their relationships to others and knowledge of the target variables for some of those other individuals.

In the next section, we do a survey of the previous research on using social graphs as predictive instruments. We then discuss in Section 3 how we apply machine learning to this problem by constructing an appropriate support vector machine. In Section 4, we describe how we obtained test data from the Facebook API[4], as well as the features and target variables that we used. In Section 5, we analyze how our algorithm performs when trained with the Facebook data using ten-fold cross validation.

2 Prior Work

We found relatively little prior research on how social connections can be used to predict tastes. Most prior work on social networks deals with finding cliques and quasi-cliques within social graphs, which is an interesting but different problem. However, some of these studies investigated the use of kernels for analyzing social graphs, which is relevant to our research. In one paper [8], researchers investigated the use of kernel-based distance for clustering a medieval peasant society and found that it did provide coherent clustering. They used the diffusion kernel, which is the discrete solution of the heat equation. We will describe the diffusion kernel further in Section 3.

We found one significant study on social influence, albeit outside the realm of internet social networks. In this study [2], researchers examined the spread of obesity through social connections. Researchers found there is indeed a significant correlation between social proximity and the spread of obesity, and that the effect becomes insignificant after three-degrees of separation. The obesity researchers used logistic regression to predict the obesity of one subject as a function of several variables, including the obesity of another subject. Notably, they measured the strength of the social influence factor by running their regression both on the true social graph and another, fabricated graph with the same topology and overall incidence of obesity, but the incidences randomized. If there were no social factor, they posited, the results of both runs should be the same. This study also found that only same-sex friendships predicted obesity, and that if the friendship was not mutual then it only predicted obesity for the participant who recognizes the friendship.

In another study [1], researchers investigated whether or not tagging in Flickr displayed signs of social influence. They looked at whether or not there is a correlation between tags from one user to the application of tags from another user. Although they found there is indeed such a correlation, they did not study the predictive power of social connections. They used the same shuffle-based test as the obesity study to test how their model compared to a random social graph. Since this paper dealt with a directed graph, they also compared the performance of their model against its performances on the test data with graph edges reversed.

3 Preliminaries

The friend matrix F, given by (1) is an adjacency matrix for an undirected graph of friendships.

$$F_{ij} = \begin{cases} 1 & \text{if users } i \text{ and } j \text{ are friends} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Since we want to make predictions solely based on understanding of the friend matrix, we cannot use algorithms that require explicit representation of features for each training example; we need to use an algorithm that requires only a notion of comparison between two examples which are, in this case, nodes in our friendship graph. Thus we decided to use a support vector machine (SVM) method since these tend to perform well for high-feature models and allow us to use the kernel trick.

The difficulty lies in choosing a kernel function that provides a notion of relevance or similarity between users derived solely from the (complete) friend matrix. We need to choose a kernel function that appropriately exploits the friend relationships. Since F is not guaranteed to be positive semi-definite, we must find a function of F that forms a valid kernel, and has the properties we desire.

The first kernel we tried was a very simple one. As suggested in [6], we calculated the square of the friend matrix (the "Square Transformation"). This forces the matrix to be positive semi-definite; and the entries happen to correspond to the number of links between users of length 2 (i.e. in terms of "friends of friends").

The second kernel we tried was the diffusion kernel, which was used with some success in [8] and described with greater detail in [5]. The diffusion kernel involves defining the Laplacian of the friend graph as shown in (2), and taking its matrix exponential to define a kernel as described in (3). Because it can be expensive to compute the matrix exponential of a large matrix, we optimized this by diagonalizing the matrix by computing the eigenvalues and eigenvectors (as shown in (4)) and using this representation to calculate the kernel for different values of β much more quickly, as shown in (5).

$$L_{ij} = \begin{cases} -F_{ij} & \text{if} \quad i \neq j \\ \sum_k F_{ik} & \text{if} \quad i = j \end{cases} \tag{2}$$

$$K = e^{-\beta L},\ \beta \in \mathbb{R} \tag{3}$$

$$L = U \Lambda U^{-1} \tag{4}$$

$$K = U e^{-\beta \Lambda} U^{-1} \tag{5}$$

4 Experiments

Because of the numerous privacy restrictions built in to the Facebook API, data collection was not as straightforward as one might expect. We first collected a small list of "groups" from a random set of users whose profiles we could access. We requested a list of users who were members of each of these groups and also affiliated with the San Francisco network. We then requested data for each of the possible $\frac{n \cdot (n-1)}{2}$ friend relationships, excluding users for whom we could not determine friend relationships (due to privacy constraints, or because they had no friends among the users queried). Lastly we fetched the accessible profile information for each user.

The Facebook API only allows you to check whether two specific users are friends; there is no way to request the friend list of an arbitrary user. This means that we need to independently "discover" both users before determining if they are friends. On the one hand, this seems like a disadvantage because it means we'll have incomplete friend relationships with which to train our model. On the other hand, it also means our training data will be more representative of the entire social graph, rather than being an arbitrary subgraph.

Using the group-based technique, we collected a data set of 8373 users. Unfortunately, this proved to be too much data to process in a timely manner, so we deleted users that had less than 27 friends from our set in an attempt to make the eigenvector calculation more tractable. This left us will 1103 users. We will call this data set the "trim" set.

We collected another set of data by doing a breadth-first search of friends starting at one of the authors, using the web interface. We stopped after collecting data for 1000 users. One consequence of this technique is that any two users in this data set are separated by a very short number of links, so it provides an interesting contrast to the other data set. We will refer to this data set as the "1k" set.

Figure 1 shows the number of friends per user in each data set.

We should note that neither of these techniques provide a particularly random sample of users because they are heavily influenced by the "seed" groups (in the first set) or by the starting user (in the second set).

As target variables, we used whether users (1) are interested in "music", (2) are single, and (3) were born after 1981. The distribution of positive and negative examples for each variable is shown in Tables 1 and 2.

We used 10 fold cross validation to train and test our support vector machines for each data set, and

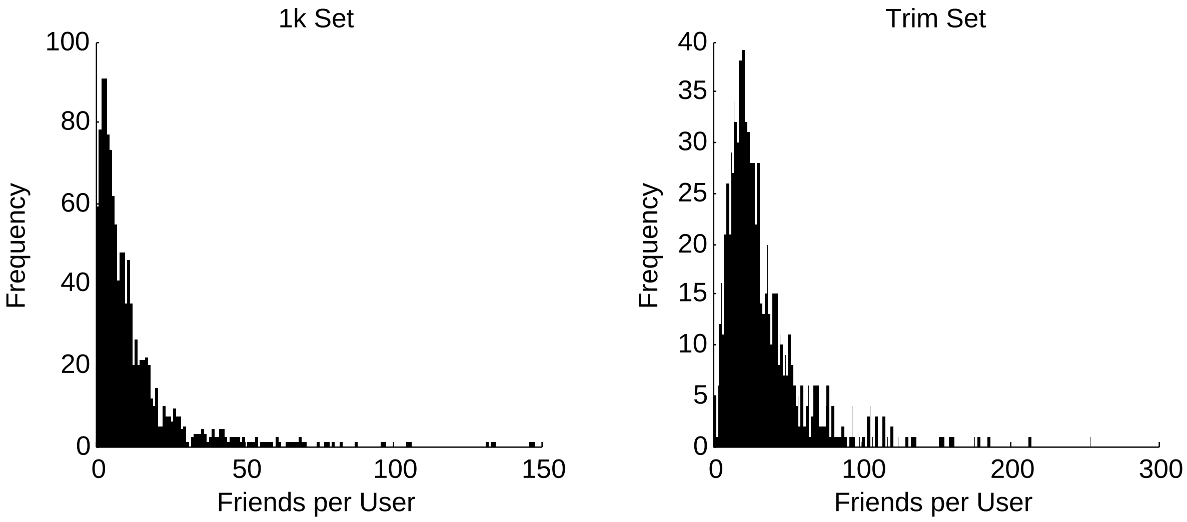

Figure 1: Histogram of Friends per User

	music	single	> 1981
$y^{(i)} = 1$	58	128	152
$y^{(i)} = -1$	230	237	201
Total	282	365	353

Table 1: Class distribution for 1k set

	music	single	> 1981
$y^{(i)} = 1$	73	143	132
$y^{(i)} = -1$	250	244	174
Total	323	387	306

Table 2: Class distribution for trim set

each class, with both the square kernel, and the diffusion kernel for numerous values of β. We used a simplified version of the SMO algorithm described by Platt[7].

Lastly, as a sanity check, we performed the same experiments using the same friend matrix, but assigning the class labels to random users in the graph.

5 Results

Plots of our results are shown in Figure 2. There is one graph for each of the three classes across both data sets. For each set of data and each class, we used 10 fold cross validation to train and test our support vector machine. The dashed lines on the graph show the training error, while the solid lines show the test error. Results with both kernels are shown on the same graph. Since the square kernel is parameterless, we represented it with a horizontal line. Since the diffusion kernel requires the parameter β, we plotted the training and test error for values of β between 0 and 2.

Lastly, with a dot-dashed line, we plotted the test error that would be obtained using a trivial policy, which is to predict that all users are in the more common class.

There are several notable features of these graphs. First, our test error (for all kernels) for predicting whether a user is interested in music or (for the "1k" set) whether a user is single is worse than that found by the trivial algorithm, even though test error is low for low values of β. Furthermore, when we ran the algorithms against the randomly shuffled data set, we saw the same low training error and high test error (higher than the trivial policy error) for all data sets. This confirms that there is minimal or no correlation between whether a user is interested in music (or is single) based on whether his or her friends are interested in music.

For predicting age, we found that with very low values of β the training error was extremely low, while the test error was quite high; this suggests that the system has high variance. As β approaches 0, the diffusion kernel weights the first-order relationships much more than second and third level effects, and these local effects make the algorithm much more prone to overfitting. As β increases, we see that the training error and test error converge somewhat. This

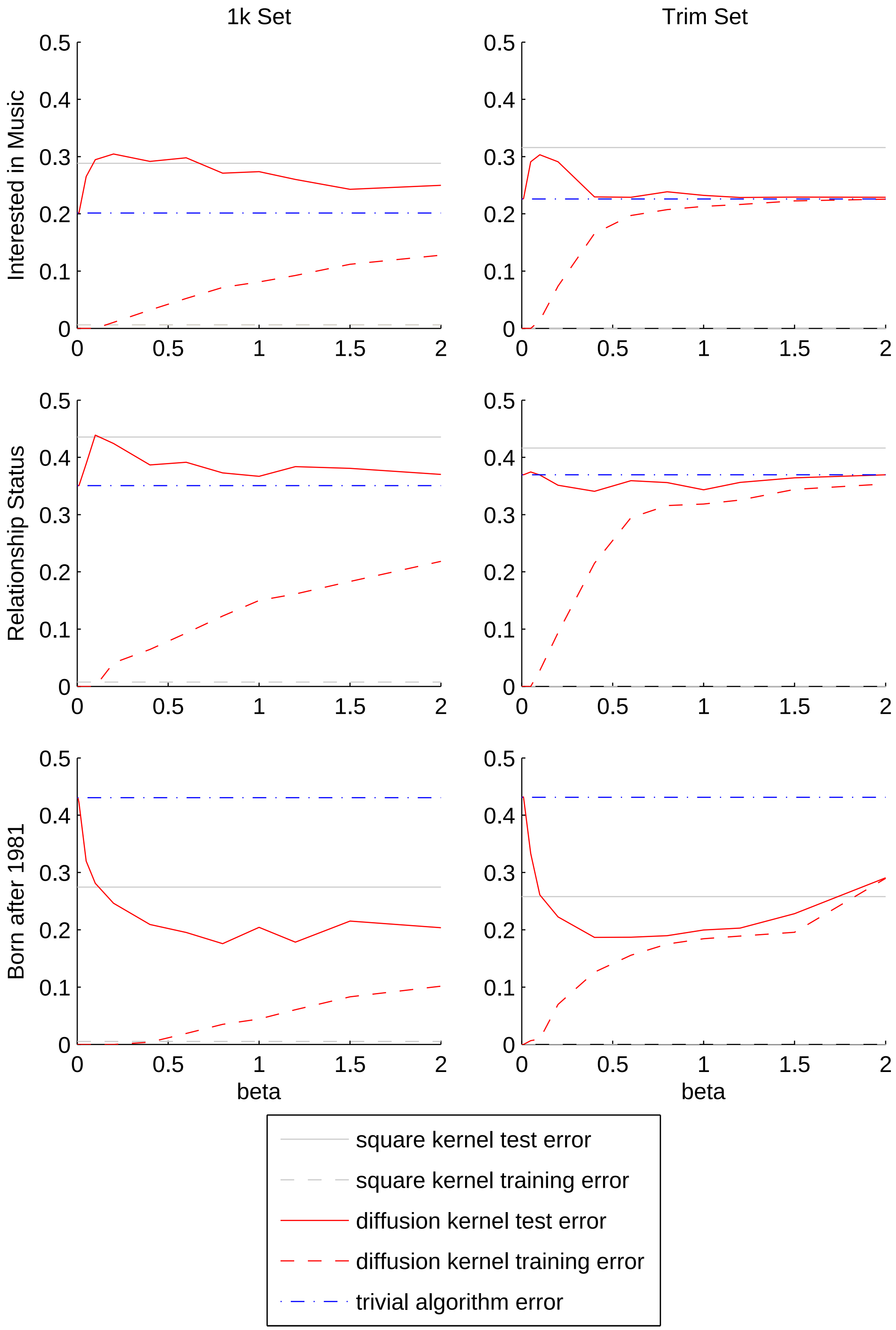

Figure 2: Plots of training and test errors

suggests that at higher values of β, the kernel generalizes better, since it avoids depending on these local effects. The square kernel was prone to the same sort of overfitting as the diffusion kernel for small values of β.

Of the three classes we tried to predict, we were best able to predict whether a user was born after 1981. This suggests that it is, of the three classes, the one that is most likely to be in common between friends.

The two data sets show somewhat different relationships between test and training error. In the "trim" data set, training and test errors converge at much lower values of β. This might be because the friend matrix is much more dense (on average) than that of the "1k" set.

6 Conclusion

Taken together these results suggest that the system has high variance and that we might do better with more data, allowing us to achieve better generalization results for the diffusion kernel at lower values of β. This makes sense intuitively, as human tastes and preferences naturally have high variance and so many complicated dynamics are involved in friend relationships. The basic premise taken in this research is that people with similar interests and characteristics are more likely to be friends with each other. That is obviously true to some extent, but there are also many external factors that add noise to this and make it difficult to predict on such a small scale.

It would be very interesting to repeat this analysis with much more data, but this would require more computing power than we have at our disposal.

7 References

[1] Anagnostopoulos, A., Kumar, R., and Mahdian, M. Influence and correlation in social networks. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (New York, NY, USA, 2008), ACM, pp. 7–15.

[2] Christakis, N. A., and Fowler, J. H. The spread of obesity in a large social network over 32 years. *The New England Journal of Medicine 357*, 4 (2007), 370–379.

[3] Du, N., Wu, B., Pei, X., Wang, B., and Xu, L. Community detection in large-scale social networks. In *WebKDD/SNA-KDD '07: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis* (New York, NY, USA, 2007), ACM, pp. 16–25.

[4] Facebook. Facebook developers API http://developers.facebook.com, 2008.

[5] Kondor, R. I., and Lafferty, J. Diffusion kernels on graphs and other discrete structures. In *In Proceedings of the ICML* (2002), pp. 315–322.

[6] Muoz, A., and n de Diego, I. M. From indefinite to positive semi-definite matrices. *Lecture Notes in Computer Science 4109* (2006), 764–772.

[7] Platt, J. C. Sequential minimal optimization: A fast algorithm for training support vector machines. Tech. rep., Advances in Kernel Methods - Support Vector Learning, 1998.

[8] Villa, N., and Boulet, R. Clustering a medieval social network by SOM using a kernel based distance measure. In *ESANN '07: Proceeding of the 15th European Symposium on Artificial Neural Networks* (2007), pp. 31–36.