CS221 Project Progress Report

Shijie Liu, You Zhou

Nov 14, 2014

**1. Problem Description**

Our project is inspired by Lianpu, a Chinese dating recommendation application which was recently published by Tencent. (You can download the app at <http://101.226.30.208/>.) Users, especially those who are looking for girlfriends or boyfriends, can register for the app by filling in their personal information, photo, character tags, blogs etc. and the app will recommend several other users who the user might like. For example, a male user will be recommended with several female users. After viewing the information of the recommended users, the male user will tag each female user either ‘like’ or ‘dislike’. If user A tags user B as ‘like’, user B will be notified and choose to take another step if they want to develop some kind of relationship. We have access to the user database with their personal information and the data of the actions they performed, namely user A tagged user B as either ‘like’ or ‘dislike’ from Tencent. Ultimately our goal is to design an algorithm to recommend other users for a particular user such that potentially the particular user will like all the users recommended. The first part which we are focusing on right now is to model this as a machine learning problem, and predict the following: given a user A, predict whether he/she will tag user B as like (or dislike). After that the second part will be to design an efficient algorithm to recommend other users which the particular user will tag as ‘like’ with the highest probability under some constraints (e.g. we don’t want to recommend a user to all the other users, so there is a maximum number that the user can be recommended to others). Our project progress will mainly focus on the first part.

**2. Model**

We will use logistic regression to predict whether user A will ‘like’ user B, given their information, which is a binary classification problem We have collected the raw data1 in the format of

‘user A’s id’, ‘user B’s id’, ‘like or dislike’, ‘operation source’, ‘operation time’.

For each pair of user A and user B, we constructed the feature vector (x(i)) (For simplicity we say the feature vector is x instead of φ(x)) by selecting relevant values of both the information of A and B, by browsing the profile data2. Currently our feature vector has 448 rows, which concatenates the information of user A followed by the information of user B, where each is in the format of:

nickname length, in terms of Chinese characters(1 row)

a feature template of the form born\_in\_{month} (12 rows)

height(1)

a feature template of the form

indicator\_{province}(34 rows)

a feature template of the form

indicator\_{city}(168 rows)

minLikeAge(1 row)

maxLikeAge(1 row)

hasHobbyTag(1 row)

numberOfHobbyTag(1 row)

a feature template of the form

indicator\_{bloodtype}(4 rows).

The feature data matrix (X)3 thus have 392 rows, with each row represent 1 dataset (a pair of user A and B), and 448 columns. Node that for each feature (column), we divided each entry by the maximum value in the column in order to normalize the values to a scale from 0 to 1.

And our target data vector (Y)4 are just whether user A tagged user B as ‘like’(which has value 1) or ‘dislike’(which has value -1). It is thus a 392\*1 vector. Note in logistic regression, it is better to represent ‘dislike’ as 0, but in our file we used -1 to represent dislike. We use 70% of the data as training set and 30% as test set.

**3. Implementation**

**3.1 Norm Baseline5**

Firstly we implemented a naïve Norm Baseline, which involves little learning. We made the assumption that if user A and user B are very much alike, meaning their individual feature vectors are close to each other (if the norm of their difference vector is small), then user A will like user B. Hence for x(i) in the dataset, we calculated the 2-norm of the difference vector of its first half with second half, and if is greater than a threshold then we will predict y as ‘1’ and otherwise ‘-1’. As the dimension of feature vector x is very large, we will illustrate with a contrived vector.

For example, , and threshold=0.5, then we will calculate , and thus our prediction will be 1. If the *y* associated with it is 1 then we have made a correct prediction and otherwise our prediction is wrong. We used the training dataset to get the best threshold, which predicts the training dataset with the highest accuracy, and tested on the testing dataset. The results are in the next section.

**3.2 Oracle**

Our oracle implementation involves manual inspection on the profile of the two users, and predicts whether user A will like user B by hand. This is pretty straightforward. We divided the tasks among ourselves on the test dataset, and get an overall accuracy of 74.3%. Note the features used in this oracle implementation involves features like profile picture, specific hobby tags etc., which haven’t been captured in our models’ feature selection.

**3.3 Logistic Regression6**

Then we implemented logistic regression, by performing stochastic gradient descent. Our hypothesis is that we can predict , by discovering the weight vector w from the training set. Recall from lecture that the objective function of logistic regression is , and in order to find the optimal weight vector *w*, we can perform stochastic gradient descent by conducting the update rule

Where is the step size, which we set as , and is our hypothesis, .

Thus in the training phase, our algorithm’s pseudo code can be written as

Repeat until convergence {

for *i*=1 to *m* {

}

}

where *m* is the number of data in our training dataset.

And in our testing phase, we can test our correctness as follows:

*correct*=0

for *i*=1 to *n* {

if ()

}

where *n* is the number of data in our testing dataset.

We are not going to present a concrete example as the dimension of feature vector is large. The pseudo code should be clear and our final result is presented in the next section. You may want to visit our links in the reference to get the data files and our implementation.

3.4 SVM7

We also experimented our SVM, by performing SVM using Matlab’s LIBLINEAR SVM library (<http://www.csie.ntu.edu.tw/~cjlin/liblinear/>).

The implementation is easy and we used its result to compare with our logistic regression implementation, which is presented below.

4. Results and Observation

We used 70% of the data in training and 30% of the data in testing, and get the following training accuracy, test accuracy and number of iterations:

|  |  |  |  |
| --- | --- | --- | --- |
| Model | TrainAccu | TestAccu | #iter |
| NBaseLine | 65.33% | 56.78% | N.A |
| LogRegress | 93.43% | 66.10% | 610 |
| SVM | 93.43% | 61.02% | 78 |

We observed the following:

1. Logistic Regression and SVM performs better than Norm Baseline as expected.
2. Logistic Regression and SVM has similar training accuracy, but Logistic Regression performs better in terms of testing accuracy.
3. While Norm Baseline generalizes (has similar training and testing accuracy), the training accuracy of Logistic Regression and SVM is much higher than testing accuracy, this is a sign of overfitting.

**5. Future Improvement**

1. As pointed out earlier, our logistic regression suffers from overfitting, as in the training accuracy is much higher than the testing accuracy. We will conduct a number of techniques to reduce overfitting, like regularization and removing irrelevant or useless features. For example, after running the logistic regression we find that the weight associated with the feature indicator\_40500, which is an indicator of whether user A is from city id 40500, is 0. This may be caused by the fact that not a lot of people come from that city and thus there are not a lot of non-zero entries in that column. Thus we may want to remove that feature or replace it with more useful ones.

2. Our feature is far from complete. In our oracle implementation, we viewed the complete profile of the users including photos and specific hobby tags, which are very crucial for a user to make ‘like’ decisions. We experimented to only look at the features we had selected and make predictions, the result showed that only around 60% of the time we can get it right. This shows that our features haven’t captured some important information in the user profile, which may lead us towards the oracle accuracy numbers. Thus our next step will be more feature selection, involving procedures like image processing on the profile photo and word processing on hobby tags.

3. So far we have only explored 1-layer linear classifiers. We may want to use more complicated classifiers like neural networks to improve our performance by discovering the hidden interaction between features by going through more layers.

4. After we have a reasonably good logistic regression result and a mechanics to measure the probability whether user A will like user B based on it (for example, we can look at the score in the logistic regression, the higher score means higher probability of like), we will design a way to select several users from the user data pool who the user will like with high probability. As the user data pool is large, we may have to do some optimization and pruning. For example, we can use K-means to cluster the users into several clusters and only look for profiles within one cluster to recommend. Moreover in order to add the rule that each user cannot be recommended more than a fixed number of times, we may model this as a constraint satisfaction problem. We will start this part of the project after our current logistic regression task.

**6. Reference**

1.<https://github.com/OdetteDu/LianPu/blob/master/Data/data.txt>

2.We are not allowed to disclose profile data

3.<https://github.com/OdetteDu/LianPu/blob/master/Data/x.dat>

4.https://github.com/OdetteDu/LianPu/blob/master/Data/y.dat

5.https://github.com/OdetteDu/LianPu/blob/master/Milestone/NormBaseline.m

6.https://github.com/OdetteDu/LianPu/blob/master/Milestone/LogisticR.m

7.https://github.com/OdetteDu/LianPu/blob/master/Milestone/svm.m