Predicting Conversational Likability On Anonymous Chat Networks

Arpad Kovacs, Aditya Somani, Daniel Velkov

Abstract

We implemented a learning system that is able to predict if two users on the Chatous anonymous chat platform are likely to have a meaningful conversation. This has the practical application that by examining the past conversations and activity of existing users, we can better match together new conversations and generate more personalized suggestions of which other members a user would be interested in connecting with.

1 Introduction

Chatous is a 1-on-1 anonymous online chat network which matches users from around the world with one another in text-based conversations. However, the anonymous nature of the Chatous platform can easily be exploited by users who engage in antisocial behavior (eg: taunting or threatening the other party) which results in an undesirable user experience. Another issue is that some users might start a chat, only to end it immediately upon determining that they are not interested in the other member.

User acquisition in a social network is a costly investment. A new user is much more likely to be retained if her initial conversations turn out to be highly satisfying. Our project is focused on using a user's profile information and past conversations to predict which users tend to be good conversationalists. We use conversation length as a proxy for the quality of conversation, based on the observation that a long chat which holds the user's attention is more likely to be meaningful and engaging than a short or zero length conversation.

2 Model

In this section we will describe the different approaches and sets of features that we use to model the conversation likability problem. For a given pair of users, we want to predict the length of their conversation and whether they will establish a friendship connection. We do this by looking at pairs which have actually engaged in a conversation and trying to predict the outcome by only looking at the information we have at the time when the conversation starts.

To establish a baseline, we started with the following user and profile features: profile information for each user, the absolute-value of their age differences, the xor of their declared sex (0 if identical, 1 if not), and whether one of them specified their location. Profile information for each user consists of: a boolean indicating if the user specified a location, the user's age and sex and the length of their "about me" description.

We trained logistic regression over 1 million chat records with a 66% train, 34% test split. Since the percentage of long conversations in the original dataset was too low, we boosted it to ensure a 50-50 split between examples of each category. Initially, we utilized the labels Short (S), Long(L) and Finished(F) provided by Chatous as our output class. We soon determined that many of the training examples were labelled incorrectly with regard to the problem we were trying to solve, e.g. a chat might be labelled as Finished even if the users engaged in a long conversation. Similarly, we found cases in which a chat with little content was marked as Long. Therefore we switched to a binary class output which indicates whether the chat was long or not based on whether the conversation contains at least 20 words.

We also looked at predicting if the conversation will be reported by one of the users due to several predefined reasons: spam, harassment, filth. We evaluated several algorithms and different sets of features for this task. Our conjecture is that knowing the probability that a conversation will be reported is useful signal for the general conversation likability task. We performed analysis on the frequency of reports and the possible reasons for reporting a conversation.

3 Results

3.1 Predicting conversation likeability

Our classifier was able to attain 81.9% accuracy using a dataset of 1 million chat records. The following table shows the results we obtained:

Class	TP Rate	FP Rate	Precision	Recall	F-Measure
Short conversations	0.726	0.088	0.893	0.726	0.801
Long conversations	0.912	0.274	0.768	0.912	0.834

In order to determine which features are most relevant to predicting conversation length, we utilized the J48 tree implementation of C4.5 from the Weka machine learning package [1]. The C4.5 algorithm [2] builds a decision tree from training data consisting of already-classified examples, by recursively splitting branches based on the attribute which most effectively distinguishes between the available classes. We determined the most relevant decision features (in order of importance) to be whether users are of opposite sexes, whether the difference between the users' ages is low, and whether both users specified their locations in their profiles, and if yes then whether they are from the same country.

3.2 Predicting user quality

We performed preliminary analysis of the Chatous data by sampling 300,000 conversations out of the 9 million available. This sample was joined with the other provided dataset which has labels for a set of users. The labels are: 'clean', 'dirty' and 'bot'. We chose to look at the problem as a binary classification task, predicting if the user is 'clean' or not. After joining the sample conversations with the user quality dataset, we had 25,140 conversations with a quality label. The number of 'clean' and not 'clean' labels was almost evenly split: 13859 clean, 11281 not clean.

As a baseline we looked at the number of lines that each user has sent per conversation. The breakdown depending on the user quality is as follows:

quality	numlines1	numlines2
not clean	1.482182	1.377127
clean	2.005244	2.015034

This suggests that we can use conversation lengths as a baseline for modeling the user quality problem. Using the number of lines that each of the users wrote as our features and running a 10 fold cross validation gives an F1 score of 0.69 for the user quality task. We found that when optimizing other scores like accuracy, precision or recall it was easy to get high values but it didn't seem like those results would be applicable in a real-world scenario.

Another feature that we considered is the word vectors for each user in a conversation. The raw counts were taken without applying any transformations. For optimizing the cost function, we used a stochastic gradient descent algorithm from the scikit-learn machine learning package for Python [3]. We explored different parameter values for the SGDClassifier: regularization terms, penalties (l1, l2, elastic net), loss functions (hinge, logistic, perceptron), number of iterations. We choose the optimal parameters by running a grid search over the parameter space and performing a 10-fold cross validation for each combination. Using those optimal values, we obtained a F1 score of 0.73 using the word vectors as a feature matrix.

We implemented a "user proximity" feature by extracting the countryname from each user profile's declared location, and checking whether the country names of two users engaged in a chat matched exactly. This feature yielded an F1 score improvement of approximately 0.01.

Surprisingly the best performance was achieved by just using the word vectors from the 'about' field of the user profile and the country extracted from the user's location. The best cross validated F1 score was 0.91, at 1,000 iterations of SGD. This might be due to the fact that the about fields for some problematic users are actually the same due to creating multiple accounts.

Other features which were considered but did not result in higher performance are the ages and genders of the two users, and boolean values indicating if they are the same gender and age group. Other features that might be worth exploring are the geographic distance between users, as well as the time elapsed between when a user created their account and a particular chat occurred, however we predict these will probably only result in small marginal performance improvements over our already-existing features.

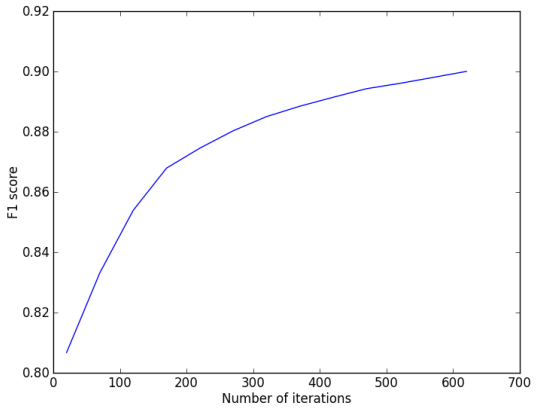

Figure 1: Model performance as a function of SGD iterations

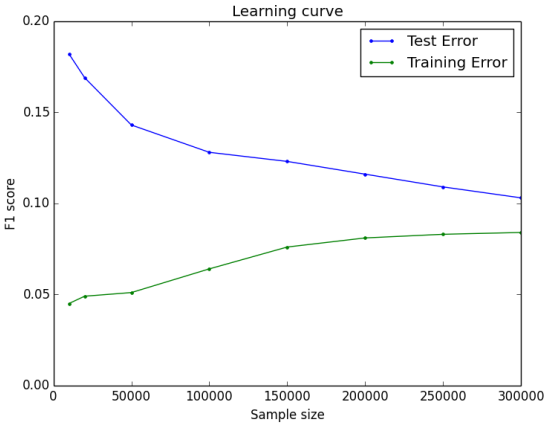

Figure 2: Error as function of sample size

We experimented with running the SGD algorithm for increasing numbers of iterations. Since we used regularization, increasing the number of iterations led to improvements in the F1 score. Those improvements started diminishing above 500 iterations and that's why we didn't continue experimenting with much higher values. The results are summarized in Figure 1.

3.3 Taking past behaviour of users into account

We added features modeling user's past behaviour. The premise was that low quality users are likely to exhibit previous antisocial and disruptive behaviour, while high quality users may have a history of congenial, friendly conversations. For each user, we created the following features:

1. enough_cov: an indicator function indicating if we have enough conversations (10) from a user to make a meaningful prediction from history
2. perc_ended_self: percentage of conversations which were ended by the user herself
3. perc_non_zero_ended_other: percentage of non zero length conversations which were ended by other users
4. perc_long: percentage of long conversations that the user has had
5. perc_zero_ended_self : zero length conversations ended by the user herself

Now it is logical to assume that the outcome of a conversation will be dependent upon the user with the lower quality amoung the participants. So for each conversation, we generate the following features based on the features of the two partcipating users (u1, u2):

1. u1.enough_cov && u2.enough_cov
2. min(u1.perc_ended_self, u2.perc_ended_self)
3. max(u1.perc_non_zero_ended_other, u2.perc_non_zero_ended_other)
4. min(u1.perc_long, u2.perc_long)
5. max(u1.perc_zero_ended_self, u2.perc_zero_ended_self)

We tried various classifiers (Logistic, Neural networks, J48 trees) using these features in conjunction with existing features. Surprisingly, these features didn't result in any improvement in performance.

4 Error Analysis

After executing the logistic regression classifier to predict conversation length, we used weka's "visualize classifier errors" feature to determine which classes of examples are being misclassified, and how correlated various features are to the predicted and actual labels. The charts below can be interpreted as follows:

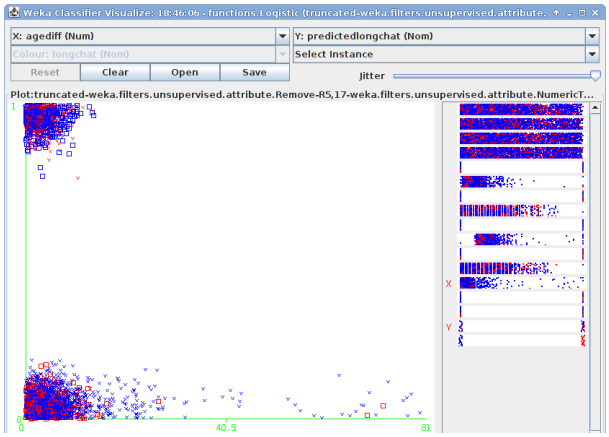

Figure 3: Age difference between users (x-axis) vs predicted conversation length (y-axis).

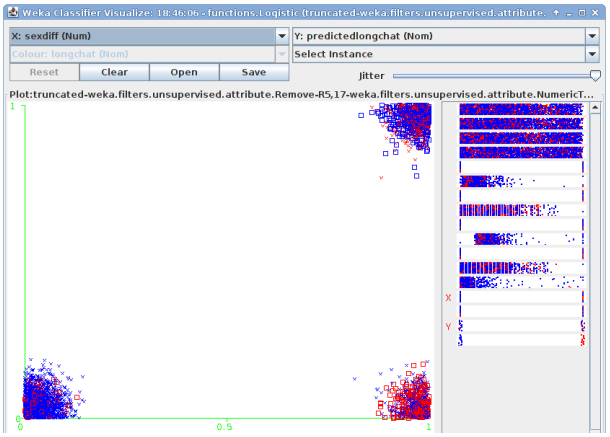

Figure 4: Sex difference between users (x-axis) vs predicted conversation length (y-axis).

- $\times$: correct prediction (predicted matches actual longchat label)
- $\square$: incorrect prediction (predicted does not match actual longchat label)
- Blue: actual longchat label=0 (this was a short conversation)
- Red: actual longchat label=1 (this was a long conversation)

Examples for which the feature matches the predictedlongchat label are shown in the bottom-left (both predicted and actual labels are 0) and top-right (both predicted and actual labels are 1) of each figure.

In the age difference plot, most of the examples are being clustered on the left (eg: the absolute-value difference in ages between most users is small). The classifier does not lend much weight to the age difference feature (the classifier's predictions for examples with a given age are almost evenly split between 0 and 1). The error rate for this feature is also high; for instance, we observed that examples in the top-left (where age difference is low and the classifier predicted longchat=1) have lots of blue $\square$ indicating actual longchat=0 label does not match the prediction, while there are lots of red $\square$ examples in the bottom-left. However, notice that the vast majority of examples with age difference more than 17 appear to have short conversations (as indicated by dominance of blue markers on the right). This suggests that turning age difference into a quantized or binary feature (agediff$>$ 17) might improve its predictive power.

In contrast, it seems like the sex difference feature (0 if users participating in a conversation have the same sex, 1 if users are of opposite sex) is a better predictor of the length of the chat. Notice the abundance of blue $\times$'s in the lower-left corner; this indicates that when two users have the same sex, the classifier correctly predicts that their conversation will be short. The classifier has more difficulty predicting when opposite-sex partners will have a long conversation, as evidenced by the large number of false positive blue $\square$ (classifier predicted long conversation, but actual conversation was short) in the top-right and false negative red $\square$ (classifier predicted short conversation, but actual conversation was long) examples in the bottom-right. To make conversation-length prediction of opposite-sex users more effective, we may add features that capture sentiment as a proxy for the chemistry between users, eg: positive sentiment may be linked to longer "romantic" conversations, while negative sentiment may be associated with offensiveness and aggression. Another hypothesis is that some "troll" users lie when declaring their sex, which we will be able to confirm or refute upon combining the conversation length model together with the abusive/flagged users model.

5 Diagnostics

In order to determine the sources of error in our SGD algorithm and how we could improve it, we implemented several of the recommendations from the "advice for applying machine learning" lecture.

First, we plotted the learning curves of the training error and the test error as seen in figure 2.

As the size of the training set increases, we observe that the training error continues to decrease, while the training error increases. The gap between the training error and the test error, as well as the shape of the train and test curves indicate that we are experiencing high variance. In response, we pruned the number of features we use down to the users' about-descriptions and countries, which yielded the highest F1 score of 0.91 and confirmed our hypothesis.

Since we are in a high-variance regime, a larger training set should also improve our classifier's performance, but unfortunately we began to run into out-of-memory errors at around 300,000 chats. We believe that we could achieve better results with improved hardware that could fit more data into memory. Alternatively, we could implement a streaming solution that loads subsets of the data into memory, batch-processes each subset, and then combine the results, however this would be significantly slower due to disk-access becoming the bottleneck.

We also plotted the F1 performance as a function of the number of iterations of stochastic gradient descent. As you can see from figure 1, there are improvements when increasing the number of SGD iterations, but we encounter diminishing marginal returns. We elected not to try running more than 1,000 iterations of SGD in due to computation time limits.

6 Future Work

We developed two separate models: one for predicting conversation length and the other for predicting user quality. There could be a potential benefit from combining those two models, for example knowing the predicted label for user quality could be useful when predicting the conversation length and vice versa. This could be achieved by stacking the predictions of the two models and then training a simple regression model over the resulting two column matrix.

The largest sample size we processed was 10% of the entire Chatous dataset due to computational constraints (mostly memory limitations). That sample size was the upper limit for tuning our models since we are running a grid search over a large parameter space which requires several hundred cross validation runs. Given enough computaing resources it would be interesting to see how will the model performance change when the complete dataset is taken in consideration. Also more computational resources would allow us to explore a larger parameter space which will likely further improve our model's performance.

We also observed from the dataset that some low quality users often create multiple profiles with contradictory information about their age, sex and location. We believe that such features would be very useful in filtering out low quality users.

One area of possible further improvements is feature engineering. The country-based proximity feature is very coarse, and could be improved by computing geographical distances between the actual users' locations. Another possibility is to refine the user profile features, for example by computing the time between when a user created an account and a particular chat as a proxy for the user's experience level on chatous (although to accurately measure engagement, we would need login times which we did not have access to). Finally, since we found that since the sex difference is the greatest predictor of whether users will have a long chat, it would be worthwhile to perform semantic or tone analysis on the conversation to determine whether users' actual sex and interests match those which are declared in their profile (however, we were unable to do this since we did not have access to the actual conversation text).

References

[1] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *SIGKDD Explorations*, vol. 11, no. 1, pp. 10–18, 2009.

[2] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.