Hearst Analytics Challenge: Prediciting Email Solicitation Responses

Steven Luyee, Jason Yu

December 15, 2011

1 Introduction

This project is based on the 2011 $25,000 Hearst Analytics Challenge sponsored by Hearst Magazines. The goal of the challenge is to develop a model to predict whether a recipient of an email solicitation with new magazine offers will a) open the email and b) click on the link provided in the email. Provided by the challenge are records to 1,785,421 email solicitations from 2010 to up to early February 2011. Each record contains solicitation history and outcome, including 273 variables detailing demographics information about the individual being solicited and the household information for the individual solicited. Our goal with respect to the project is to build a quality supervised learning algorithm to predict the outcome of a particular email solicitation based on the different features of each individual's demographic and household information.

Information provided by Hearst about each solicitation includes the following:

I. Solicitation outcome:

(a) OPEN_FLG: Indicates email was opened.
(b) CLICK_FLG: Indicates email link was clicked.

II. Household related information (for up to 8 other members of the individual's household):

(a) Stacd_hh_mem1-mem8: The code that identifies the status of the specified name in the household (i.e. H= head of household, P = elderly parent, U = other adult, W = spouse, Y = young adult, null = no match).
(b) Age_hh_mem1-mem8: The age of household member.
(c) Gender_hh_mem1-mem8: The gender of the specified household member.

III. Individual related information:

(a) I1_EXACT_AGE: Age of individual
(b) I1_GNDR_CODE: Gender defines the gender of the individual
(c) EXPERIAN_INCOME_CD: Total estimated income for a living unit (discretized)
(d) TRW_INCOME_CD: Total estimated income for a household
(e) HH_INCOME: Census median income for households located within a census tract or block group.
(f) INDIV_MARITAL_STATUS: Marital status
(g) ETHNIC_GROUP_CODE: Ethnicity
(h) ADDR_VER_CD: Whether address is verified
(i) Y_OWNS_HOME: Indicates the likelihood of a consumer owning a home
(j) DWELLING_TYPE: A dwelling type code
(k) LENGTH_OF_RESIDENCE: Length of residence. Zero values indicate less than one year.
(l) DWELLING_UNIT_SIZE: Dwelling unit size.
(m) PRESENCE_OF_CHILDREN: Number of children in household information
(n) INDIV_EDUCATION: Directly reported survey data that indicates education level for the individual.
(o) OCCUP_DETAIL: Occupation codes
(p) RELIGION: Indicates likely religion of the individual
(q) COUNTRY_OF_ORIGIN: Indicates country of origin
(r) LANGUAGE: Indicates language prefer-

ence

IV. Trait Information (Up to 64 traits):

(a) Music Preference
(b) Reading Preference
(c) Activities Preference

2 Initial Considerations

Before we can consider which supervised learning algorithm to use as a model for this problem, we need to address a few critical issues with the dataset. For one, we are faced with more than 1.7 million responses with 273 variables each, which is an unnecessarily large dataset to train a good model on. Secondly, the number of "positive open examples" (i.e. number of individuals who opened the email) represents only 7.73% of the entire dataset. Likewise, the number of "positive click" examples represents only 1.58% of total responses and 13.55% of all "positive open" examples (i.e. 13.55% of all people who opened the email clicked on the link in the email).

Another important issue is the problem of validating the model. Although the challenge provided a separate validation test set to iteratively check the quality of a particular model and compare results with other teams, we are unable to use the set as the actual challenge ended in August 2011. Although the challenge provided an error metric to use at the main criteria for judging the quality of the model, it is not possible to compare to previous enteries as we do not have information on the mix of positive and negative examples in the evaluation test set and thus cannot accurately compare our results to others.

The error metric is as follows:

$$\sqrt{(\frac{1}{n}\Sigma[(0.34*(acto-predo)+0.66*(acte-prede))^2]} \quad (1)$$

acto = actual open event
predo= predicted open event
actc= actual click event
predc = predicted click event

We also need to address the issues of missing or sparse data. Because the majority of the data is given as categorical data, we may face the problem of seeing levels for a particular feature that are present in the test set, but not the training set. Some classifiers, including SVM, require categorical data to be expanded into multiple binary variables. Some features are highly sparse to start with and unlikely to contribute meaningfully to the overall prediction model. We may also run into issues with highly correlated data features, such as location information (i.e. zip code, city, state, etc.). Other features are simply irrelevant to the overall model, including unique ID numbers for each recipient.

3 Feature Space Reduction

Before implementing a model, we wanted to reduce the feature space as not all features were important or relevant to the implementation of the model. We began by manually eliminating irrelevant or redundant features, including unique ID numbers.

We noticed that a lot of features have many missing entries. Naturally, a feature with the majority of the entries missing should not have much predictive power. Hence, we heuristically decided to remove features with more than 70% of the entries missing. That leaves roughly 100 features remaining.

4 Implementation

We experimented with two primary learning algorithms: Naïve Bayes and SVM. While implementating the Naïve Bayes algorithm is quite straightforward, the SVM implementation requires additional pre-processing due to the presence of categorical data. The SVM algorithm requires categorical variables to be expanded into multiple binary variables. Because many categorical variables have multiple levels, the original 100 features become more than 1000 variables after the expansion.

As a result of this ten-fold growth in data size, we could only realistically train the SVM with linear kernel via liblinear on a personal computer.

5 Initial Results

To investigate the performance of our learning algorithms, we produced the learning curves for both Naïve Bayes and SVM. Since our goal is to classify highly skewed labels, we characterized the learning curve with F-score instead of error rate. 70% of the sample size is used to train the classifier, and the model is then tested on 30% of the sample.

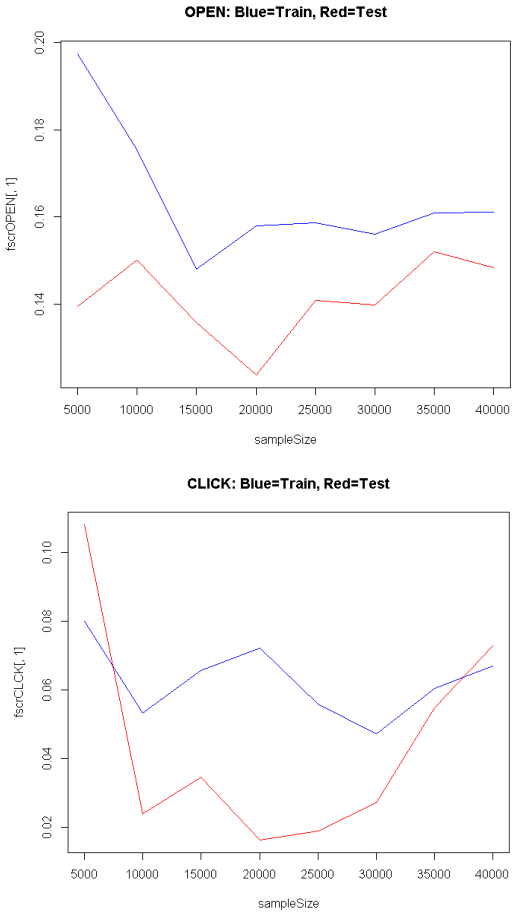

Figure 1: *This figure shows the results of a basic Naive Bayes classifier.*

As shown in Figure 1, the F-score is rather low, especially for the CLICK label prediction. However, the gap between in-sample F-score and out-of-sample F-score for Naïve Bayes classifier is quite narrow even for small sample size, suggesting that Naïve Bayes stabilizes rather quickly and more training samples will not improve the performance.

The poor performance of both in-sample and out-of-sample data suggests that the classifier suffers from high bias. In an attempt to fix the high bias, we tried to train a SVM with linear kernel.

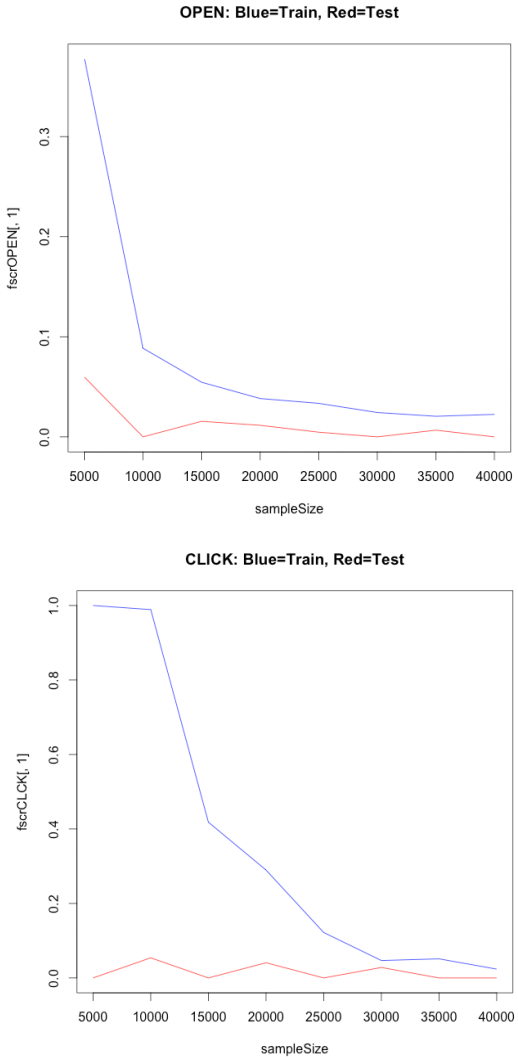

Figure 2: *This figure shows the results of a basic SVM classifier.*

The in-sample F-score starts out high but degrades very rapidly with increasing sample size, while the out-of-sample F-score always remains low. The fact that even in-sample F-score is low suggests the SVM algorithm also suffers from high bias. To improve the bias-variance trade-off, we try to vary the C parameter to see if we can achieve an optimal

compromise between bias and variance.

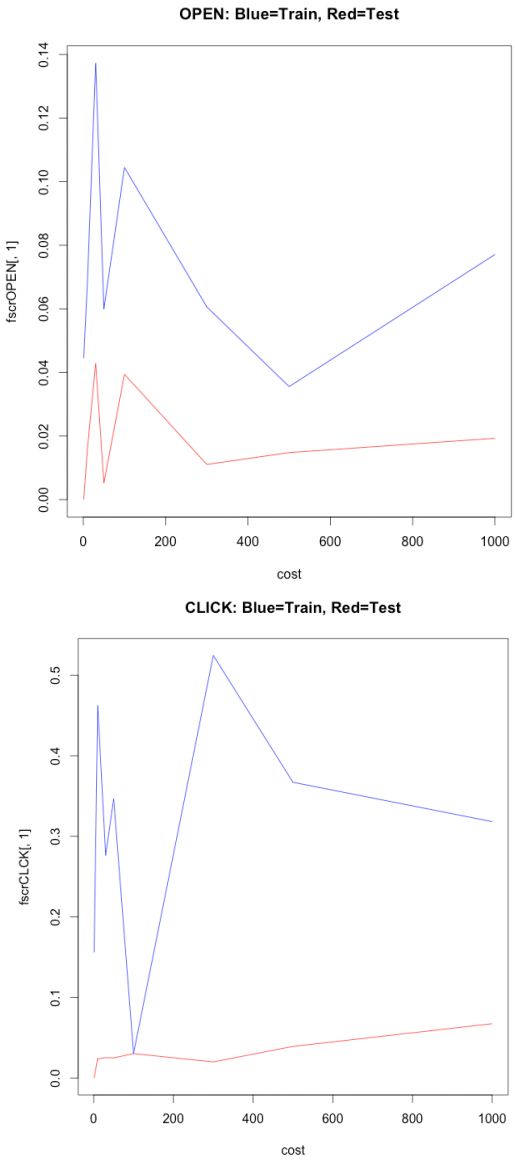

Figure 3: *This figure shows the results of a basic SVM classifier with a cost parameter.*

Interestingly, as observed in Figure 3, the F-score for the OPEN labels is very insensitive to the cost parameter. The F-score for the in-sample CLICK labels has been successfully improved to about 0.4, but the out-of-sample F-score does not catch up. Now the SVM seems of suffer from high variance, and adjusting the cost parameter does not seem effective in finding a compromise in the bias-variance tradeoff.

5.1 Improvement Through Boosting

The highly skewed distribution is the root cause for the difficulty in training algorithms. We adopted advices from the Teaching Staff to implement a boosting algorithm to mitigate the problem of skewed distribution.

There are a variety of boosting algorithms. The particular one we adopted works as follows:

1. Build a training data set with a reasonably balanced distribution (50/50 or 30/70).
2. Repeat {
a) Train the learning algorithm
b) Use model to predict on fresh data with the original unbalanced distribution
c) Construct a new training set by incorporating the mis-classified test data with the training data from previous iteration
}
3. Test the resulting model on out-of-sample data.

Such an algorithm effectively grows the training set in a smart way by combining only the samples that are near the decision boundary. The rational is that samples near the boundary contain more valuable information. For the sake of computational time, we heuristically chose to run the training for six iterations instead of using any convergence criteria. Also we only applied boosting to the CLICK labels because the performance for CLICK is worse than the performance for OPEN.

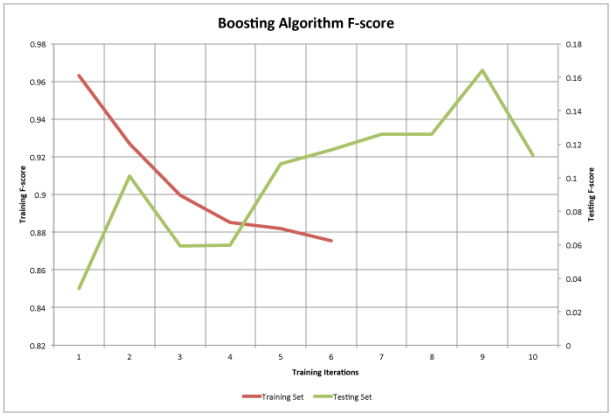

Figure 4: *F-score generated by the boosting algorithm using Naive Bayes.*

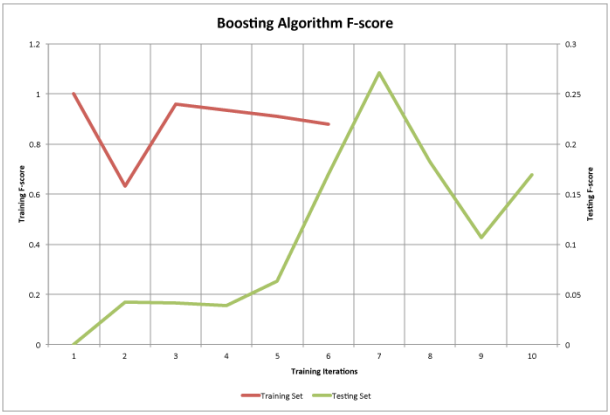

Figure 5: *F-score generated by the boosting algorithm using SVM.*

From Figures 4 and 5, we can clearly see that the out-of-sample F-score rises slowly during the training phase (the first 6 iterations) and stays above 15% for the 4 iterations of out-of-sample testing. Even though an F-score of 15% is still rather low, it is a considerable improvement over the previously single-digit percent F-score. This shows that boosting does add value in classifying highly skewed distribution.

6 Conclusion

In summary, the three most difficult aspects of this Challenge are 1) highly skewed class distribution, 2) large data set with sparse features, 3) feature expansion of categorical variables for SVM.

Direct applications of two types of classifiers, Naïve Bayes, and SVM, all turned out to perform poorly due to the high skewness in the class distribution. After incorporating a simple boosting algorithm, we were able to considerably improve the F-score for both Naïve Bayes and SVM. This improvement suggests that more sophisticated boosting schemes are worth pursuing for further improvement. One such approach to boosting is called Granular Support Vector Machines - Repetitive Understampling algorithm (GSVM-RU) [4], which requires repetedly selecting and removing negative support vectors from the raw data set and iteratively training the model on a new data set made of all positive examples and the aggregated negative support vectors removed from the original dataset.

Another major challenge is feature selection. More than half of the features are categorical variables with multiple levels. When training SVM, more than 80% of the computational time is actually spent on expanding categorical variables. The need for feature expansion makes stepwise feature selection prohibitive. Any dimension reduction technique, analogous to PCA for categorical data, may help to eliminate non-informative features and reduce computational time.

References

[1] F. Provost, G. Weiss, *The Effect of Class Distribution on Classifier Learning: An Empirical Study.* Technical Report ML-TR-44, Department of Computer Science, Rutgers University.

[2] F. Provost *Learning with Imbalanced Data Sets 101.* Invited paper for the AAAI'2000 Workshop on Imbalanced Data Sets.

[3] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar *Introduction to Data Mining.* Addison Wesley, 2006.

[4] Yuchun Tang, Yan-Qing Zhang, Nitesh V. Chawla, Sven Krasser, *SVMs Modeling for Highly Imbalanced Classification.* IEEE Transactions on Systems, Man, and Cybernetics, Part B 39(1): 281-288 2009.