Final Project Writeup **Teammates: Mai Ho, Maury Quijada**
CSCI158
Prof. Jessica Wu
Due Date: December 11, 2015

Controversial Reddit Comments

Introduction

Reddit is a popular social networking site where users submit web content, comment on it, and upvote or downvote comments. One interesting attribute of these comments is controversiality, which is a flag denoting if a comment has been upvoted and downvoted significantly. Our classification task is: given information about a Reddit comment, classify whether or not it is controversial.

Datasets

Our dataset comes from the Reddit Comments Kaggle competition page, where we obtained a database of all 54,504,410 Reddit comments posted in May 2015. From this database, we created a training and testing set, whose metrics are shown in Table .

When we performed our data exploration, we found that only 2.4 percent of comments were flagged as controversial. That means that if we just randomly sampled from the database, the training set would have very few controversial comments. Having an imbalanced training set could cause the classifiers to not generalize as well to other controversial comments. Therefore, to avoid this problem, we created a training set where 50 percent of comments were controversial.

However, we also wanted to see how our classifiers perform under the actual distribution of controversial comments. So for our testing set, we created a dataset with approximately the same distribution of controversial comments as the original dataset, with no overlap with the training set.

Features

The database of Reddit comments had a total of 21 features. Out of this total, we found 5 relevant features, as summarized in Table 2.

Methods

Our approach relied heavily on the Scikit-Learn Python machine learning library [1]. We used scikit-learn to train and test our different classifiers, to perform cross-validation on the test set, and to evaluate the performance of our results.

	Total Dataset	**Training Set**	**Testing Set**
Total Comments	54,504,410	800,000	102,480
% Controversial	2.4 %	50%	2.4%

Table 1: Dataset Information

Feature Name	Description
score	net upvotes
gilded	number of times Reddit gold has been given to the author
edited	boolean of whether comment was edited
subreddit	subreddit that the comment was posted in
body	comment body, represented as a bag-of-words using the top 10 common words in the bodies of controversial comments

Table 2: Feature Summary

Evaluation Metrics

We evaluated the performance of each classifier based on three different metrics on the test set: precision, recall, and F1-score. We do not consider training and testing accuracy here, because we deemed those metrics to not be informative given our imbalanced training set.

First, we compute precision to tell us the percentage of labeled controversial comments that were actually controversial. We included this metric because we wanted to make sure that our model was not simply overgeneralizing and labeling all comments as controversial.

Then, we compute recall to tell us the percentage of controversial comments that were labeled as such. This metric helped us evaluate how good our classifier was at picking out controversial comments out of the ones we gave it.

We finally compute F1-score, which is a function that strikes a good balance between precision and recall. Because we equally value what information that precision and recall gives us, we use this metric as the main indicator of performance.

Baseline Classifier

Our baseline was a Decision Tree Classifier with only one feature: subreddit [1]. We trained the baseline on the training set mentioned above, so that given a comment, the baseline would predict that it is controversial if the comments subreddit has more controversial comments in the training set.

Decision Tree Classifier

To improve on the baseline, we tried an unlimited depth Decision Tree Classifier using all the features mentioned in the dataset section, using entropy to measure the quality of a split [1].

Depth-limited Decision Tree Classifier

The depth-limited Decision Tree Classifier limits the depth of the resulting tree to some maximum depth. We selected the maximum depth using hyperparameter tuning with 5-fold cross-validation on the training set. Once we found the maximum depth, we created a Decision Tree Classifier with that depth, again using entropy to measure the quality of a split [1].

Logistic Regression

Using Scikit-Learns Logistic Regression module, we created a Logistic Regression Classifier using all of our features with a default 0.5 threshold [1]. We also added regularization to the classifier, which we tuned using 5-fold cross-validation.

Adaboost Ensemble with Decision Stumps

The last classifier that we attempted was an Adaboost Ensemble Classifier using decision tree stumps [1]. We limited the number of decision tree stumps to 100 since the dataset that we were training on was large.

Results

Classifier	Training Acc.	Testing Acc.	F1-score	Precision	Recall
Baseline	0.7139	0.6470	0.0966	0.0515	0.7787
Decision Tree	0.8601	0.8066	0.1575	0.0880	0.7523
Depth-limited Decision Tree	0.8263	0.8197	0.1683	0.0946	0.7588
Logistic Regression	0.6569	0.5736	0.0774	0.0408	0.7438
Adaboosting	0.7731	0.8542	0.1867	0.1078	0.6963

Table 3: Classifier Results

Comparing across each metric, we see that the following classifiers performed best.

Training Accuracy	Testing Accuracy	F1-score	Precision	Recall
Decision Tree	Adaboosting	Adaboosting	Adaboosting	Baseline

Table 4: Best performing classifiers

As stated earlier, the metric that we decided to place the most importance on was the F1-score. Looking at Table , Adaboosting with decision stumps had the highest F1-score. However, to determine if our results were significant, we decided to compare them against our baseline classifier, which was a Decision Tree with only the subreddit feature. Compared to the baseline, the adaboosting classifier only increased the F1-score by 0.0901. Since 0.0901 is such a small increase, we found that trying out different classifiers did not give us any significant results.

Conclusions

We found that since the different types of classifiers did not improve on our baseline, the most likely issue is that our extracted features were not informative enough. For example, the bag-of-words feature that we used contained words that, to us, did not immediately show any connection to controversiality. This is due to a number of reasons. First, we did not do enough preprocessing on the comments, so very common words like really and like, as well as HTML character escapes such as `gt` for the greater than sign, showed up in our bag-of-words. This caused the bag-of-words feature to not be as informative as we liked.

For the future, we see two potential features that may help improve the models. First is creating a more sophisticated bag of words by taking the top 50 words in controversial comments (A), the top 50 words in non-controversial comments (B), and setting our bag of words to be the words that are in A but not in B. This would hopefully give us a bag-of-words that is more informative. Another possible feature could be a sentiment score of the comment. Comments with extreme scores could be more likely to be controversial and it would be a good feature to explore.

References

[1] Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Pas- sos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Re- search 12:28252830.