EECS 349 - Machine Learning (Spring 2018)
Project Status Report
Team: Yixue Wang, Hanlin Li, Katya Borgos-Rodriguez

In our project, we used data from The New York Times (NYT) to build a classification model that could predict whether a comment left on an article was picked by editors or not. This is important because managing and moderating online news comments at scale is a challenging and overwhelming task for editors. We have learned from a prior study that picking a high-quality comment often requires a series of considerations from editors that hard-coded rules could not resolve[1]. Besides, the classification model makes it possible for the researchers to understand the social impacts of the picks by evaluating the comment quality trends of individual users and promotes more engagements in the online community in the long term. The New York Times has provided a valuable dataset including comments that are flagged by professional editors as high-quality ones. Applying inductive learning algorithms to this dataset will be able to build a classifier that helps to detect high-quality comments at scale and help the platform promote users' engagements in the future. The classification model makes it possible for the researchers to understand the social impacts of the picks by evaluating the comment quality trends of individual users and promotes more engagements in the online community in the long term and help editors to moderate online comments at scale.

The source of our data was The New York Times Community API. We sampled 100,000 posts, dating from March 21, 2015 to April 2, 2015, from a dataset that was collected in an ongoing research project. Only 2,031 of the comments in this set are NYT picks; thus, the data was a very imbalanced dataset. The data was partitioned such that there was a training set of 90,000, and a test dataset of 10,000.

The features we have used in our classifiers include reply count, recommendation count (i.e., how many times the post is voted as 'recommended' by the community), negative and positive sentiment score computed by NLTK's SentimentAnalyzer, word count, the hour when the comment was created and tf-idf vector from the comment. We stemmed the words using snowballstemmer from NLTK and stripped the stop words from the text before running the tf-idf vectors in order to boost the performance. Then, we tested the following learners: Naive Bayes, Support Vector Machine (SVM), Random Forest, LinearSVC and Logistic Regression, all from the scikit-learn library for the Python programming language. As previously described, our dataset was imbalanced. In order to mitigate this problem, we applied balanced bagging classifier from imbalanced-learn library, which randomly undersamples the non-picks and train the

[1] Park, D. et al. (2016). Supporting Comment Moderators in Identifying High Quality Online News Comments. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16). https://dl.acm.org/citation.cfm?id=2858389

processed balanced data for 10 different times to improve the performance. The f1-score was around 0.2 and the average precision score was around 0.2 for all the classifiers we tried. Among all of the classifiers, Random Forest and Logistic Regression achieved the highest average precision score (0.29 and 0.25, respectively). Figures 1 and 2 are the precision-recall plots for these two models.

After running all the models mentioned before, there is another technique that we tried in order to improve the performance - stacking from mlxtend library. Base models in the stacking model included all the algorithms we tried before and the probabilities produced by these base models were used as meta-features, and BalancedBaggingClassifier with base model as logistic regression was used as the meta model to build a stacking model. The new stacking model achieved the best average precision score, 0.33. The precision-recall plot is shown in Figure 3. The details of preprocessing data, feature engineering and and models comparisons are all covered in the jupyter notebook available on the project website.

Member Contributions

Yixue Wang: Data cleaning and preprocessing, feature engineering, bagging and stacking, model evaluation, final paper

Hanlin Li: Data cleaning and preprocessing, base models testing, model evaluation, final paper, website

Katya Borgos-Rodriguez: Data cleaning and preprocessing, base models testing, model evaluation, different metrics comparison, final paper, website

Appendix:

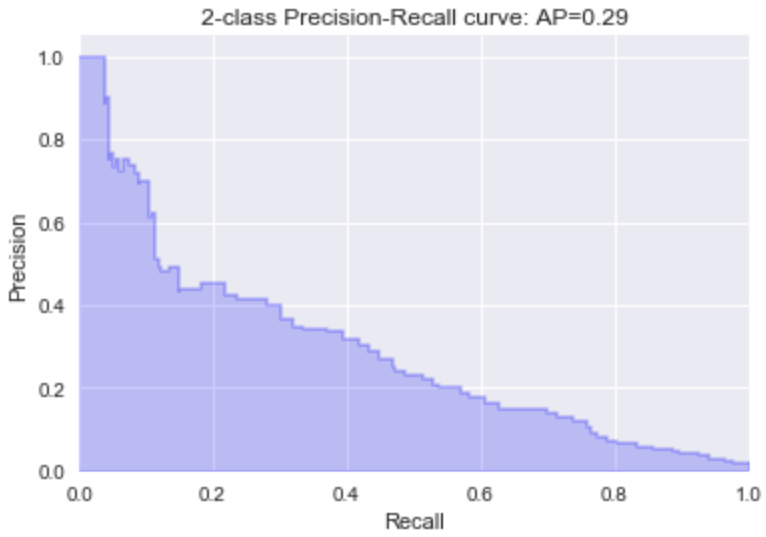

Figure 1: Precision-Recall plot, Random Forest (the number of trees = 400, class_weight = 'balanced').

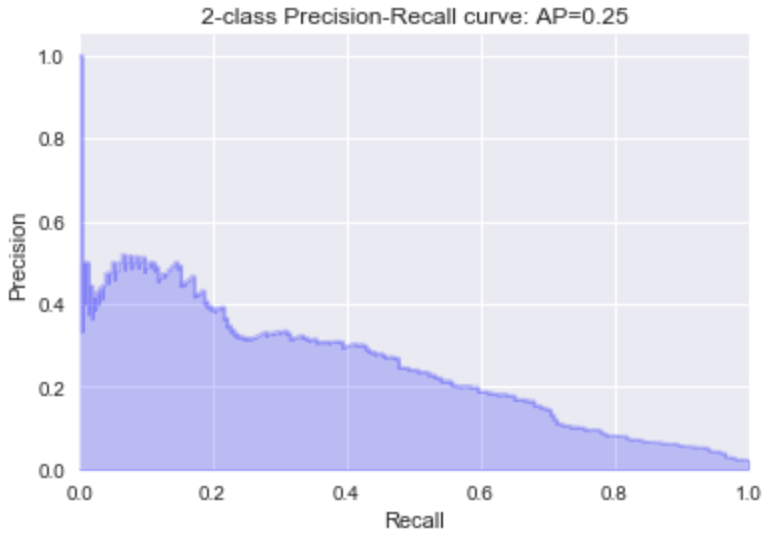

Figure 2: Precision-Recall plot, Logistic Regression using BaggingClassifier BalancedBaggingClassifier(base_estimator= LogisticRegression(), replacement = True).

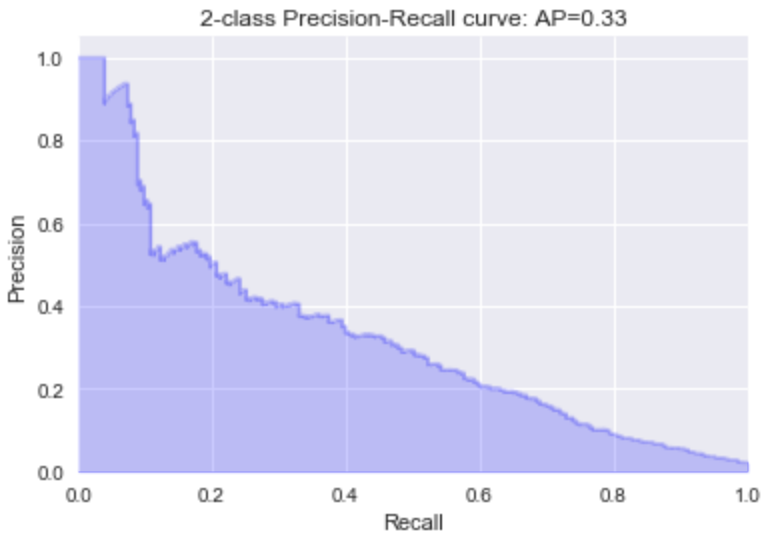

Figure 3: Precision-Recall plot, StackingClassifier.