Opinion Spam Classifier

Max Schachere
New York University
mschachere@nyu.edu

Parth Jayaram
New York University
ppj217@nyu.edu

Abstract

In a world where more and more purhcasing devcisions are made online, the problem of Opinion Spam (Fake Reviews) is becoming far more rampant. This paper aims to describe an approach to identifying and classifying fake hotel reviews using various linguistic features. The dataset consists of real hotel reviews from TripAdvisor for certain hotels and Chicago, and MTURK generated fake reviews for the same hotels. we aim for our work to help others grapple with the problem of fake reviews. We took a supervised learning approach to the problem, using Logistic Regression for classification. Through the course of feature engineering, We found that using POS and word N-Gram features improves classification performance. However, given the nature of the dataset, we are unable to determine if fake review writers behaved differently when compared to real reviewers. Hence, we are limited to using linguistic features for classification. Nonetheless, we were able to achieve Accuracy scores of up to 86% on a validation set of our data.

1 Introduction

Deception detection is an ancient task that has been attempted by humans since the evolution of language. Identifying a lie usually requires several parameters, but by far in a way the most important is the language and grammar itself. As such, deception detection has emerged as one of the most researched fields of Natural Language Processing in recent years. In addition to deception detection as an interesting topic conceptually, it has received much attention in the media and contemporary culture. Particularly with this past U.S. presidential election, there was widespread clamor to limit deceitful media, otherwise known as fake news. Fake news, however, is not new nor is it limited to media. Fraudulent reviews on websites has been a problem since the ability to review seemingly anything online. The reviews for restaurants, hotels, and products all suffer from a blight of fake reviews - what is called opinion spam. Since the user cannot determine real from fake, this decreases the authenticity and trustworthiness of the reviews overall. Our aim for this project is to develop a classifier that can reliably determine if a review is authentic or fake in order to help combat opinion spam. Primarily, we look to do so purely using linguistic properties of the review.

2 Dataset

The dataset we used was provided by Ott et al. (2011)(Ott et al., 2011). In their research, Ott et al. develop a corpus of 1,600 reviews of 20 different Chicago hotels. 800 of these reviews are real, while 800 are fake and were fabricated by workers on Amazon Mechanical Turk. In the original dataset, half of the reviews have a positive sentiment and half have a negative sentiment. For the purposes of this project, all reviews were included in a generic bag of reviews.

3 External Libraries Used

To featurize and organize our data, as well as fit our classification models we used a host of external open-source libraries. These libraries include

1. PyEnchant - PyEnchant is a spell-checking library which contains locale-specific dictionaries such as 'En-US' and 'En-UK'. This library allows spell checking by checking for

the existence of a word within the dictionary chosen.

2. Pandas - We used Pandas to organize our featurized data in table form, and to correctly split our data into three components - Train, Validation and Test. We fit our model on the training set, while tuning it on the basis of performance on the validation set. We evaluted our model against the Test set only once, to avoid overfitting to performance on the test set. The data were split by using a random sample function provided by Pandas.

3. SkLearn - We used SkLearn (Scikit Learn) to fit the various models we tested and to evaluate performance of our chosen features and model. SkLearn allowed us to quickly fit a Logistic Regression model and evaluate its performance using the F-Score, Accuracy and AUC performance measures.

4. NLTK - We used NLTK to assist with basic tasks such as tokenizing words, POS tagging as well as named Entity Recognition.

5. GrammarCheck - We tried using GrammarCheck to check for the grammatical correctness in the reviews, but unfortunately were unable to do so due to limits on the number of calls permissible to the GrammarCheck server.

4 Baseline

To benchmark our classification performance, we used a simple Bag-of-Words appraoch as our baseline. This appraoch involved extracting unigrams from our training set, and using them as features. Wherein if these unigrams were present in reviews in our validation and test-set, the unigram feature would receive a value of 1, otherwise it would have a value of 0. Using this approach, the number of features used fluctuated between 10,000 and 11,000 and we achived F-Scores ranging between 49 and 52%. In our opinion this was an effective baseline as it would be the simplest way to featurize the data, ignoring any contextual, linguistic or orthographical information present within the reviews.

5 Feature Engineering

To improve over Baseline classification performance, we used a number of orthographic (relating to the structure of words) and linguistic features extracted from the reviews. Some of these include features such as the presence of spelling errors, or the number of named entities mentioned in a review. Below is a description of all the features we tested, with how some significant features affected classification performance.

5.1 N-Gram Features

One major feature class that we used were N-Gram features, specifically Bigram features. We used both word and POS bigram featuyres to fit our classiciation model. However, in our tests, we saw that using word and Bigram features reduces Accuracy, F and AUC measures. The maximum Accuracy we achieved with these features was close to 55%. It is worth noting that adding these features also significantly increased run-times for feature extraction. Using Word and POS Bigrams improved classification performance only slightly from Baseline performance, and so we decided to leave these features out of our final feature set.

We considered using POS Unigram counts for a subset of the POS tags. For instance, originally, we used NN (including proper nouns) and JJ counts as two POS tag related features. We found that including just these two POS related features added 5% to the F-score of the classifier.

So, we decided to use all POS Unigram counts as part of our final feature set. We found that using POS tag unigrams added 8-10 percentage points to our F-score, as compared to just 5% for just JJ and NN counts.

5.2 Review-Level Features

1. Word Count - We used the word count of a review as a feature. This feature is aimed to help the classifier distinguish if there is a significant word-count difference between real reviews and fake reviews. Using this feature, we saw our F-Score improve by between 2-3 percentage points.

2. Length - We used the length of a review (character count) as a feature. This allows for reviews where the reviewer uses larger words than other reviews with similar word counts. (This feature is expected to be collinear with the average word length feature described below)

3. Average Word Length - This featues was calculated as $sum(len(word))/(len(words))$

for each word in the review.

4. First Person Frequency - This feature measured the relative number of times the reviewer used a first person pronoun in the review. These pronouns are 'i', 'we', 'me', 'us'. The number of first-person pronouns used was then divided by the word count of the review to get a relative count.

5. Misspell Frequency - This feature measured the relative number of typos in the review. Using the PyEnchant library each word in each review was spellchecked, and the number of spelling errors was divided through by the word count of the review to calculate the value of this feature. More often than not, Opinion Spammers are hired from countries where labor is cheap and so it would be fair to hypothesize that Opinion Spammers are not as proficient in the English language as the real reviewers from TripAdvisor. So, we used misspelling frequency as a proxy for low proficieny in the language. Adding this feature improved F-Scores by 1-2 percentage points.

6. Number of Words in Review Used only Once - We also tested the number of words in the review used only once as a feature. The hypothesis behind this feature was that reviews with low numbers of words used only once represent a lack of breadth in information covered in the review. In comparison, if a review uses a higher number of words used only once, it can be inferred that they are likely to cover a broader range of topics (because different topics are likely to have different words associated with them).

7. Number of unique words in the review - Similar to the word count feature, this feature counts the number of unique words in the review. Together, this feature and the previous feature added around 3 percentage points to our F-score.

8. All Caps - The relative frequency of all-caps words in each review.

9. Number - The relative frequency of numeric words in each review.

10. Alphanumeric - The relative frequency of Alphanumeric words in each review. This, and the previous two features contributed 3% to the F-score.

11. Grammar Check - Using the GrammarCheck library, we attempted to featurize the number of grammatical errors in each review. Unfortunately, the GrammarCheck Python library limits the number of calls to its service, preventing us from being able to calculate this feature for our entire dataset.

6 Classification Algorithm Selection

We evaluated a host of classification algorithms before settling on Logistic Regression. Following are the algorithms we tested with a brief explanation of what each algorithm is doing.

1. Support Vector Machine (SVM) - Support Vector Machines aim to find a decision boundary between the two classes within our data such that the distance between the decision boundary and the points in each class closest to the decision boundary is maximized. When we used SKLearn's SVM Algorithm to classify the fake reviews we experienced large amounts of instability in measures of classification performance. F-Test measures fluctuated between 50% to 65% on successive runs and so we decided against using SVM.

2. K-Nearest Neighbors - K Nearest Neighbors is a classification algorithm that involves assigning new observations the average class label of the new observation's K-Nearest Neighbors. For instance, if K is chosen to be 15, the estimated class label of a new observation is the average class label of the 15 observations closest to the new observation in the vector space. We used K-Nearest Neighbors, with K ranging between 8 and 15, and we obtained low F-scores (between 50-55%). Given that these scores were very close to our baseline performance, we decided against using KNN classification.

3. Naive Bayes Classifier - Naive Bayes Classifiers work by calculating conditional probabilities with respect to each feature and the class labels and then uses these probabilities to estimate the class of a new observation. We decided against using Naive Bayes Classification because Naive Bayes obtained even lower F-Scores than baseline performance.

4. Logistic Regression - Logistic Regression is the classification algorithm we ended up settling with. Going off research presented by Mukherjee et al., which also used Logistic Regression, we obtained the largest improvements in performance measures after featurizing our dataset and comparing against Baseline performance (Mukherjee et al., 2013). With Logistic Regression and the features described in the Feature Extraction section, F-Scores improved by up to 20%, with most runs leading to improvements of up to 15%. Logistic Regression also gave us the most stable performance measures across runs.

7 Discussion

When evaluating the performance of our chosen features and classification algorithms, we chose three metrics.

1. F-Score - The F-Score is the harmonic mean of the precision and the recall. Precision is defined as the number of a class output by the classifier divided by the true number of the members in that class. Accuracy is defined as the percentage of the classifiers output that is correctly identified. The F-score is an average measure that shows the overall performance of the classifier.

2. Accuracy - Accuracy is the proportion of the classifier's output that is correctly identified. The accuracy score was calculated using SKLearn's accuracy score function.

3. AUC Score - AUC score is a very useful metric in determining how effective the model is at correctly predicting the outcome while simultaneously mitigating the the rate of false positives. Essentially, it is a benchmark for how well the algorithm separates the two classes. This facet is crucial in evaluating this project. In deployment, a project such as this may have a very high accuracy if it simply labeled everything as real. A classifier that does exactly this - label all reviews as real - would have a 84% accuracy rate, since only 16% of reviews on Yelp are fake. (Luca and Zervas, 2016). However, this is obviously not an effective method for approaching this problem. Therefore, AUC is a crucial metric when evaluating how well this project would work on real data. Furthermore, by evaluating the associated ROC curve, a non-optimal barrier can be establish that captures a greater percentage of fake reviews, even if it does so by incurring false positives as well. In essence, an effective fake review classifier would benefit the most from overlabeling reviews as fake rather than under-labeling them.

Baseline Performance Measures:
Accuracy - 49.16%
F-Score - 59.33%
AUC - 54.89

Featurized Logistic Regression Performance Measures on Test Set:
Accuracy - 76.67%
F-Score - 75.22%
AUC - 85.42%

The chosen feature-set with Logistic Regression achieves significant improvements in classification performance over the bag-of-words baseline. The bag-of-words baseline was essentially only as good as a coin flip. Even though our method was not perfect, it still greatly outperforms manual human labeling. Ott et al. finds that across human judges, the maximum F-Score achieved was 69.7, even if the human was biased to be skeptical to the authenticity of the review.

It is important to Note that Ott et al. obtained an 89.8% F-score using Linguisting Inquiry and Word Count features (comes from proprietary software) and bigrams on an SVM classifier, but as will be seen in our concerns section, the usage of Bigrams greatly hightens the likelihood of unitinentionally overfitting to this dataset.

When comparing this classifier to other text-only classification engines on the basis of AUC, we actually see our methods yield very successful results. For instance, Jindal and Liu are only able to obtain an AUC of 0.63 on their dataset of fraudulent Amazon reviews. Although the datasets differ, our model performs in the same range if not better when it comes to one determiner of effectiveness(Jindal and Liu, 2008).

8 Division of Tasks

1. Max - Max was responsible for acquiring the dataset, writing the code to ingest the data

into python and extracting roughly 75% of the features discussed int this paper.

2. Parth - Parth was responsible for the remainder of the features as well as implementing the Baseline.

The code to evaluate the selected model, select the classification algorithm and write this paper was divided equally.

9 Concerns

It is common to say that overfitting is a concern for anyone pursuing a Machine Learning or Statistics problem, but we have taken certain steps to minimize this concern. For instance, if we did included word N-Gram features in our feature set, our features would be constrained to the words seen in the training dataset. In this case, since the dataset consists of reviews of hotels in Chicago, by using Word N-Gram features, we would be unintentionally overfitting to words seen in relation to Chicago. Also, any Word N-Gram not seen in the training dataset would be OOV, and information relating to new words in new reviews would not be captured by the feature extractor.

So, we decided to use a combination of linguistic and Orgtographic features that do not have to do with the specific words, but more with the structure of the review. But, this introduces the concern of a different kind of overfitting. Since the fake reviews were generated by MTURK workers, it could be assumed that a small number of people were responsible for the fake reviews in our dataset. Thus, by using structural features related to these reviews, we could be overfitting to the writing styles of the MTURK-ers who wrote the fake reviews in our dataset. It is unlikely that the same people will write real-world fake reviews and so our system may not generalize to real world fake reviews.

Additionally, we limited this project to purely linguistic features. Undoubtedly, there are several valuable features that cannot be captured by only looking at the raw text of the review. According to Rayana and Akoglu, metadata is a crucial factor when effectively identifying opinion spam (Rayana and Akoglu, 2015). For example, an opinion spamming account would likely post fake reviews on several products. Also, the age of the account could also be an important predictor. The deviation from the average rating of reviews is also known to be an important contributing feature(Crawford et al., 2015). Interestingly, an important feature is the username of the account that is posting the review (Fornaciari and Poesio, 2014). Account usernames that do not contain a real first or last name are known to be more suspect to posting fake reviews, according to Fornaciari (2014). By evaluating a user's behavior, a system could more holistically determine if a review is fake.

10 Conclusion

The problem of opinion spam identification is becoming an increasingly important problem to handle in our online world. On December 14th, 2017, the Federal Communications Commission voted to repeal Obama-era Net Neutrality rules, citing multiple comments on its website in favor of the repeal. Multiple investigations by independent news organizations found that many of these comments that the FCC referenced were fake, and posted under the names of individuals who are no longer alive. Opinion Spam is now affecting the laws of this country, and automating the identification of opinion spam is crucial. Our system achieves high performance on fake review identification across a host of evaluation metrics, and we believe that our system could be a great first pass at flagging possible dubious reviews (while not deciding to eliminate them altogether) for further investigation. Also, it is important to note that any real-world fake review classification system will have accesss to behavioral features of the user posting the review, which, when added to the features discussed in this paper, should produce a fairly robust system to tackle this problem.

References

Michael Crawford, Taghi M. Khoshgoftaar, Joseph D. Prusa, Aaron N. Richter, and Hamzah Al Najada. 2015. Survey of review spam detection using machine learning techniques. *Journal of Big Data*, 2(1):23, Oct.

Tommaso Fornaciari and Massimo Poesio. 2014. Identifying fake amazon reviews as learning from crowds. Association for Computational Linguistics.

Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pages 219–230. ACM.

Michael Luca and Georgios Zervas. 2016. Fake it till you make it: Reputation, competition, and yelp review fraud. *Management Science*, 62(12):3412–3427.

Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie S Glance. 2013. What yelp fake review filter might be doing?

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 309–319, Portland, Oregon, USA, June. Association for Computational Linguistics.

Shebuti Rayana and Leman Akoglu. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 985–994, New York, NY, USA. ACM.