Deceptive Opinion Spam: Yelp.com

Allen Feng, Marina Siu Chong, Nikhil Gosike, William Chen

Abstract

In modern society, technology has allowed for the rapid sharing of information. Companies have made complete businesses around compiling reviews for services to help consumers make educated decisions. As more trust is put into these peer reviews, Deceptive Opinion Spam becomes a bigger and bigger issue. This paper's goal is to delineate an approach to identifying and classifying fake reviews on the Yelp platform specifically. Our approach uses linguistic features of reviews to classify them as real or fake. The dataset that we are using consists of real reviews on Yelp along with fake reviews made by Turks and reviews that have been flagged by Yelp's detection algorithm. We approached this problem using a supervised learning approach, and used Logistic Regression for classification. Through our feature engineering process, we found a feature set that was able to greatly improve classification performance. However, because of the lack of access to meta-data, our system is limited to linguistic features for classification. Even without this meta-data, however, we were able to achieve F-Scores of up to 79%

1 Introduction

Opinion mining, also called sentiment analysis, is the field of study that analyzes people's opinions, evaluations, and emotions from their written language. It is currently one of the most researched areas in natural language processing since opinions are central to almost every human activity. Fake reviews detection is a major section of opinion mining. Many companies and researchers have built detection models using linguistic features from the review's text content, and meta-data features such as star rating, user ID of reviewer, time review was posted. etc. to determine the authenticity of reviews.

Fake reviews have become so widespread and can be found on almost every website, but they also are very hard to recognize. Our aim for this project is to develop a classifier using linguistic properties that can reliably determine the authenticity of reviews in order to combat fake reviews.

2 Previous Works/Research

We looked into previous works that researchers have done to combat opinion fraud in order to better understand existing solutions. In *Towards a General Rule for Identifying Deceptive Opinion Spam*, researchers attempt to identify more universal and general cues towards identifying such reviews. One of the main issues presented here is the rule of lack of spatial details in fake reviews. Using a cross-domain "gold-standard" dataset (Hotel, Restaurant, and Doctor) as well as a feature-based additive model known as SAGE (Sparse Additive Generative Model) the experiments tests unigram, LIWC (Linguistic Inquiry and Word Count), and POS (Part-of-Speech) features. It observes that Unigram tends to outperform LIWC and POS features for restaurants and hotels. In terms of cues, it also identifies that truthful

writings tend to use more nouns, adjectives, prepositions, determiners, and coordinating conjunctions in the case of POS. In terms of LIWC, the articles considers space, sentiment, and use of first person pronouns. Spatial details seemed inconclusive in identifying, while sentiment and first-person pronouns seemed to be overused when identifying fake reviews. One issue raised in the course of the study is when fake reviews are employee-generated or customer-solicited. With these types of fake reviews, employees and customers tend to include many indicators of a truthful review such as spatial detail.

In another article *The Enemy in Your Own Camp: How Well Can We Detect Statistically-Generated Fake Reviews - An Adversarial Study*, researchers experiment with how well a logistic regression model can distinguish real from fake reviews from models that 1) have access to meta-information and 2) do not have access to meta-information. They also test how well human judges compare when detecting fake reviews generated by the model with meta-information. The logistic regression model they used to detect fake reviews was regularized with L2 norm and fit it on a data set of 10,000 true reviews and varying amount of fake reviews. The base features of the classifier are word n-grams with n ranging from 1 to 4. They measured F1 performance over 5 fold stratified cross-validation and compared performance of the logistic regression model on a test set including fake reviews generated by an unconditioned 7 gram LM and a test set whose fake reviews have been conditioned on meta-information, to establish whether conditioning LMs on demographic information has any effect on detection. The results of the experiments showed that using access to meta-information can significantly improve detection of fake reviews, but that generated reviews conditioned on meta-information are also considerably harder to detect than ones generated without. Furthermore, human judges have a much lower detection rate than the logistic regression model; therefore, indicating the viability of an adversarial setup to test detection tasks.

The final article *Finding Deceptive Opinion Spam by Any Stretch of the Imagination* compares three more approaches to detecting fake reviews and ultimately develops a classifier with 90% accuracy. In it, researchers compare a text categorization task, which uses n-gram classifiers to label writings as truthful or deceptive, an instance of psycholinguistic deception detection that makes use of LIWC features, and a problem of genre identification which views truthful and deceptive reviews as sub-genres of imaginative and informative writing. Like the previous article, this one finds that human judges are not effective in finding deceptive opinions online as they suffer from truth-bias. Based on a 5-fold nested cross validation procedure, all three automated approaches outperformed human significantly. Like the first article, it found that POS features in truthful reviews tended to make more use of nouns, adjectives, prepositions, etc. Using the genre identification as a baseline, the former two approaches outperform, with psycholinguistic performing 3.8% more accurately and text categorization 14.6% to 16.6% more accurately. However, combining the LIWC+Bigrams achieved even better performance at 89.8% accuracy.

3 Dataset

For our systems, we will be using two datasets that provide both truthful and deceptive reviews. The first is provided by Ott (2011)(Ott at el.). This corpus consists of 1600 fake and real reviews of 20 Chicago

hotels. Of these 800 are real, while the other 800 are fake and fabricated through the crowdsourcing platform Amazon Mechanical Turk. The dataset also indicates sentiment of the reviews, of which half are positive and half are negative. The truthful reviews are also sourced from TripAdvisor, Expedia, Hotels.com, Orbitz, Priceline, and Yelp.

The second dataset is the Yelp Filter Dataset, which was provided by Professor Bing Liu from The University of Illinois at Chicago, who gave us permission to use it (2013)(Mukherjee et al.). For their research, they compiled a dataset of Yelp reviews which are flagged with a Y or N marker. The Y indicates a fake reviews that Yelp determined to be fake through their own algorithms, while the N indicates real reviews. It should be noted that these are all restaurant reviews. We were unable to estimate the accuracy of the dataset.

In order to split the data in training, validation (or development), and testing sets, we randomly split the data using a 70%, 15%, and 15% split, respectively. This was done using the Pandas *sample()* function.

4 External Libraries Used

1. SkLearn or SciKit Learn - SciKit Learn was used to fit the different models and evaluate our features. For example, it allowed us to quickly and efficiently fit a Logistic Regression, Naive Bayes, and K Nearest Neighbor models. In addition, we could use the SciKit metrics library to quickly and easily calculate our three performance metric: Accuracy, F-Score, and Area Under the Curve (AUC).
2. Pandas - Pandas was used in order to arrange our data into an easy-to-use table or DataFrames. Furthermore, we used the *sample()* function to correctly and randomly split our data into the training, validation, and testing sets. The DataFrames created with the baseline and our model also worked well to include features as columns and fit into the SciKit Learn models.
3. NLTK - NLTK was used to help us tokenize and tag the data with parts of speech in order to featurize the data. This also assisted us in removing any and all words that NLTK considers stop words.
4. PyEnchant - PyEnchant is a useful spell-checking library that includes specific location dictionaries such as “En-US” (English, US version). This library tells use whether words are spelled correctly according the US English dictionary. It also lets us know whether these words exist within the current chosen dictionary.
5. GrammerCheck - GrammerCheck is a python language library used to check for grammatical correctness within text. We wanted to include this as a feature, but found that it was not able to download using the pip command within terminal. Even with all necessary requirements, the library signaled errors when trying to download it.

5 Baseline

We used a baseline model consisting of a simple bag of words approach to benchmark our performance. This procedure consists of the extraction of unigrams from the training set, and using those as features. This entails marking unigrams present in both the validation and test-set with a 1 and marking with a 0 otherwise. We chose this approach, as we thought this was a very effective baseline. In our opinion, it was the simplest way to featurize the data as it ignores other factors such as contextual,

linguistic, and orthographical information that could be extracted from the data. We found that the number of features fluctuated between 9,000 to 10,000 features. Using this approach, we obtained F-Scores in the range of 56% to 65%.

6 Feature Engineering

In order to improve the performance over the baseline classification, we will be using a number of other features such as grammar, length of review, presence of spelling errors, number of unique words mentioned, and perhaps even the time the review was posted.

6.1 N-Gram Features

One major feature that we experimented with was N-Gram features, specifically Bigrams. We attempted to implement word and POS bigram features, but the tests showed lower Accuracy, F and AUC measures. Integrating these features also increased runtime by a significant amount. We therefore decided to exclude these features from our final feature set.

6.2 Review-Level Features

1. Word Count: We used review word count as a feature. The goal of this feature is to help determine if there is a significant word count difference between real and fake reviews.
2. Length: Using the length of reviews as a feature helps differentiate reviews that use larger words than other reviews with similar word counts.
3. Average Word Length: Calculated as sum(len(word))/(len(words))
4. Number of Once-Used Words Per Review: We used this metric as a feature to be used as a proxy for number of topics covered in a review. We theorized that a low number of words used only once represented a limited breadth of information covered in the review. In contrast, a review containing a larger number of words used only once represented a likelihood that more topics were covered in the review.
5. Number of Unique Words: This feature counts the number of unique words in each review to analyze breadth of vocabulary. Combined with Number of Once-Used Words Per Review, this added approximately 2 percentage points to the F-Score.
6. Misspell Frequency: We thought that number of typos per review was also a good differentiators between real and fake reviews. We used the PyEnchant library to spell-check each word in the review. We then divided the number of spelling errors by the word count for the review to standardize the feature. Often times, Opinion Spammers are laborers in foreign countries where these tasks are cheaper than the US. Because of this, fluency in the English language is usually lacking. By using the misspell frequency as a detector for low proficiency in the language, we see improved F-Scores.
7. First Person Frequency: By accounting for the number of times a review uses first person pronouns, we are able to differentiate between reviews based on frequency of first person use. We used the pronouns ‘I’, ‘we’, ‘me’, ‘us’. We also divided the number of first person pronouns (per review) by the word count to standardize this feature.
8. Caps: Frequency of all-caps words in the review.
9. Numbers: The frequency of numeric word appearances

10. Alphanumeric: Frequency of Alphanumeric words in the reviews.

7 Classification Algorithm Selection, Sample Generation and Results

We evaluated and tested a selection of classification algorithms including K-Nearest Neighbors, Naive Bayes Classifier, and Logistic Regression. We eventually settled on using Logistic Regression because it performed the best. Below is a more in depth explanation of what each algorithm does.

1. K-Nearest Neighbors is a classification algorithm that involves assigning new observations the average class label of its K-Nearest Neighbors. For example, if K is set to 15, the estimated class label of a new observation is the average class label of the 15 observations closest to it in the vector space. We used numbers 8 through 15 as K and obtained F-scores between 50 and 55%. Since these scores showed little improvement over our baseline, we decided not to use K-Nearest Neighbors as our final classification algorithm.
2. Naive Bayes Classifier is a classification algorithm that works by calculating conditional probabilities with respect to each feature and the class labels and uses these probabilities to estimate the class of a new observation. We decided not to use Naive Bayes Classification because the F-scores obtained were even lower than what we got from baseline.
3. Logistic Regression is a predictive analysis classification algorithm that uses data to explain the relationship between one dependent binary variable and one or more independent variables. Basing our logistic regression off of research presented by Mukherjee et al., we got the largest improvements in performance measures after featuring our dataset and comparing it against baseline performance. Using logistic regression along with the features described in Feature Extraction section, F-Scores improved significantly (up to 20%).

8 Discussion

In order to evaluate the performance of our chosen classification algorithm as well as all our features, we calculated the F-Scores, Accuracies and AUC scores. These metrics help provide quantitative feedback on our algorithm and features.

1. The F-Score is the harmonic mean of the precision and the recall. The Precision is the number of a class output by the classifier divided by the true number of the members in that class, and accuracy is the percentage of the classifier's output that is correctly identified.
2. Accuracy is the proportion of the classifier's output that is correctly identified. We calculated the accuracy using SKLearn's accuracy score function.
3. The AUC score is used to determine how effective the model is at correctly predicting the outcome while also mitigating the rate of false positives. It is a benchmark for how well the algorithm separates the two classes. The AUC score is very important because it takes into account false positive rates. Without it, the algorithm could just label every review as real and receive a very high accuracy score since most of the reviews are real. We need the AUC score to ensure that the classifier is actually labeling reviews

as fake. A fake review classifier that over-labels reviews as fake is much more effective than one that under-labels reviews as fake.

Using Mechanical Turk Dataset

Baseline Performance:
Accuracy: 47.81%
F-score: 63.77%
AUC: 56.11%

Featurized Performance using Logistic Regression
Accuracy: 55.31%
F-score: 71.23%
AUC: 54.33%

Featurized Performance using K-Nearest Neighbors
Accuracy: 52.92%
F-score: 5.04%
AUC: 53.49%

Featurized Performance using Naive Bayes Classifier
Accuracy: 50.42%
F-score: 1.65%
AUC: 50.42%

Using Yelp Filtered Dataset

Baseline Performance:
Accuracy: 53.04%
F-score: 61.66%
AUC: 50.55%

Featurized Performance using Logistic Regression
Accuracy: 60.59%
F-score: 74.42%
AUC: 57.38%

Using logistic regression as a classifier for the featurized reviews showed higher performance levels in accuracy and F-Score measurement as opposed to the unigram baseline performance. While the KNN and NBC classifiers also demonstrated higher accuracy, it greatly suffered in F-Score measurements. Since F-Score is the weighted average of precision and recall, it also takes into account the likelihood of a false negative or positive. F-score is typically a better measurement of performance, especially when there is an uneven class distribution. Therefore we decided to move forward using logistic regression when testing the performance using the filtered dataset.

When using the filtered dataset, the F-Score and Accuracy performance using the featurized reviews was on average 7% and 13% higher respectively compared to the baseline measurements. It is important to note that the filtered dataset uses Yelp's algorithm for filtering out reviews which it deems to be spam.

9 Division of Tasks

Nikhil Gosike and William Chen are responsible for gathering the datasets and coding most of the project together. Marina Siu Chong and Allen Feng are responsible for the remainder of the code, research, compilation of result, and evaluation of the work.

10 Concerns

Overfitting is a common concern for most solutions to Machine Learning and Statistics problems. However, we have taken steps to reduce this risk to a minimum. If we had included N-Gram features in our feature set, our features would be at risk of overfitting due to our features being constrained to the words that appear in the dataset. If we were to use Word N-Grams,

any word that did not appear in our dataset would be considered OOV.

Instead, our approach uses a combination of linguistics and orthographic features to avoid word-specific issues. Our approach instead focuses on the structure of review text rather than a dictionary of words and their likelihoods. However, this subjects our features to another type of overfitting. Because the fake reviews in the dataset were written by Turks, it is possible that a minimal amount of people were involved with creating the reviews in our dataset. In that case, our dataset would be heavily skewed to the structure of few Turks. Because of this, our system may not work as well in real world scenarios. Furthermore, the Yelp dataset also cause some concerns. The Yelp Filter dataset comes from Yelp themselves, which uses their own proprietary algorithms to identify fake reviews, which are flagged. It should be noted that part of this dataset may not correctly identify truthful and deceptive reviews.

Our final concern is that we only explored the use of linguistic features. The existence of robust metadata could be potentially very valuable to distinguishing fake and real reviews. Star rating, user ID of reviewer, time review was posted, and age of account are all valid differentiators that could separate fake reviews from real reviews. Star ratings that are wildly different from the average and young accounts that make many reviews immediately could be signs for a false review. Taking these sorts of meta-data into account could make for a much more robust system that does not depend only on linguistic features.

11 Conclusion

As more and more people rely on online reviews before making any purchasing decisions, more fake reviews are also surfacing the web, making it extremely difficult for customers to differentiate between real and fake. Without trust, none of these opinions will matter, and a huge resource for consumers will be invalidated. Fake review detection is needed to create a better and more transparent world for everyone.

Our system achieves a high performance on fake review identification across a host of evaluation metrics. We've tested various classification algorithms as well as features before settling on the best ones. Although it is not perfect, we believe that our system could be a great first pass at flagging fake reviews for further investigation. Further investigation could include checking the identity of the user who posted the review as well as analyzing the behavioral features of the user posting the review (data that we do not currently have). When combining these techniques with what we have created, companies will be able to eliminate most fake reviews from their websites, instilling consumer trust again.

References

Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie Glance. What Yelp Fake Review Filter Might Be Doing. Proceedings of The International AAAI Conference on Weblogs and Social Media (ICWSM-2013), July 8-10, 2013, Boston, USA.

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.

Li, Jiwei & Ott, Myle & Cardie, Claire & Hovy, Eduard. (2014). Towards a General Rule for Identifying Deceptive Opinion Spam. 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference. 1. 1566-1576. 10.3115/v1/P14-1147.

Hovy, Dirk. (2016). The Enemy in Your Own Camp: How Well Can We Detect Statistically-Generated Fake Reviews – An Adversarial Study. 351-356. 10.18653/v1/P16-2057.

Ott, Myle & Choi, Yejin & Cardie, Claire & Hancock, Jeffrey. (2011). Finding Deceptive Opinion Spam by Any Stretch of the Imagination.

Fornaciari, Tommaso & Poesio, Massimo. (2014). Identifying fake Amazon reviews as learning from crowds. 279-287. 10.3115/v1/E14-1030.