# Spam Detection in Hotel Reviews

Zhiyu Lin, Audrey Yuan, Yiran Liu, Yongzheng Chen

## INTRODUCTION

As global networks rapidly develop, more and more industries are moving their businesses online. Customers can purchase products, book tickets and hotels, and even pick favorite restaurants through online applications. User-generated online reviews become one of the most important references for customers making purchase decisions. Unfortunately, among online reviews, a portion of them are deceptive. These reviews are deliberately written to sound authentic and misleads customers into making purchase decisions. In order to solve this problem, our team proposes to employ supervised machine learning algorithms to detect deceptive opinion spam, specifically we are interested in answering the question "*Do Classifiers Effectively Detect Opinion Spam?*". Our team will build several machine learning classifier models and evaluate the performance of each classifier model in detecting deceptive opinion spam.

## DATA

We have a corpus consists of truthful and deceptive hotel reviews of 20 Chicago hotels, the corpus contains 400 truthful positive reviews from TripAdvisor, 400 deceptive positive reviews from Mechanical Turk, 400 truthful negative reviews from Mechanical Turk. This dataset consists of 20 reviews for each of the 20 most popular Chicago hotels.

We build a Preprocessor class to clean and tokenize the reviews, removed all the punctuations in the corpus and the stopwords as well. The lemmatizer we used is WordNetLemmatizer, in the parameter of lemmatizer we put the results of the pos_tag in it, which are the token's text and token's tag, and TweetTokenizer as tokenizer.

## EDA

### Training Data Set


*Figure 1. Word clouds for truthful and deceptive reviews categorized by stance.*

From the simple word clouds, we observe that all four categories feature "hotel" and "room". However, there are some interesting words that stand out and could potentially be important features for prediction. For example, comparing the truthful positive and deceptive positive reviews, we see that deceptive reviews highlight "great" and "staff", whereas these two words do not seem important in truthful reviews at all.

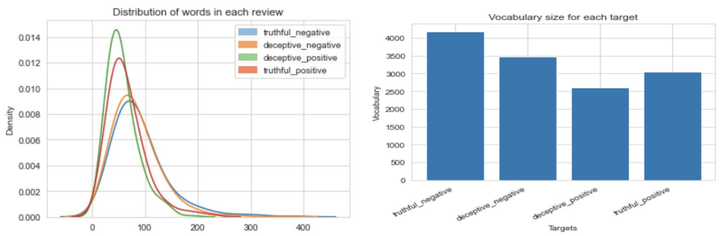



*Figure 2. Distribution on the number of words (left) and vocabulary size (right) for truthful and deceptive reviews categorized by stance.*

In the density plot, we see that negative reviews are more left-skewed than positive reviews, meaning that negative reviews tend to be shorter than positive reviews. We do not observe significant differences in review length distribution between deceptive and truthful categories while controlling for stance. This gives us confidence that the model will be fitted on a balanced dataset. On the right hand side, we observe that the vocabulary sizes are roughly equal across categories while controlling for stance.

### Test Data Set

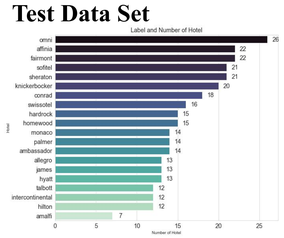



*Figure 3. Reviews per hotel*

(Fig 3.) In test data set, there are total 320 reviews from 20 different hotels. The distribution of each hotel is shown in the "Label and Number of Hotel" chart. Omni is the most frequent hotel with 26 reviews and Amalfi is the least frequent hotel with 7 reviews.

(Fig 4.1) There are 23373 tokens and 3497 distinct words in the test data set. Overall, 14.96% of tokens are vocabulary.

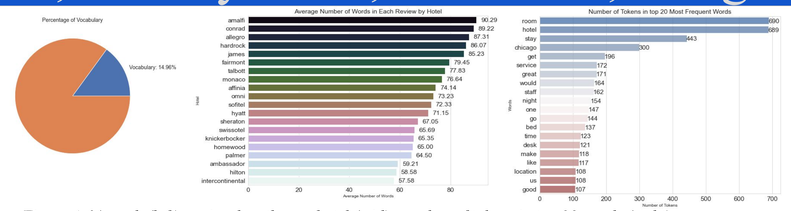



*Figure 4. %vocab (left), review lengths per hotel (mid), number of tokens in top 20 words (right)*

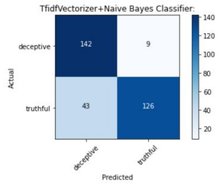
(Fig 4.2) On average, there are 73.04 words in each review. In "Average Number of Words in Each Review by Hotel" chart, Amalfi hotel has the highest average number of words in each review 90.29, and Intercontinental hotel has the lowest average number of words in each review 57.58.

(Fig 4.3) There are 4371 tokens corresponding to the top 20 most frequent words in the vocabulary. The "Number of Tokens in top 20 Most Frequent Words" chart is shown the distribution of top 20 most frequent words, "room", "stay", "service", "call", "staff", "desk" are very frequent words in reviews.

## MODELS

### Naive Bayes

Naive Bayes method is a classifier based on bayes' theorem and independent assumption of characteristic conditions. For a given training data set, we first learn the joint probability distribution of input and output based on the independent assumption of feature conditions, and then, based on this model, we use Bayes' theorem to find the output y with the maximum posterior probability for a given input x. In this project, we used 'Authenticity' column as our label and TfidfVectorizer to transform the dataset into vectors, then we used train_test_split function in sklearn to split the dataset into train data and test data. After the model's prediction, we used accuracy, precision, recall and Macro F1 as the indicators of the performance of the model, and confusion matrix as well.

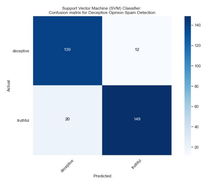



| | precision | recall | f1-score | support |
|---|---|---|---|---|
| deceptive | 0.74 | 0.97 | 0.84 | 157 |
| truthful | 0.96 | 0.67 | 0.79 | 163 |
| accuracy | | | 0.82 | 320 |
| macro avg | 0.85 | 0.82 | 0.81 | 320 |
| weighted avg | 0.85 | 0.82 | 0.81 | 320 |

```
Accuracy: 0.8375
Precision: 0.8504504504504504
Recall: 0.8429797405854462
Macro F1: 0.837092731829574
```

### SVM

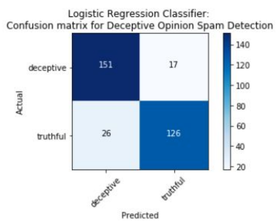
A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. In the project, "Authenticity" as the label data with two categories Deceptive and Truthful, and Reviews in the training dataset are used to train SVM model to predict the review either deceptive or truthful. The project evaluates the linear SVM with Countvectorizer and TF-IDF vectorizer, by comparing macro-average F1-score, linear SVM with TF-IDF vectorizer has better performance with macro-average F1-score 0.90.




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| deceptive | 0.87 | 0.92 | 0.90 | 151 |
| truthful | 0.93 | 0.88 | 0.90 | 169 |
| accuracy | | | 0.90 | 320 |
| macro avg | 0.90 | 0.90 | 0.90 | 320 |
| weighted avg | 0.90 | 0.90 | 0.90 | 320 |

### Logistic Regression

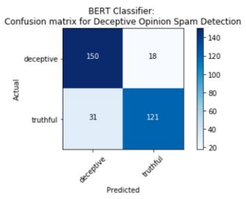
We used another supervised learning model logistic regression to classify authenticity. For logistic regression model, we take the processed textual data and transform to TF-IDF vectors as the input data.




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| deceptive | 0.85 | 0.90 | 0.88 | 168 |
| truthful | 0.88 | 0.83 | 0.85 | 152 |
| accuracy | | | 0.87 | 320 |
| macro avg | 0.87 | 0.86 | 0.86 | 320 |
| weighted avg | 0.87 | 0.87 | 0.87 | 320 |

### BERT

We also used BERT which is a transformer-based machine learning model for our classification problem. For BERT model, it is different from logistic regression. We pass in the textual data and BERT will tokenize the texts by inserting some tokens to indicate the start and end of the sentence. BERT will also use mask to separate different sentences. Each token will be mapped to a token id. Position embeddings are also added to indicate the position of each token. After tokenizing the input, the token ids as well as the attention masks will be fed into classifier.




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| deceptive | 0.83 | 0.89 | 0.86 | 168 |
| truthful | 0.87 | 0.80 | 0.83 | 152 |
| accuracy | | | 0.85 | 320 |
| macro avg | 0.85 | 0.84 | 0.85 | 320 |
| weighted avg | 0.85 | 0.85 | 0.85 | 320 |

## MODEL EXPLANABILITY

| | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| **Naive Bayes** | 0.82 | 0.85 | 0.82 | 0.81 |
| **SVM** | 0.90 | 0.90 | 0.90 | 0.90 |
| **Logistic Regression** | 0.866 | 0.867 | 0.864 | 0.864 |
| **BERT** | 0.868 | 0.868 | 0.869 | 0.869 |

*Table 1. Model performance report comparison.*

The table on the left shows the accuracy, precision, recall and F1 for each of the models tested. SVM achieves the highest metrics in all four categories, although all models have relatively decent results of over 80%.

To further understand the model predictions, we applied LIME and SHAP explainers and their results are shown below. LIME creates multiple "perturbed" instances around the instance to be explained, and builds a linear model for local interpretation. SHAP utilizes Shapley game theory and builds a global explainer with local smoothing. In the top two figures by LIME, we observe two instances of reviews predicted as truthful and deceptive, respectively, and the keywords helped the model to make these decisions are highlighted. The two figures at the bottom are population level model explanations produced by SHAP.


*Figure 5. LIME instance explanation for a negative prediction*

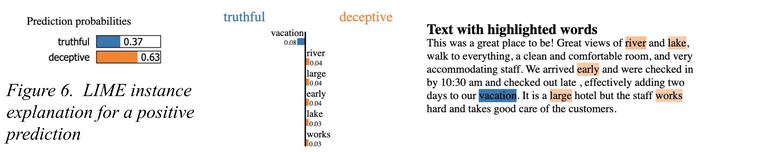



*Figure 6. LIME instance explanation for a positive prediction*

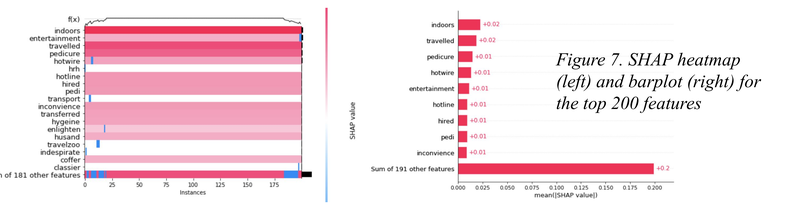



*Figure 7. SHAP heatmap (left) and barplot (right) for the top 200 features*

## CONCLUSION

Based on the evaluation on the test set, we can observe that BERT model is slightly better than logistic regression and Naive Bayes, but not as good as SVM. This could be because that the dataset is not a complex task. For more complex tasks, BERT model can be expected to perform better. But overall, all the models we trained made pretty accurate predictions on this dataset.

In the future, we could improve by incorporating word2vec embeddings instead of using TF-IDF and count vectorizers, or include more feature engineering techniques before modeling.

**CONTACT INFO:** zl281@georgetown.edu, ay432@georgetown.edu, yl1068@georgetown.edu, yc903@georgetown.edu