Detecting Fake Reviews on Yelp

Jiajun Bao[1], Meng Li[1] and Jane Liu[1]

[1] Department of Computer Science, New York University, New York City, United States

Abstract

Customers are increasingly relying on online reviews when making a decision to purchase a product or service. According to a study conducted in 2016 by the Pew Research Center[1] 50 percent of all American adults always check online reviews before making a purchase. When accounting for individuals who occasionally check online reviews before making a purchase this number jumps to 82 percent. Various marketing research firms believe this number may be even higher—up to 90 percent. Because of the outsized influence of online reviews on consumer purchasing decisions a shadow industry has emerged where wrongdoers create false reviews to artificially promote or devalue products and services. The detection of fake reviews, also known as opinion spam, has led to much research in the past decade. The majority of research in recent years has focused on machine learning—specifically natural language processing. The goal of this paper is to offer a comparative study of several machine learning techniques for opinion spam detection and to evaluate their accuracy. We chose Yelp restaurant reviews for our training, testing, and validation datasets.

Keywords: fake user reviews, opinion spam, spam detection

---

1. Introduction

Online reviews have become widely used by individuals and organizations to make purchase and business decisions. Positive reviews can result in significant financial gains and fame for businesses and individuals. A 2011 study shows that a one-star increase in Yelp rating leads to a 5-9 percent increase in revenue for independent restaurants without a chain affiliation.[1] Unfortunately, this also creates a strong incentive for imposters to manipulate the system by creating fake reviews to promote or discredit a target business and their products. These individuals are called *opinion spammers* (Jindal and Liu 2008). In the past few years, the problem of opinion spam has become widespread, and many high-profile cases have been reported in the news [2].

While positive reviews can temporarily boost a business's revenues, negative review spam can cause permanent, irreparable damage due to loss of consumer trust. This issue is serious enough to have attracted the attention of mainstream media and governments. The BBC and New York Times have reported that "fake reviews are becoming a common problem on the Web, and a photography company was recently subjected to hundreds of defamatory consumer reviews" [3]. In 2014 the Canadian government issued a warning that urged consumers to be wary of fake online endorsements that appear to have been made by ordinary consumers. Because review spam is a harmful and widespread problem that can damage local economies developing effective and reliable methods to distinguish truthful from false reviews is an important and ongoing challenge.

Current research show that businesses have started to "pay" customers for positive reviews using incentives, such as coupons, freebies, promotions, and even outright cash to encourage positive review posts. In fact the prevalence of fake reviews has proliferated to such an extent that Yelp.com has launched a "sting" operation to publicly shame businesses who buy fake reviews [4]. Since researchers first began studying methods of opinion spam detection a wide variety of techniques have emerged for detecting individual [5] and group [6] spammers. Recent research has focused on supervised and unsupervised learning techniques for opinion spam detection due to their efficacy.

For our project we attempted to build several binary classifiers that take the review text and metadata about the review as input and outputs whether the review is truthful or

[1] Pew Research Center. (2016). Online Shopping and E-Commerce. [Data file]. Retrieved from https://www.pewinternet.org/2016/12/19/online-shopping-and-e-commerce/

fake. The learning algorithms we experimented with include Logistic Regression, Neural Networks, topic modeling with Latent Dirichlet Allocation, and k-Nearest Neighbors. The results show that the Neural Network model offers the best performance with an accuracy rate of 71 percent.

2. **Related Works**

Dixit et al proposed three categories for review spam [10]: (1) Untruthful Reviews—reviews that falsely praise or disparage a business for the purpose of benefitting or undermining that business—the subject of this paper, (2) Reviews on Brands—reviews that discuss a brand or seller but do not actually discuss the product, and (3) Non-Reviews—reviews that contain unrelated marketing, advertising, or other spam text. Untruthful reviews are of great concern due to their ability to to undermine the integrity of the online review system. Detecting fake reviews is a challenging task because it is nearly impossible for a human being to distinguish between fake and real reviews by simply reading them [9].

2.1 Feature Engineering for Review Spam Detection

Feature engineering is the process of constructing or extracting features from data. In this section, we describe some commonly used feature extraction techniques for review spam detection. One of the most common types of features that can be extracted from reviews are the words found in the review's text. This is typically implemented using the bag-of-words approach, where features for each review consist of either individual words or small groups of words known as ngrams, found in the corpus. Researchers have also used other characteristics of reviews including metadata about the review itself, the reviewer, and and the products itself [11]. Features can be broken down into two categories of review- and reviewer-centric features. Review-centric features are constructed using the information contained in a single review while reviewer-centric features account for qualities and characteristics of the author. Often spammers leave behavioral "footprints", such has having multiple user accounts, that can be used to identify them.

Among review-centric approaches individual words and *n*-contiguous words from a given sequence are chosen. These are typically referred to as unigrams, bigrams, trigrams, etc. These features are used by Jindal et al. [12], Li et al. [8] and Fei et al. [13]. Fei et al. observed that using ngram features alone proved inadequate for supervised learning when learners were trained using synthetic fake reviews, such as those created by Amazon Mechanical Turk, since the features created were not present in real-world fake reviews and turkers often don't share similar traits with actual restaurant patrons in the target city or region.

Another review-centric approach is Part of Speech (POS) tagging, which involves tagging word features with their grammatical part of speech based on its context and definition within the sentence [15]. Ott et al. [9] achieved better results by also including these features than with bag-of-words (ngrams) alone.

Metadata about a review, such as its length, date, time, rating, reviewer ID, review ID, business ID, and feedback can also be useful for classifying a review as real or fake.

2.2 Supervised Learning

Supervised learning can be used to detect review spam by looking at it as a classification problem where reviews are separated into two classes—fake and non-fake reviews. To the best of our knowledge the first researchers to have studied deceptive opinion spam using supervised learning were Jindal et al., who discussed opinion mining techniques [12]. Ott et al. [9] devised and compared three approaches for performing fake review detection. For their study, they produced a new dataset using Amazon Mechanical Turk to create deceptive reviews for TripAdvisor. The deceptive reviews were created by requesting a group of people to deliberately write 400 fake reviews expressing positive sentiment (i.e., 5 star reviews) for a set of hotels. Another 400 "truthful" 5-star reviews were collected from the TripAdvisor website for the same hotels. The resulting dataset consisted of 800 reviews—400 fake and 400 non-fake. All reviews expressed positive sentiment towards the hotels. In a later paper they created a second dataset of the same size and similarly balanced, but of negative sentiment (i.e., 1 and 2 star reviews). They combined the entire set of dat and claimed this was the first "gold-standard" dataset for review spam.

2.3 Unsupervised Learning

Because of the difficulty of producing accurately labeled datasets of review spam, supervised learning is sometimes not an option. Unsupervised learning provides a solution for this, as it doesn't require labeled data. A novel unsupervised text mining model was developed and integrated into a semantic language model for detecting untruthful reviews by Raymond et al. [3] and compared against supervised learning methods. Their model creates an approximation method for

calculating the degree of untruthfulness for reviews based on the duplicate identification results by estimating the overlap of semantic contents among reviews using a Semantic Language Model (SLM). Besides performing unsupervised review spam detection, they also developed a high-order concept of association mining to extract context-sensitive concept association knowledge. Their model presumes that if the semantic content of a review is close to those of another review, it is likely that the two reviews are duplicates and thus examples of spam reviews.

2.4 Semi-supervised Learning

In other domains, it has been found that using unlabeled data in conjunction with a small amount of labeled data may be able to improve learner accuracy considerably compared to completely supervised methods [17]. In a study by Li et al. [8], a two-view semi-supervised method for review spam detection was created by employing the framework of a co-training algorithm to make use of the large amount of unlabeled reviews available. The co-training algorithm was developed by Blum and Mitchell [18], and is a bootstrapping method that uses a set of labeled data to incrementally apply labels to unlabeled data. It trains two classifiers on two distinct sets of features and adds the instances most confidently labeled by each classifier to the training set. This effectively allows large datasets to be generated and used for classification, reducing the demand to manually produce labeled training instances.

Research in opinion spam detection has mostly focused on logistic regression and support vector machines. However, recent research has started to shift toward deep learning models [20]. In our project, we experiment with various supervised learning techniques for fake review detection, including logistic regression, neural networks, latent dirichlet allocation, and k-nearest neighbors.

3. **Dataset and Preprocessing**

A major challenge in the study of fake review detection is the difficulty of obtaining reliable or gold-standard fake review data. This has been an ongoing difficulty and existing works have mostly relied on ad-hoc fake and non-fake labels for model building. We obtained our dataset from ODDS (Outlier Detection Datasets), a data mining research group at the State University of New York at Stony Brook that provides ground truth data containing outlier or anomalous information for various industries, including fake restaurant reviews from Yelp.

3.1 Data Exploration

From the ODDS dataset we chose two files—*metadata.txt*, which contains meta information about the reviewer and *reviewContent.txt*, which contains the actual reviews. The metadata file contained Yelp user IDs and a column that indicates if the user is a spammer or an actual person. A value of 1 indicates a real person while -1 indicates a spammer. The reviewContents file contains user IDs and all the reviews.

We performed data exploration with the pandas library. We started by looking for columns that could help our predictive model determine if a review is truthful or fake. The pandas library has great features for exploring numerical data, such as income, age, or mortgage rates, but other than review ratings not much of our data was numerical. User ID and date are displayed as numbers but meaningful calculations could not be extracted from them. Therefore, we mostly looked at count and frequency values and discovered that (1) the ODDS data is well-formatted and there were no missing values in any of the columns for both files. (2) Reviews range across 3,417 unique days. (3) January 5, 2015 has the most number of reviews in the dataset. (4) The average rating for restaurants in this datatset is slightly over 4 stars. Clearly, the reviews in this corpus express predominantly positive sentiment. (5) The average value for the "label" column (-1 for fake reviews and 1 for non-fake reviews) is 0.794542. This tells us that the dataset is skewed and that there are many more truthful reviews than fake ones. Also, the 25th, 50th, and 75th percentiles for the “labels” column show a value of "1" providing more clues that the original dataset is highly skewed. (6) There are many more real reviewers than spammers in the data—the final clue that indicates class imbalance in the original ODDS data.

	userID	b	rating	label	date
count	359052.00	359052.00	359052.00	359052.00	359052
unique	NaN	NaN	NaN	NaN	3417
top	NaN	NaN	NaN	NaN	2015-01-05
freq	NaN	NaN	NaN	NaN	454
mean	53992.205533	459.929601	4.025871	0.794542	NaN
std	45806.707721	259.923732	1.055061	0.607210	NaN
min	923.000000	0.000000	1.000000	-1.000000	NaN
25%	13840.000000	247.000000	4.000000	1.000000	NaN
50%	40523.000000	468.000000	4.000000	1.000000	NaN
75%	87314.000000	672.000000	5.000000	1.000000	NaN
max	161147.000000	922.000000	5.000000	1.000000	NaN

The distribution of Yelp review ratings:
5.0 141157
4.0 135250
3.0 47646
2.0 20775
1.0 14224
Name: rating, dtype: int64

The number of reviewers (spammers are indicated by '-1' value):
1 322167
-1 36885
Name: label, dtype: int64

Figure 1. Data exploration for metadata.txt

3.2 Feature Engineering

Further exploration of the data revealed that less than 20 percent of the original dataset is made up of fake reviews. In order to avoid training on data with class imbalance we created a smaller dataset of equal-sized fake and non-fake reviews. This resulted in a total of 209,234 reviews. This data was split into 0.6 for the training set, 0.2 for the validation set, and 0.2 for the test set.

Because there was no metadata about Yelp reviewers other than to indicate if they were a spammer or a real person we chose to look at review-centric structural features, such as length of the review, average word length, and number of sentences in the review. We also examined unigram features (discussed below).

We began by cleaning the training data—removing stopwords, punctuation, empty strings, and whitespace. Next, we performed tokenization and lemmatization on the corpus and found the 100 most frequent unigrams. From the training data we calculated the term frequency of the top thirty unigram tokens in fake and non-fake reviews. Then we chose the top 10 unigrams that had the largest differences in term frequency in fake and non-fake reviews as our features.

3.3 Determining Ngrams

When choosing the ngrams we encountered a dilemma. Should the models be trained on unigrams alone or a combination unigrams, bigrams, and trigrams? We decided to set up a small experiment. First, we trained our logistic regression model using unigrams only. Then we trained it again using both unigrams and bigrams. We discovered that using unigrams alone gave us much higher accuracy (around 70 percent) compared to using both unigrams and bigrams (58 percent).

We believe that this discrepancy is due to the fact that bigrams, trigrams, and larger ngrams create an increasingly sparse data set. When data is sparse a learning algorithm has an insufficient amount of data to learn from. Because the most frequent unigrams appear far more often than the most frequent bigrams there are many more training data examples of unigrams to learn from, which yields greater accuracy. Additionally, logistic regression requires large sample sizes to perform well, so using bigrams instead of the more numerous unigrams may have also contributed to a decrease in accuracy.

3.4 Normalizing the Data

After generating all the features above, we found that the average value for the "length of review" feature is much larger than values for other features. So we decided to normalize the data using a min-max normalization approach. In the scikit-learn library this is performed by the MinMaxScaler class.

$$v' = \frac{v - minA}{maxA - minA}(new\_maxA - new\_minA) + new\_minA$$

4. **Logistic Regression**

We decided to experiment with several different learning techniques—logistic regression, neural network, latent dirichlet allocation (LDA), and k-nearest neighbors. An overview of the mathematics of each is presented below followed by its implementation in our project.

4. 1 Definition of the Logistic Function

Logistic Regression is a type of regression that returns the probability of occurrence of an event by fitting the data to a mathematical function called a *logit function*. It is a member of a class of models called generalized linear models. Unlike linear regression, logistic regression can directly predict probabilities—specifically, values that are restricted to the (0,1) interval. The coefficients of the model can provide some hint of the relative importance of each input variable.

It is necessary to begin first with an explanation of the standard logistic function. The logistic function is a sigmoid function, which takes any real input t, $(t \in R)$, and outputs a value between zero and one; for the logit, this is interpreted as taking input log-odds and having output probability. The logistic function $\sigma(t)$ is defined as follows:

$$\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}}$$

A graph of the logistic function on the t-interval (−6, 6) is shown in Figure 2 below.

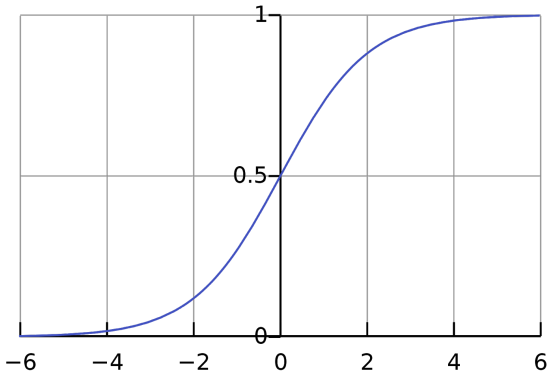


Figure 2. The standard logistic function $\sigma(t)$, where $\sigma(t) \in (0, 1)$ for all t.

Assume that t is a linear function of a single explanatory variable x. We can then express t as follows:

$$t = \beta_0 + \beta_1 x$$

The logistic function can now be written as:

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

Note that $p(x)$ is interpreted as the probability of the dependent variable equaling a "success" or "case" rather than a failure or non-case.

We can now define the inverse of the logistic function, g, the logit (log odds):

$$g(p(x)) = \operatorname{logit} p(x) = \ln\left(\frac{p(x)}{1 - p(x)}\right) = \beta_0 + \beta_1 x,$$

After exponentiating both sides:

$$\frac{p(x)}{1 - p(x)} = e^{\beta_0 + \beta_1 x}$$

4.2 Model Implementation

To implement the logistic regression learning algorithm we chose the sm.Logit module from the statsmodels package. The features chosen include the length of review, the review rating, and unigram features. To avoid multicollinearity and ensure that independent variables remained independent of each other we deleted rating_2.0.

Based on the results if the predicted value was bigger than 0.5 we assigned it as non-fake, otherwise we labeled it as fake. The dev set was used to evaluate the model and generate the confusion matrix.
Below are the word cloud images obtained for fake and non-fake reviews.

Figure 3. Word cloud for fake reviews.

Figure 4. Word cloud for non-fake (real) reviews.

Words that are much more likely to appear in fake reviews include good, side, thing, one, go, back, recommend, food, great, love. Words more likely to appear in truthful reviews include delicious, one, also, even, restaurant, though, dish, amazing, dessert, make. We initially assumed that fake reviews would contain more vague, emotional adjectives and truthful reviews would contain specific nouns that describe a restaurant, such as dish, sauce, decor, or flavor. However,

after running the program several times we did not see a clear pattern of words in our results. It is possible that these features may reflect the sophisticated psychology of spammers, who may have developed strategies to evade simple assumptions about fake user reviews (and simple methods of filtering them). More research in the habits, behaviors, and word choices of spammers will improve learning algorithms in the future. Also, more “gold-standard” review-centric and reviewer-centric metadata would help produce better performing models. The output of our model is shown in Figure 5.

Optimization terminated successfully.
Current function value: 0.588500
Iterations 6

Logit Regression Results

Dep. Variable:	label	No. Observations:	125531
Model:	Logit	Df Residuals:	125515
Method:	MLE	Df Model:	15
Date:	Sat, 18 May 2019	Pseudo R-squ.:	0.1510
Time:	03:36:20	Log-Likelihood:	-73875.
converged:	True	LL-Null:	-87011.
		LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
length_of_review	7.7870	0.093	83.894	0.000	7.605	7.969
order	0.4303	0.168	2.563	0.010	0.101	0.760
thing	0.3251	0.142	2.282	0.022	0.046	0.604
good	-0.1892	0.088	-2.145	0.032	-0.362	-0.016
side	1.3556	0.139	9.723	0.000	1.082	1.629
day	0.8866	0.164	5.419	0.000	0.566	1.207
bit	0.9866	0.144	6.839	0.000	0.704	1.269
flavor	1.4055	0.172	8.165	0.000	1.068	1.743
pretty	5.0092	0.165	30.420	0.000	4.687	5.332
sauce	1.3159	0.167	7.888	0.000	0.989	1.643
star	1.4048	0.272	5.163	0.000	0.872	1.938
rating_3.0	1.9621	0.043	45.205	0.000	1.877	2.047
rating_4.0	1.7422	0.041	42.060	0.000	1.661	1.823
rating_5.0	1.0375	0.042	24.975	0.000	0.956	1.119
intercept	-2.4525	0.042	-58.597	0.000	-2.534	-2.370

Figure 5. Results of the logistic regression training model.

Our model had an accuracy of 70 percent. In the confusion matrix (shown in Figure 6 below) the “0” value indicates fake reviews and “1” value indicates truthful reviews. The precision of this model for identifying fake reviews is slightly lower than for identifying truthful reviews.

	precision	recall	f1-score	support
0	0.67	0.79	0.72	20925
1	0.74	0.61	0.67	20918
accuracy			0.70	41843
macro avg	0.71	0.70	0.70	41843
weighted avg	0.71	0.70	0.70	41843

Figure 6. Linear Regression confusion matrix.

5. **Neural Network**

The next learning algorithm we chose was an artificial neural network (ANN) called Multilayer Perceptron (MLP).

5.1 Definition of the Multilayer Perceptron

An artificial neural network is a computational model that is inspired by the way biological neural networks in the human brain process information. An MLP is a type of artificial neural network where connections between the nodes do not form a cycle and are different from recurrent neural networks. An MLP consists of at least three layers of nodes—an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function.
If a multilayer perceptron has a linear activation function in all neurons, that is, a linear function that maps the weighted inputs to the output of each neuron, then linear algebra shows that any number of layers can be reduced to a two-layer input-output model. In MLPs some neurons use a nonlinear activation function that was developed to model the frequency of action potentials, or firing, of biological neurons.

The two historically common activation functions are both sigmoids, a mathematical function having a characteristic "S"-shaped curve, and are described as:

$$y(v_i) = \tanh(v_i) \text{ and } y(v_i) = (1 + e^{-v_i})^{-1}$$

In recent developments of deep learning the rectifier linear unit (ReLU) is more frequently used as one of the possible ways to overcome the numerical problems related to the sigmoids.

Since MLPs are fully connected, each node in one layer connects with a certain weight w_{ij} to every node in the following layer. Learning occurs in the perceptron by changing connection weights after each piece of data is processed, based on the amount of error in the output compared to the expected result. This is an example of supervised learning, and is carried out through backpropagation, a generalization of the least mean squares algorithm in the linear perceptron.

For deceptive opinion spam detection, some preliminary work has been done using neural network models. Those studies have largely focused on convolutional neural network and recurrent neural network models [20].

5.2 Model Implementation

After researching the literature we determined that a multilayer perceptron would work well as a binary classification model for detecting fake user reviews. Based on the dataset with all the features that we have identified, we applied this model with 3 hidden layers and ReLU activation; sigmoid activation for the output layer. The tuned number of neurons in the three hidden layers are 15, 50, 20 respectively.

The accuracy of our MLP model was 71%. This is not surprising due to the fact that neural network models have great non-linear fitting capabilities. Neural network models can learn intrinsic features from raw data fully automatically without any significant human effort to carefully construct patterns. Also, neural network models with well-trained word embeddings are able to effectively capture the syntactic structures of a text or semantic relations between context words in a more scalable way.

6. **Latent Dirichlet Allocation**

For our project we also implemented the Latent Dirichlet Allocation (LDA) algorithm together with *k*-nearest neighbors for comparison. In our earlier work on the *Unsupervised Joint-Sentiment Topic* (UJST) technique we analyzed how its underlying component, LDA, actually infers the topics in a corpus. We present a summary of how LDA works below.

The LDA model of a document $d = (w1, w2, \dots, w_n)$ is as follows:

1. Model a k dimensional random vector θ as a Dirichlet distribution Dir(α). This step generates θ as parameters for the probability distribution of latent variable z_n.

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k}\alpha_i)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\prod_{i=1}^{k}\theta_i^{\alpha_i-1}$$

2. For n-th term,
(a) Model the n-th latent variable z_n as a multinomial distribution Multi(θ). z_n is a k dimensional vector with k -of-1 schema.

$$p(z_n|\theta) = \prod_{i=1}^{k}\theta_i^{z_{n,i}}$$

(b) Model w_n as a multinomial distribution Multi($z_n\beta$), where β is a $k \times V$ parameter matrix. This step is using z_n to choose one row of β as the parameter for the probability distribution of z_n. Together with the whole space of z_n, it indeed makes a mixture multinomial distribution for w_n.

$$p(w_n|z_n,\beta) = \prod_{j=1}^{V}(\sum_{i=1}^{k}z_{n,i}\beta_{i,j})^{w_{n,j}}$$

Once given the parameters α and β, the joint distribution of θ, $z = \{z1, z2, \dots zN\}$ and $d = \{w1, w2, \dots, wN\}$ is given by:

$$p(\theta, z, d|\alpha, \beta) = p(\theta|\alpha)\prod_{n=1}^{N}p(z_n|\theta)p(w_n|z_n,\beta)$$

The marginal distribution of a document d is given by:

$$p(d|\alpha,\beta) = \int_\theta \sum_z p(\theta, z, w|\alpha, \beta)$$

$$= \int_\theta p(\theta|\alpha)\Big(\prod_{n=1}^{N}\sum_{z_n}p(z_n|\theta)p(w_n|z_n,\beta)\Big)d\theta$$

$$= \frac{\Gamma(\sum_{i=1}^{k}\alpha_i)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\int_\theta\Big(\prod_{i=1}^{k}\theta_i^{\alpha_i-1}\Big)\Big(\prod_{n=1}^{N}\sum_{i=1}^{k}(\theta_i\prod_{j=1}^{V}\beta_{i,j}^{w_{n,j}})\Big)d\theta$$

$$= \frac{\Gamma(\sum_{i=1}^{k}\alpha_i)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\int_\theta\Big(\prod_{i=1}^{k}\theta_i^{\alpha_i-1}\Big)\Big(\prod_{n=1}^{N}\sum_{i=1}^{k}\prod_{j=1}^{V}(\theta_i\beta_{i,j})^{w_{n,j}}\Big)d\theta$$

For a corpus D of M documents, the data log-likelihood is given by:

$$L(D;\alpha,\beta) = \log p(D|\alpha,\beta) = \sum_{d\in D}\log p(d|\alpha,\beta)$$

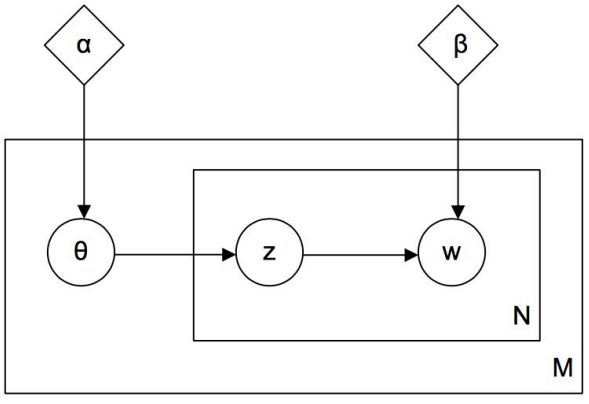


Figure 7. The graphical model of LDA

To implement LDA we used the Gensim library. We first used the preprocessed unigrams to generate a document term matrix. Next, Gensim created a unique ID for each word in the document and produced a dictionary of word IDs and their frequencies. This matrix was then passed to the LDA algorithm. We then determined the Euclidean distance for each row of the matrix and determined the *k* nearest neighbors to classify each review as fake or non-fake. The predicted and actual values are output to the console. We calculated the accuracy as:

$$\begin{aligned} accuracy &= (TP + TN)/(TP + FP + FN + TN) \\ &= 0.6061 \end{aligned}$$

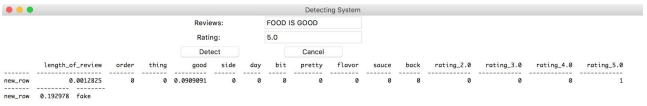
With a 60 percent accuracy rate the LDA model did not perform as well as the bag-of-words (ngrams). Initially, we considered the possibility that certain feature extraction algorithms, such as TF-IDF, may not be as effective when (a) there is a significant class imbalance in the dataset (this is unlikely to affect our results since the dataset we created contains equal parts fake and non-fake reviews) and (b) when there are words with high frequency that are very predictive of one of the classes (words that are found in most documents of that one class) [21]. However, as far as we know LDA does not exhibit these behaviors. It is unclear why the LDA and knn model did not have higher accuracy.

7. **Conclusion and Discussion**

While our models were not as accurate as expected, we gained a lot of insight into how to build machine learning models for opinion spam detection. More metadata about the reviews and reviewers (especially spammers) would improve the accuracy of our models as well as a larger amount of "gold-standard" fake Yelp reviews. We realized that sparse data, especially when it comes to models that require large amounts of example data to train, such as as logistic regression, can dramatically decrease the accuracy of our models. We also experienced first-hand how opinion spammers employ a variety of sophisticated strategies when writing fake reviews to make detection very difficult even for computer algorithms. Spammers seem to avoid words with vague positive or negative sentiment and use many neutral-sounding nouns, making detection difficult even for machine learning classifiers. Further research is needed to assess whether LDA is less likely to perform well on short documents, where there isn't much text to model. Additionally, more research is needed to determine the effects of data sparsity on LDA models when applied to fake review detection.

8. **Deployment**

For the deployment part, we decided to write a "detecting system" window using the tkinter package in Python. A user could input a review text and the review's rating respectively into the entry cells, and then hit the "Detect" button. Our review detection system will take these two inputs and run the logistic regression model. The features being used will be listed in the first row while the predict value and predict label will show up in the second row. An example screenshot is shown below:



9. **Acknowledgements**

We would like to thank researchers Shebuti Rayana and Leman Akoglu of the State University of New York at Stony Brook for providing the Yelp dataset through the ODDS library.

References

[1] Luca, M. (2016). Reviews, reputation, and revenue: The case of Yelp.com. Boston: Harvard Business School.

[2] Streitfeld, D. (2012, September 04). His Biggest Fan Was Himself. Retrieved from https://bits.blogs.nytimes.com/2012/09/04/his-biggest-fan-was-himself/

[3] Lau RY, Liao SY, Kwok RCW, Xu K, Xia Y, Li Y (2011) Text mining and probabilistic language modeling for online review spam detecting. ACM Trans Manage Inf Syst 2(4):1–30

[4] Streitfeld, D. (2012, October 18). Buy Reviews on Yelp, Get Black Mark. Retrieved from https://www.nytimes.com/2012/10/18/technology/yelp-tries-to-halt-deceptive-reviews.html

[5] Lim, E., Nguyen, V., Jindal, N., Liu, B., & Lauw, H. W. (2010). Detecting product review spammers using rating behaviors. Proceedings of the 19th ACM International Conference on Information and Knowledge Management - CIKM 10. doi:10.1145/1871437.1871557

[6] Mukherjee, A., Liu, B., & Glance, N. (2012). Spotting fake reviewer groups in consumer reviews. *Proceedings of the 21st International Conference on World Wide Web - WWW 12*. doi:10.1145/2187836.2187863

[7] Lin, Y., Wang, X., & Zhou, A. (2016). Opinion Analysis for Online Reviews. *East China Normal University Scientific Reports,4*, 79-94. doi:10.1142/9789813100459_0007

[8] Li, F. H., Huang, M., Yang, Y., & Zhu, X. (2011). Learning to Identify Review Spam. In *Special Track on Integrated and Embedded AI*. Barcelona, Spain: Twenty-Second International Joint Conference on Artificial Intelligence.s

[9] Li, J., Ott, M., Cardie, C., & Hovy, E. (2014). Towards a General Rule for Identifying Deceptive Opinion Spam. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. doi:10.3115/v1/p14-1147

[10] Dixit S, Agrawal AJ (2013) Survey on review spam detection. Int J Comput Commun Technol ISSN (PRINT) 4:0975–7449

[11] Shojaee S, Murad MAA, Bin Azman A, Sharef NM, Nadali S (2013) Detecting deceptive reviews using lexical and syntactic features. In: Intelligent Systems Design and Applications (ISDA), 2013 13th International Conference on (pp. 53–58). IEEE, Serdang, Malaysia

[12] Jindal N, Liu B (2008) Opinion spam and analysis. In: Proceedings of the 2008 International Conference on Web Search and Data Mining (pp. 219–230). ACM, Stanford, CA

[13] Fei G, Mukherjee A, Liu B, Hsu M, Castellanos M, Ghosh R (2013) Exploiting Burstiness in reviews for review spammer detection. ICWSM 13:175–184

[14] Yafeng Ren and Donghong Ji. Neural networks for deceptive opinion spam detection: An empirical study. Information Sciences, 385-386:213–224, 2017.

[15] Part-of-speech tagging. (2019, April 30). Retrieved from http://en.wikipedia.org/wiki/Part-of-speech_tagging

[16] Ott M, Cardie C, Hancock JT (2013) Negative Deceptive Opinion Spam. In: HLT-NAACL., pp 497–501

[17] Chapelle O, Schölkopf B, Zien A (2006) Semi-supervised learning. Vol. 2. Cambridge: MIT press.

[18] Blum A, Mitchell T (1998) Combining labeled and unlabeled data with co-training. In: Proceedings of the eleventh annual conference on Computational learning theory (pp. 92–100). ACM, Madison, WI

[19] Liu B, Dai Y, Li X, Lee WS, Yu PS (2003) Building text classifiers using positive and unlabeled examples. In: Data Mining, 2003. ICDM 2003. Third IEEE International Conference on (pp. 179–186). Melbourne, Florida, IEEE

[20] Ren, Y., & Ji, D. (2019). Learning to Detect Deceptive Opinion Spam: A Survey. *IEEE Access,7*, 42934-42945. doi:10.1109/ACCESS.2019.2908495

[21] Idf :: A Single-Page Tutorial - Information Retrieval and Text Mining. (n.d.). Retrieved from http://www.tfidf.com/