Fake Review Detection

Rahul Verma – Info 290T, MIMS 2015, School of Information

rahul.jverma@berkeley.edu

Introduction

Fake reviews are a major problem in e-commerce today. Users of sites like Yelp, Amazon use reviews to gauge the quality of the business or product and decide whether to use their hard-earned money to support a said business or product. The level of influence that these reviews have over the success of businesses, especially small ones is undisputed. The level of rating on Yelp or Amazon can make or break a small business. My project tries to answer the question of fake review or opinion spamming detection, the effectiveness of certain features and gaps if any in my approach and understanding of the subject.

Problem

It is then understandable that some businesses try to influence their ratings on such websites by hiring professional 'opinion spammers' or by asking friends and families to post non-genuine reviews and ratings. The scale of this problem has only grown over the years. Some estimates suggest that upto 20% of the reviews on sites like Yelp or Amazon can be fake! If the proportion of such reviews increases further, it would undermine customer confidence in sites like Amazon and Yelp and deprive consumers of an otherwise robust method conducting e-commerce. It is imperative that such practices be discouraged.

Motivation

So here we have a primarily qualitative problem, deeply intertwined with human nature and on the other end we have these powerful, scalable mathematical techniques. My motivation in choosing this subject is primarily to pit the power of data mining techniques against a very human problem of lying and see if these advanced quantitative techniques are really capable of imputing intent of a person even if the person tries to hide the true motive.

Insights

A completely 'independent' method of identifying opinion spamming or fake reviews would be linguistic analysis. Can we extract features from just the text to figure out if the reviewer is a genuine one or not? If such a method can be devised, it would be a breakthrough in imputing intent from just a persons writing. This is currently the subject of wide research in the Natural Language Processing field. Apart from the text of the review, there is a whole host of information that websites have about the reviewer, business etc. Can these be somehow leveraged to identify non-genuine reviews? These are the questions that I dealt with in my project. My ideas regarding these features have also been developed further by referring to academic papers in this discipline.

Dataset

I started out with the assumption that the reviews present in the Yelp Academic Dataset would be more than enough for me to conduct linguistic analysis. But once I started to design the code, I realized that I did not have any neatly labeled data that would help me train the

classifier! It was a huge handicap. I searched online for fake review dataset and come across a Amazon Mechanical Turks (AMT) generated fake dataset on http://myleott.com/. After registering and downloading data, I noticed and also read in research papers that AMT data is not a good approximation of real-life fake reviews. The best bet for getting fake reviews was from Yelp. I dropped and email to Yelp requesting for fake review dataset but due to business and confidentiality reasons, that data was not publicly available. That is when I decided to scrape the Yelp website for getting genuine and non-genuine records. The attributes extracted are –

ReviewDataset attributes–

ReviewerName, ReviewerURL, ReviewLocation, ReviewDate, ReviewRating, ReviewText, FriendCount, ReviewCount, BusinessName, BusinessURL, Fake/Genuine Flag, BusinessAverageRating.

UsersDataset attributes–

ReviewerName, ReviewerURL, ReviewLocation, ReviewDate, ReviewRating, ReviewText, FriendCount, ReviewCount, BusinessName, BusinessURL, Fake/Genuine Flag, BusinessAverageRating.

Solution

Feature engineering took some amount of effort as initially I started out with lesser number of features. The classifier was not able to use the features that I had initially selected to accurately

differentiate between fake and genuine reviews. The final list of features used by me can be divided into two parts –

Linguistic Features

a) Unigrams

I have used nltk's POS tagger to identify the most likely category for a unigram. The parts of speech used as a measure by are number of personal nouns (John, Mary) which indicate specific information as opposed to common nouns such as wife or car. Apart from these, I have used a count of number of adjectives as a probably indicator of the intention of the reviewer.

b) Bigrams

Bigrams are a combination of two words. I have used nltk's bigram tagger and looked specifically for a combination of adverb + adjective/adverb. This combination is valid for phases such as 'exceptionally good' and 'truly remarkable'. These phrases indicate extra emphasis by the reviewer and could be a measure of what the reviewer is trying to convey.

c) The number of exclamation marks could be an indication of extra effort by the reviewer to convey an extreme emotion.

Behavioral features

a) The length of the text

Fake reviewers are paid to increase/decrease business ratings. Small review and an extreme rating might be a significant indicator.

b) Deviation from other reviewers

A large deviation from average rating (zscore) might indicate an attempt to influence the ratings.

c) Number of friends and reviews

Large number of friends and reviews indicates more active participation and lesser chance of the reviewer being fake.

d) User Activity

Lot of reviews in short bursts and then long periods of inactivity could indicate a non-genuine reviewer. Measure used is 'Reviews per day' which is calculated by *Number of reviews / (Number of days since first review).* This measure is really low for inactive accounts and large for active accounts.

Technical Details

The majority of my time was dedicated to actually obtaining the data. As I was scrapping the website to get the data, I decided to rely on Selenium python package and the corresponding Chrome Driver to get the data. My web scrapping application would get the Recommended and Not Recommended reviews for a given business, and then search of the user profile page of all the reviewers that had provided reviews for that particular business.

Nltk's POS tagging and unigram/bigram identifier along with SciKit Learns SVM classifier were used by me for linguistic feature generation and classification.

Challenges faced

- Web scrapping dynamically generated websites meant that I could not use Pythons Beautifulsoup to scrap the data. Knowing which element to grab and find an appropriate method of grabbing it in the form of X-path or CSS/Class selection meant a lot of effort and care was involved in selecting unique way identification of an element in the DOM. Dynamically generated web content meant that many times, X-path or CSS selector would not work as the structure of the DOM would keep on changing.
- Connection rate related server disconnection. Like any major website, Yelp has measures in place to prevent DDOS attacks. My initial attempts are a regular/fast paced data scrapping meant that server disconnection would take place. I had to add a buffer of random delay in excess of 7 seconds between each connection so that I could scrape for 7-8 hours at a stretch. This reduced the amount of data that was available to me.
- Creating a modular web scrapping code in order to maximize code reusability and reduce code size and troubleshooting time.
- Some features such as 'Activity window' aggregate User records. Although a substantial number of records were grabbed from user profile page, there are some users that have in excess of 1500 reviews! This meant even though a lot of time was spent grabbing their data, in the end, the aggregation meant only few records would be available for them. Thus the size of dataset was not commensurate with the time spent scrapping the records for some users with very high review count.
- For NLP feature generation, the nltk documentation is not as clear as SciKit Learns SVM classifier. As a result, it took a lot of trial and error to understand the processes involved

like, downloading of English corpus, POS tagging and token creation, passing tokens to unigram/bigram functions and find the count.

Parameter selection

Kernel - Linear

SVM's polynomial kernel requires quadratic time for any increase in the input records. As a result, I was not able to use the poly kernel although I believe it would have given me better accuracy.

Class weight

Due to huge difference in the number of Recommended reviews and Not Recommended Reviews, after the initial run, the classifier classified all the records as Recommended reviews. This was because the Recommended reviews were present in much larger numbers than Not Recommended reviews. I have use 'Auto' parameter as the class weight which gives the weight as the inverse of class frequency. This parameter did a much better job of identifying probable fake reviews in the test data.

Cross – Validation

Due to size of the training and testing data available, I have used cross-validation in order to train and test the classifier.

C value

A very large value would lead to over fitting the sample data, especially in my case where the size of training data is relatively smaller. In order to have good generalization, the C value has been kept small.

Conclusions

The scores for individual classifiers are as follows

Behavioral - Precision, Recall and Fscore
(0.92452830188679247, 0.71014492753623193, 0.80327868852459017)

Linguistic - Precision, Recall and Fscore
(0.75, 1.0, 0.8571428571428571)

Behavioral + Linguistic - Precision, Recall and Fscore
(0.92592592592592593, 0.72463768115942029, 0.81300813008130079)

Inference

As can be observed, the nlp techniques implemented by me were not strong enough to differentiate between the fake and non-fake reviews, as can be noticed from the recall value of 1. The fscore is deceptive and should not be considered while considering the efficiency.

On the other hand, behavioral features did a better job of identifying potentially non-genuine reviews as defined by Yelp. Hence behavioral feature, in my opinion, should be leveraged in order to further increase the accuracy of this exercise.

The combination of behavioral and linguistic features did boost the fscore further and hence improved the overall accuracy. Hence, a combination of behavioral and linguistic features is the best strategy to identify opinion spamming.

Related work

Resources used

1. Selenium python package and chrome webdriver.
2. nltk
3. SciKit Learn's SVM classifier
4. Python Pandas and Numpy
5. Yelp Phoenix area business reviews and user profiles

Resources referred to

1. Detecting Group Review Spam - Arjun Mukherjee, Bing Liu, Junhui Wang, Natalie Glance, Nitin Jindal, Dept. of Computer Science, University of Illinois at Chicago, Google Inc .
2. What Yelp Fake Review Filter Might Be Doing? - Arjun Mukherjee, University of Illinois at Chicago, Vivek Venkataraman, University of Illinois at Chicago, Bing Liu, University of Illinois at Chicago, Natalie Glance, Google.
3. Using PU-Learning to Detect Deceptive Opinion Spam - Donato Hernandez Fusilier, Rafael Guzman Cabrera, Manuel Montes-y-Gomez, Paolo Rosso.
4. http://www.wikihow.com/Spot-a-Fake-Review-on-Amazon.Com

Please note –

The webscrapping program requires Yelp username and Password for execution.

Future work

1. There are probably other features like IP addresses, server logs that are being used to filter out 'suspicious' reviews by Yelp. Would like to train classifier with these features.
2. There are a few more behavioral features that I can probably derive from the current user profile data that I have, like the distribution of ratings for a reviewer. Extreme ratings would indicate a user is mostly trying to influence the ratings of all the businesses that he reviews which can be used as a feature.
3. The NLP techniques used by are still quite simple as compared to what techniques a seasoned researcher would use. I would like to experiment with more advanced NLP techniques to strengthen the precision of linguistic features. I will be taking the NLP course at the iSchool in Fall 2014 and hope to implement some of the techniques learnt to this problem.
4. The current feature set tries to re-engineer the Yelp filtering system as the gold standard data in this case was generated by Yelp. I would like to also work on a more source-independent version of the classifier which would require higher proportion of linguistic techniques than my current classifier.