YELP REVIEWS: A STUDY

Aishwarya Soni, Praveen Subramaniyam, Shivika Sodhi, Prathiba Nagarajan

CS-223: Bioinformatics

Dr. Len Wesley

May 16, 2016

## Yelp Reviews- A Study

# ABSTRACT

Yelp is a local business directory service and review site with social networking features. It allows users to give ratings and review businesses. The review is usually short text consisting of few lines with about hundred words. Often, a review describes various dimensions about a business and the experience of user with respect to those dimensions. In this project, we build a classifier that classifies restaurant business reviews into multiple dimensions. We manually inspected a few hundred reviews for restaurant businesses and found \_\_ important dimensions and these include "\_\_", "\_\_" and "\_\_". We experimented with a testing dataset that consists of some reviews and labels as "positive" or "negative" and based upon the given review, the model predicted that the review is a positive or negative review. The classifier produces the best results with a precision and recall of 0.72 and 0.71 (change it if required) respectively. The training and test dataset consisted of \_\_\_ and \_\_\_ data points respectively.

# INTRODUCTION

Yelp users give ratings and write reviews about businesses and services on Yelp. These reviews and rating help other yelp users to evaluate a business or a service and make a choice. The problem most users face nowadays is the lack of time; most people are unable to read the reviews and just rely on the business’ ratings. This can be misleading. While ratings are useful to convey the overall experience, they do not convey the context that led users to that experience. For example, in case of a restaurant, the food, the ambience, the service or even the discounts offered can often influence the user ratings. This information is not conceivable from rating alone, however, it is present in the reviews that users write. Our aim is to build a classifier that can classify the businesses into the \_\_ defined categories, based on the information present in the reviews. This information when presented to the user by classifying reviews into various relevant categories can prove to be very effective in making an informed decision. Moreover, such information can also be used to rank venues based on the categories. Consider a Yelp review: "They have not the best happy hours, but the food is good, and service is even better. When it is winter we become regulars". It is not difficult to identify that this review talks about only "food" and "service" in a positive manner, and "deals/discounts" (happy hours) are not that great. Extracting this information from this review and presenting it to the user, can help the user understand why the reviewer rated the restaurant "high" or "low" and make an informed decision, without reading the review. Although, the functionality described above is desirable and useful for any kind of business, we limit the scope of our classifier to only restaurants. The classifier can be formulated as a learning problem, where the task is to build a learner. The learner can classify a given review into respective categories. A quick inspection of few hundred reviews helped us to decide important categories that are frequent in the reviews and worth extracting. We found \_\_ categories that include "\_\_", "\_\_" and "\_\_". Users often express the sentiment whether the overall experience was worth the money they paid. It is important to note that "worthiness" is different than the "Price" attribute already provided by Yelp. "Price" measures the overall expensiveness of a restaurant, whether it is "decently priced", "expensive" or "very expensive". It does not capture the sentiment or worth, which we are trying to attempt.

Table 1: Files provided by Yelp

|  |  |  |  |
| --- | --- | --- | --- |
| **File** | **Num Records** | **Description** | **Relevant Fields** |
| yelp\_academic\_dataset\_business.json | 77,446 |  | \ |
| yelp\_academic\_dataset\_checkin.json | 55,570 |  |  |
| yelp\_academic\_dataset\_review.json | 222,5214 |  |  |
| yelp\_academic\_dataset\_tip.json | 591,865 |  |  |
| yelp\_academic\_dataset\_user.json | 552,340 |  |  |
| train.csv |  |  |  |
| test.csv |  |  |  |

# RELATED WORK

# COMPUTATIONAL SOLUTION

### Overview of the entire project

The purpose of this project is to predict whether input review is positive or negative. The input data is first divided into training and testing set. The model is trained with the training set which contains the review text and their corresponding reviews. After training the model with the data set provided, the test data is given to the model to predict the review. Then the accuracy of the prediction is provided as output.

### Problem Decomposition

**High Level Problem**

The high level problem of the project is preparing the data set, creating the model, passing the data to the model and training it, then testing and finding the accuracy of the model with the known test data.

Our problem breaks down into smaller problems as follows:

**Sub Problem 1 (Preparing the Data Set)**:

Converting the input data to the format that the application understands and can process it. Dividing the data to training and testing datasets.

**Sub Problem 2 (Create the Model):**

Implementation of a model which can accept the provided dataset and predict the result. Input the known dataset to the model and train it.

**Sub Problem 3 (Test the Model):**

Input the test data set to the model, predict the values and find the accuracy of the model.

## Requirements Analysis

The yelp data set contains many json files which contains interrelated data. yelp\_academic\_dataset\_review.json contains the actual reviews and their rating to be processed. Since this contains many records of data, we picked only reviews for food category. yelp\_academic\_dataset\_business.json file contains the categories and their corresponding business id. The business id's are picked up and their corresponding reviews are taken from the yelp\_academic\_dataset\_review.json file. The yelp data set contains rating in the range of 1 to 5. The rating 1 was used as negative review and the rating 5 as positive review. A record containing the review text and their corresponding rating was created. The model takes these records as input data, splits the data into separate words and removes the stop words like the, a, an, their, they etc. Then each word's probability whether it's a positive word or negative word is calculated using the number of occurrence of the word in the positive and negative reviews. This is our training model. Now the testing data is provided to the model. The model repeats the same process i.e splits the review into words and removes the stop words. Now the probability of all the words whether it's a positive or negative review is obtained from the above model. All the positive and negative probabilities are summed up correspondingly and the review is predicted as positive is sum of positive probabilities is greater or negative otherwise.

## Implementation

We implemented a program which uses the multiple release of naive bayes algorithm to classify the review text. The review data set which we have contains a binomial ratings, i.e whether the review is positive or its negative. Naive Bayes algorithm is the simplest algorithm which is widely used for text classifications. Our program takes the input from the json files, parses the data, removes the stop words using the nltk.corpus stopwords package. Creates a list containing the sets of (review text, rating). The ;l

**Pseudo-code:**

// snippets of different .py files and some explanation of them. Also need to write and data filtering used

naiveBayesAlgoImplementation.py

nltk\_naivebayes.py

nltk\_sentiment.py

svmClassify.py

textBlobberNaiveBayes.py

Table 2 summarizes how and where we implemented each of the required Python data structures or language constructs in our project.

Table 2: Python construct requirements

|  |  |
| --- | --- |
| **Requirement** | **Implementation** |
| 1. class & inheritance |  |
| 2. class attributes & instance attributes |  |
| 3. slice |  |
| 4. lists |  |
| 5. print (using formatted strings) |  |
| 6. tuples |  |
| 7. dictionary |  |
| 8. open & close file |  |
| 9. read & write from & to file |  |
| 10. \_\_init\_\_, \_\_iter\_\_ |  |
| Extra credit for using \_\_next\_\_,\_\_str\_\_, and \_\_getitem\_\_ |  |
| 11. functions with one or more or more default argument values |  |
| 12. auto test feature of Python functions |  |

## Testing

Add testing results here

# RESULTS

Add results here

# CONCLUSION

Yelp reviews and ratings are important source of information to make informed decisions about a venue. We conjecture that further classification of yelp reviews into relevant categories can help users to make an informed decision based on their personal preferences for categories. Moreover, this aspect is especially useful when users do not have time to read many reviews to infer the popularity of venues across these categories. In this project, we demonstrated how reviews for restaurants can be classified into \_\_ relevant categories with precision and recall of 0.72 and 0.71 respectively (change of required). We compared the results obtained from our model with the NLTK sentimental analyzer and the results were quite close. After the analysis of some of the reviews, we derive that a review is treated as a positive review not only by considering the user ratings but also the context of the review.

# FUTURE WORK

Our solution to this problem could be called the word matching approach because we are essentially checking for existence of each keyword in the reviews document to determine whether it is a positive word or a negative word depending upon the probability score. Alternatively, there are various machine learning approaches such Support Vector Machine (SVM) and Maximum Entropy Classifiers that may give more accurate results. Also, Naïve Bayes algorithm is used when the features are independent of each other. If we had more time to work on this project it would be interesting to apply some machine learning techniques such as SVM for dependent features, and see whether the results can be improved. We can also check if there is a significant difference in performance when using a combination of bigrams, unigrams and trigrams instead of only unigrams.

# TEAM MEMBER RESPONSIBILITIES

All teammates contributed substantially to all aspects of the project, and all four of us worked on all parts of the code. Most of our work was done not individually, but during team meetings in which we all interacted frequently. We worked together so closely that it’s impossible to say exactly what code any one person wrote.

// individual contribution paragraph

More important though, is that all four team members contributed with different ideas that made the completion of the project successful.

# REFERENCES

[1] Hutto, C.J. and Gilbert, E.E., "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," *Eighth International Conference* *on Weblogs and Social Media (ICWSM-14),* Ann Arbor, MI, June 2014. [Online]. Available: [http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf](http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf%20)

[2] Add any paper if any

# DETAILED RUNNING INSTRUCTIONS

Copy all files in package into the same directory. All files are:

naiveBayesAlgoImplementation.py

nltk\_naivebayes.py

nltk\_sentiment.py

svmClassify.py

textBlobberNaiveBayes.py  
train.csv  
test.csv

You may need to install the NLTK library and Matplotlib library

To run in Canopy, open hd\_main.py, set the working directory to the directory where all the files are located, and click run.

The number of reviews in test set are very large, so the program typically takes 5 to 10 minutes to run.

// function of training and testing dataset in different points