ILS Z534 - Search
Yelp Dataset Challenge

Project Report

Group Members:

Amritanshu Joshi (amrijosh@indiana.edu)
Anudhriti Katanguri (anukatan@indiana.edu)
Jae Eun Kum (jaekum@indiana.edu)
Pranav Kulkarni (pnkulkar@indiana.edu)
Vishaka Brij (vbrij@indiana.edu)

Under the guidance of

Prof. Xiaozhong Liu

Indiana University Bloomington

Introduction

This project deals with dataset provided by **Yelp** - a major online business database. The dataset contains the following information:

- **61k** businesses and aggregated check-ins for each business.
- **481k** business attributes, e.g. hours, parking availability, ambience etc.
- Social Network of **366k** users for a total of **2.9 million** edges
- **500k** tips
- **1.6 million** reviews.

By using the text and numeric data of this big dataset, we conducted two different information retrieval tasks which are briefly described below.

Task 1: **Predict category** for each business from the text of the reviews and tips.

Task 2: Conduct **sentiment analysis** for the reviews of each business and **predict the reason** for the sentiment.

The detail methodology, process, result, and future works for each task is laid out and explained in this report.

Task 1

1. Research Question

In the Yelp dataset, each business is associated with one or more categories. For example, a business has **'restaurant'** and **'indian'** categories. In this task, we used text from all the reviews and tips for a particular business and built an information retrieval algorithm to predict the category for that business. We utilized the machine learning approach for training model for evaluating our system.

2. Methodology

Our proposed solution includes the following 4 steps briefly described below:

1. Gather necessary data(tips and reviews with business ID) from the Yelp dataset.
2. Get top 3 nouns with highest TF score for each category.

3. Map each business with top 3 nouns in each review and the categories in the feature vector.
4. Evaluate the system with multiple machine learning algorithms.

1) Data Collection

The Yelp dataset is provided in json format. Using MongoDB, we created a collection called ‘results’ which consisted of business_id, categories list, tips and reviews text. In this collection, all the tips, reviews and categories are stored under the corresponding business_id.

2) Data Preprocessing

i) Extract nouns from the reviews and text

From the index and collection created in MongoDB, we extracted nouns from the reviews and tips and removed other text data by using Stanford POS Tagger. Nouns were used as features since those are directly related to the category. For example, the words like ‘food’, ‘menu’, ‘waiter’ can clearly lead us to the category ‘restaurant’. On the other hand, other kinds of words like adjective, such as ‘great’, ‘terrible’, were too general for our prediction.

ii) Get top 3 nouns from each category

In this step, we computed TF for each noun and got **top 3 nouns** with the highest score from each category by implementing the **bag of words** model.

iii) Create data instances and arff file

The nouns obtained in the previous step are the features for the training set. Based on these features and categories, data instances are taken from the json file and an arff file is created. The data instances are vectors for each business. So, a data instance comprises of all the features with values 0 or 1(absent or present) and the category of that business. The file which we developed had **6000 data instances** and **200 features**.

iv) Training and evaluation

The final step is the training and testing for the algorithm. We used Weka machine language suite for comparing a number of algorithms. **Naive Bayes**, **J48 decision tree** and **ID3 decision tree** were used for comparison.

3) Analysis and Conclusion

i) Evaluation method

For the evaluation, we conducted cross validation and percentage split and compared the results to find proper machine learning algorithm for our system. On cross validation, we used 10 fold cross validation, and on percentage split, 80% of the data was set as the training set and 20% was the test set for Naive Bayes and J48 decision tree model. We set 90% as the training set and 10% as the test set for the ID3 decision tree.

ii) Results

(1) Accuracy

Here are the accuracy results from the three different algorithms and two different evaluation schemes.

Algorithm / Scheme	Cross-validation	Percentage Split
J48	76.8%	78.13%
ID3	74%	76%
Naive Bayes	27%	24%

Although the results of ID3 are slightly lower than J48, we can say that the accuracy of both algorithms are about 80% and decision tree schemes are much better fit for our system than Naive Bayes scheme. (See Figure 1)

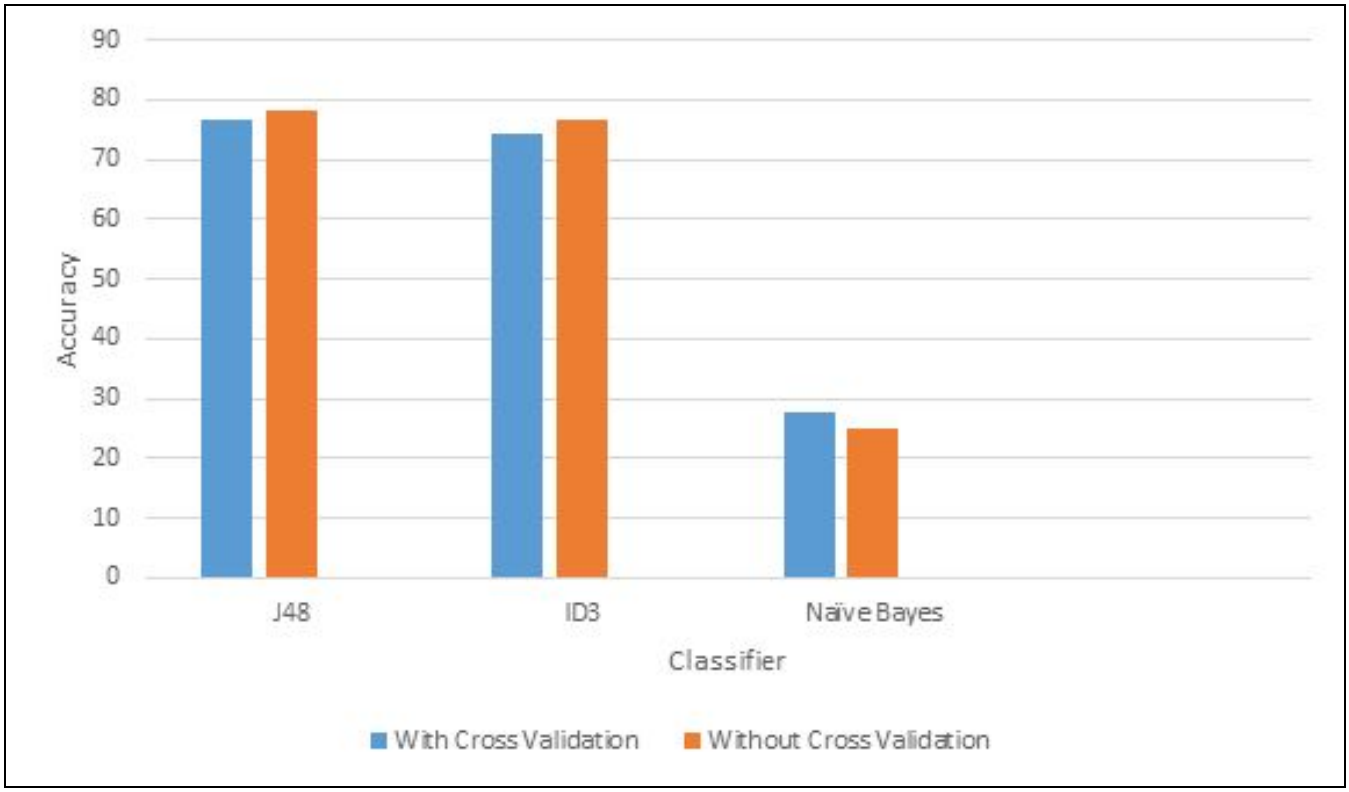

Figure 1. Accuracy

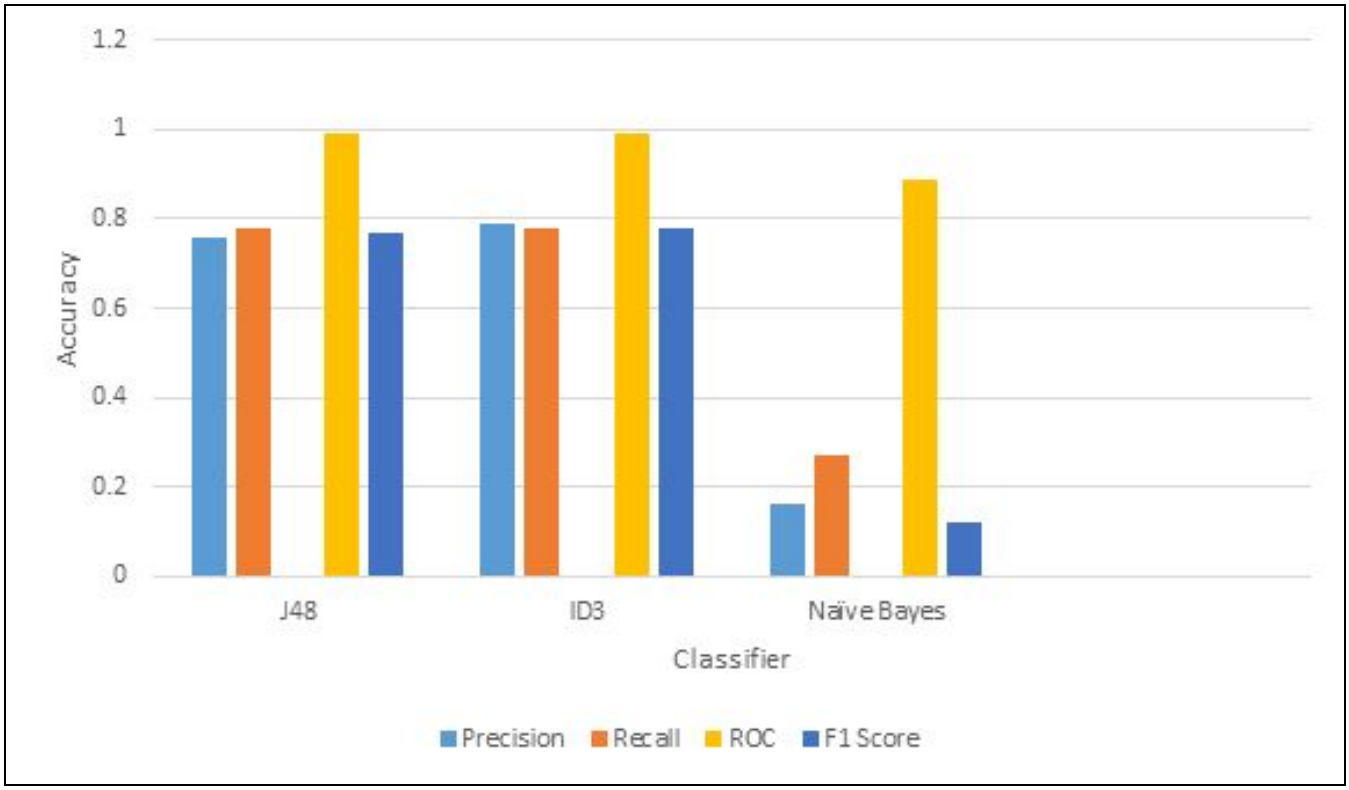

Figure 2. Precision and Recall

(2) Precision vs Recall

Here are the precision, recall, receiver operating characteristics (ROC) and f1 score result from the three different algorithms.

Algorithm/Metric	Precision	Recall	ROC	F1 Score
J48	0.75	0.78	0.99	0.76
ID3	0.79	0.78	0.99	0.78
Naive Bayes	0.16	0.27	0.90	0.14

The results of both decision tree algorithms are also similar here. The precision and f1 score from ID3 is slightly higher than those of J48. The ROC is almost perfect on the result of J48 and ID3. The results from Naive Bayes are also much worse than those from two other algorithms, so we can also conclude here that the decision tree model fits better with our system (See figure 2).

4) Future Work

a. Currently the system is predicting single category for each business. In many cases, however, each business is included in more than one category (i.e. ‘Shopping’ and ‘Drugstores’), so our system can be further expanded to predict all the categories for a business.

b. The system works only for the reviews in english. This can be expanded for different languages, such as German, French, or Spanish.

c. Getting exclusive features can result in better performance. For example, we got top 3 words from each category but not quite sure those are definitely not included in other categories, so we can give a word for each category that is clearly distinctive from other categories.

Task 2

1. Research Question

In this task we are performing sentiment analysis over a business review and predicting the causes for the sentiment. We have considered the restaurant reviews for the sentiment analysis. This task will be useful to the business owners to find the reasons for the customers satisfaction and dissatisfaction reasons. Thus they can improve their service levels by understanding the negative reasons and capitalize on their strengths based on their positive feedback.

Method/algorithm design

Sentiment Analysis

For Sentiment Analysis, we have used the Stanford CoreNLP which consists of a suite of Core NLP tools. It provides the base form of the words,normalize date, times and numeric quantities. It also mark up the structure of sentences in terms of phrases and dependencies and indicate which noun phrases refer to the sentiment, indicate the sentiment.Hence, the Stanford CoreNLP provides the functional building blocks for higher level and domain-specific text understanding applications.This analysis would be give a sentiment and score after parsing a sentence.Thus the sentiment and score can be summarized as below.

Sentiment	Score
Very Negative	0
Negative	1
Neutral	2
Positive	3
Very Positive	4

We have considered the reviews of restaurant and then for a particular business-id we split the reviews into sentences and applied the sentiment analysis and finally we

collected those sentences which contributed to the “very negative” score and the same process was followed for “very positive” score.

Reasons for Sentiment

In the review text, the nouns are usually the words directly associated with the business. The adjectives are the words which then describe those words qualitatively. So in order to find out the reason of the positive sentiment or negative sentiment, we needed to extract the adjectives and nouns from the review text and then pair them up.

Ex: Good Food can be a reason for a positive sentiment where ‘good’ is an adjective and ‘food’ is a noun.

Ex: Bad Service can be a reason for a negative sentiment where ‘bad’ is an adjective and ‘service’ is a noun.

To do this, we used 2 methods.

- **With Bigrams**
- **Without Bigrams**

A bigram is every sequence of two adjacent elements in a string of tokens, which are usually letters, syllables, or words. For this task, out tokens were words.

In the first method, we parsed the entire chunk of review text. The bigrams were read one by one. The pair of words was then used as an input to the Stanford Part-Of-Speech Tagger. The tagger then tagged the words with various tags such as Noun, Verb, Adjective, Noun singular or plural, Adverb, Pronoun, Determiner, etc. If the pair had the tags of any type of adjective and noun, we extracted it from the text and stored it in a separate HashMap<String, Integer> where the key was the pair of words and value was the count or occurrence of that pair. After parsing the entire chunk of text, the output HashMap contained a list of all the adjective noun pairs along with their frequency count. The HashMap was then sorted in descending order on values and the top 5 keys were returned as the final output. These top 5 keys contained the most commonly used pairs to express the positive or negative sentiment.

Let us consider a single review to illustrate the working of this method:

Example: *I LOVE pizza and Joel's didn't let me down! Clean place,* ***friendly staff,*** *quick to get your food and the* ***pizza was delicious****. Will definitely be back to try the rest of the menu!*

The pair “friendly staff” was tagged with adjective and noun and was added to our result HashMap.

The result given was “friendly staff” which is correct. However, the text “pizza was delicious” tells us that “delicious pizza” is also a part of the positive sentiment. We missed out that information.

Example: *They confuse SALT with SAUCE, they give you the **wrong orders**, the **food is very mediocre**, greasy, just like the staff's customer service.*
The pair “wrong orders” was tagged with adjective and noun and was added to our result HashMap.
The result given was “wrong orders” which is correct. However, the text “food is mediocre” tells us that “mediocre food” is also a part of the negative sentiment. We missed out that information.

In the second method, we did not use bigrams. We decided to parse one sentence at a time and try to extract all the adjective noun pairs from the sentence irrespective of their position. This was done with the hope of recovering the information which was lost due to using bigrams. To to this, we generated a list of all adjectives and nouns in a given sentence. The lists were then compared and the adjectives and nouns were matched based on the index. This generated a adjective noun pair which was added to a HashMap<String, Integer> where the key was the adjective noun pair and the value was the count. The process was repeated for all the sentences in the review and multiple such pairs were formed. Similar to the first method, the HashMap was then sorted in descending order on values and the top 5 keys were returned as the final output. These top 5 keys contained the most commonly used pairs to express the positive or negative sentiment.
Let us consider a single review to illustrate the working of this method

Example: *I LOVE pizza and Joel's didn't let me down! Clean place, **friendly staff,** quick to get your food and the **pizza was delicious**. Will definitely be back to try the rest of the menu!*
After using the POS tagger on the sentence 2 of this review,
adjective list = [friendly, quick, delicious]
noun list = [staff, food, pizza]
After mapping the adjectives to the nouns by index, the result was “friendly staff”, “quick food”, and “delicious pizza”. Out of these, the result phrase “quick food” doesn’t make much sense. Therefore, this result contains some noise.

Evaluation

Since this task deals with reading through the reviews and extracting meaningful phrases, the correctness of a particular result is based on human perception. Therefore, there isn't one standard method to evaluate the performance. More about evaluation is mentioned in future work.

From the results, we can see that both the methods have some pros and cons. In case of extracting reasons with bigrams, we have have a higher accuracy but we are missing on some information ie. high precision low recall.

In case of extracting results without bigrams, we are able to successfully get all the reasons. However, the output is noisy.

One possible solution to this problem is using a bigger size input. Since we are keeping a frequency counter, there is a good chance of the noisy inputs occurring fewer times than the correct inputs. This will also ensure that fewer results are missed out.

Future Work

There are certain drawbacks in our current solution. We do not have any way to deal with sarcastic comments. In the second method, we do not have a method to map the nouns and adjectives and hence sentences with conjunctions can create problems. We are also unable to deal with pronouns. This solution works only for English language. Working on these areas can greatly improve the performance of our proposed solution. The evaluation for this task can be carried out using Amazon Mechanical Turk. The sentiments and reason can be put up for people to review and rate and that can help in identifying the accuracy of our results. This will help us in getting the ground truth for the evaluation.

References :

1. http://stanfordnlp.github.io/CoreNLP/
2. http://nlp.stanford.edu/software/tagger.shtml