Yelp Challenge Dataset – Gilbert Report

As part of the Data Mining II coursework, we worked on Yelp Challenge Dataset with data related to Gilbert city. The task was to preform sentiment analysis of user review using the dataset. Each user can provide a rating between 0 to 5 and a rating of less than 2 is considered as negative, neutral when between 2.5 & 3.5 and positive if its above 3.5. The analysis was carried out using python and incorporating libraries like scipy, sklearn, pandas, numpy, nltk, matplotlib etc.

Below in section 1, we will provide overview of the dataset. In section 2, we will describe the preprocessing used for our data. Section 3 will about the feature which we will use in our classification, and in section 4, we discuss the classifiers which we have used to build model. In Section 5, we discuss about the evaluation setup and results from our classifiers and in last section we discuss the conclusion from our analysis.

1 Dataset Overview

The yelp dataset [1] is a set of 5 dataset present in json format. We further extract data which belongs to Gilbert to do our analysis.
Dataset files considered for our review were as follow:

- User Review: Consists of data related review and ratings that users provided.
- User: This dataset contains information about user and summary of their activity on the yelp website. Like number of reviews, number of likes on review, average reviews that the user gave, etc.
- Business: This dataset contains information about each business outlet. Also contains summary like no. of reviews they have received, average rating, etc.

Apart from these there were other datasets and they were not considered for our analysis.

In user review dataset, there are a total of 71015 reviews for different businesses in Gilbert city over business segments like restaurants, hotels, fitness studios etc. Out of those review, the average rating is 3.78 and varied a lot with a standard deviation of 1.5. Also, the out of all reviews more than 2/3rd of the reviews are positive, around 1/4th are negative and rest are neutral reviews.

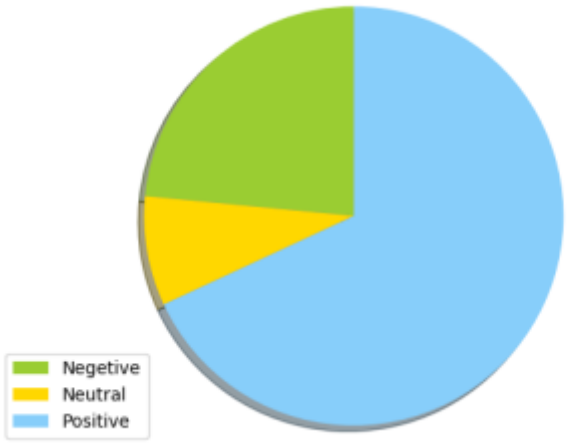

Fig.1 Distribution of positive, negative and neutral reviews on user review dataset.

Also, while analyzing other datasets we could find that the business outlets received an average of 3.58 rating and had a standard deviation of 0.83. Similarly, the users tend to give on an average 3.53 rating with 0.92 standard deviation.

2 Pre-processing

It is important to extract the necessary information for analysis. We have done the following pre-processing for preparing data to build model and predict the sentiment. Major part of our analysis was getting necessary information review text and removing information which have a very little or no impact on performance of classifier.

Following are the pre-processing measures preformed:

- Convert the json into csv file and extract only those values which are related to Gilbert city.
- Review text was processed by using following methods:
    - Convert words to lower case.
    - Remove digits
    - Remove punctuations (!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~)
    - Stop word elimination: In every language, a lot of words very frequently used and when using them to predict, it really doesnʾt improve the performance but create more challenges by adding more feature to use for prediction (curse of dimensionality).
    - Extract the total no. of words present in review text, number of positive and negative words using opinion lexicon [2].
    - Tf-idf vectorization: The processed text till now is then used to convert it into a tf-idf vector. Values of the matrix represents the frequency of the word in the document multiplied with inverse of no. of documents the word is present.
      Tf-idf matrix contains number of documents (reviews in our case) and total words.
      We had 71015 reviews and 104088 words, after application of preprocessing steps.
      In the end we got tf-idf matrix of dimensions - 71015x104088.
- Apart from the text of the review, we converted the rating into three categories:
    - negative with <=2 rating
    - neutral between 2 and 3.5
    - positive if more than 3.5

3 Feature Vector

The feature vectors are created using combination of data from user, business and user review files. The feature also consisted of the post processed data of text reviews. Following fields were used in the feature vector that were used for classification:

- User dataset: User who frequently post reviews should be more reliable and should generally, give rating around his average rating. With these assumptions, it should have a positive impact prediction model.
    - Total no reviews the user has posted
    - Avg. all review that user have posted.
- Business Dataset: A good business should generally get a good review and high no. of reviews means the outletʾs review is credible. Due to these reasons, we want use the following features from business file.
    - Total no. of reviews received by the business outlet
    - Average rating of the business.

- Review dataset: We would be using the review text to generate features and use them in our model. After initial preprocessing (removal of punctuation marks, digits, stop words), we fetch the below features:
    - Total no of words
    - No of positive words
    - No of negative words
    - Tf-idf vector which has features.
    - And sentiments from rating as our target variable

For each instance in review dataset, features were generated. Using user id and business id, present in the review, we extracted the information from user and business dataset. These features will now be used to prepare a classifier to predict the ratings.

4 Classifiers

We wanted to choose classifiers which performed well and can be enhanced for data stream classification. Also, we wanted to have a diverse set of classifiers and chose the three classifiers. For selecting parameters for the classifiers, we randomly selected 10,000 instances from our prepared dataset and then figure out the best parameters which can be used for the algorithms.
Classifiers were implemented using the 'sklearn' library of python scripting language with default parameters.

- kNN with 25 neighbors: kNN is an incremental algorithm and can be adapted to streaming dataset quite intuitively. While analyzing the no. of neighbors, after 20 the performance improvement decreased and after 25 neighbors the change in performance was very little. Hence, we decided to use 25 neighbors for the algorithm.
  Parameters used:

  ```
  KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
  metric_params=None, n_jobs=1, n_neighbors=25, p=2,
  weights='uniform')
  ```

- Logistic regression: The logistic regression can be extended to streaming analysis but not as intuitively as kNN.
  Parameters used:

  ```
  LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
  intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
  penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
  verbose=0, warm_start=False)
  ```

- Random Forest with 250 trees: Ensemble based techniques results are difficult to interpret but generally perform better than other algorithms. It uses different classifiers with different subset of features and then committee of these classifiers predict the outcome. Due to tf-idf vectorization we have a lot of features and this could help in creating lot of trees to for the ensemble. After tuning for parameter we found that after 250 trees, there was not much improvement on performance of the classifier.
  Parameters used:

  ```
  RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
  max_depth=None, max_features='auto', max_leaf_nodes=None,
  min_impurity_split=1e-07, min_samples_leaf=1,
  min_samples_split=2, min_weight_fraction_leaf=0.0,
  n_estimators=250, n_jobs=1, oob_score=False, random_state=None,
  verbose=0, warm_start=False)
  ```

5 Evaluation and Result

For evaluating the classifiers, we first applied 5-fold cross validation. In this way whole dataset in turn is used for training and testing the classifier. The result form the cross validation is then averaged for all the five folds. We have chosen the below parameters for evaluating the classifiers:

- Precision
- Recall
- Accuracy

The results from the evaluation are as follows depicting Precision and Recall values for all the ratings categories:

Metrics for Logistic Regression:
Accuracy: 0.80747729353

Review Ratings	Precision	Recall
Negative	0.74	0.70
Neutral	0.43	0.01
Positive	0.83	0.95
Total / Avg	0.77	0.81

Metrics for KNN Classifier:
Accuracy: 0.729240301345

Review Ratings	Precision	Recall
Negative	0.65	0.37
Neutral	0.23	0.00
Positive	0.74	0.94
Total / Avg	0.68	0.73

Metrics for Random Forest Classifier:
Accuracy: 0.804872210096

Review Ratings	Precision	Recall
Negative	0.74	0.71
Neutral	0.25	0.01
Positive	0.83	0.93
Total / Avg	0.76	0.80

6 Conclusion

Feature selection was an important aspect and information about user and business helped in boosting accuracy of the learning models.
It's important to handle the text as reviews tend to have lot of internet lingos which can impact the overall performance of the classifier. It would be interesting to remove words with low and very high frequency. This could help in dimensionality reduction and a better model.
Handling text will be quite challenging in streaming scenario due to unbounded source and ever changing distribution of words in the reviews.
To conclude we can observe that Logistic Regression performed well in the stated environment, but the performance of the other two classifiers are well close to each other and can be accepted.
kNN can be easily extended to stream mining and other algorithms can be also extended but not so intuitively.

7 Reference

1. Yelp Dataset: https://www.yelp.com/dataset_challenge
2. lexiconOpinion, University of Illinois at Chicago: https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#lexicon