**Method**

City-wise Sentiment Analysis over Indian Restaurants

Yelp Business and Review data is used for analyzing sentiments over Indian restaurants. We started with pre-processing of data by loading JSON files into MongoDB. We then separated all the Indian restaurant businesses into one collection called ‘IndRes’ (about 202). For all the businesses i.e unique business\_ids that appear in the collection, we extract the reviews from Review collection. This is named ‘IndRew’ (about 8230). This encompasses the reviews from all the cities (about 21) corresponding to Indian restaurant business. In further steps, we extract the reviews from this new collection for each distinct city in dataset.

We followed two approaches for sentiment analysis, Unsupervised and Supervised learning.

1. Unsupervised Learning (SentiWordNet):

SentiWordNet is a lexical resource which associates words with sentiment scores: positivity, negativity or objectivity. Java is used to implement algorithm that computes positive and negative sentiment score given a review. MongoDB client library is used with Java 1.7 to fetch the collections created in previous step. We create a map that contains distinct “city” as key and all the Indian businesses “ids” in that city as value. Using this map, we parse though the reviews for each city looking at ids which is a foreign key in review documents.

To get rid of the stop words, we jointly use Lucene Standard analyzer’s stop word set and an exhaustive list of stop words from Google. Each review is tokenized, filtered for stop words and adjectives are retained. We use adjectives (both positive and negative) of Hu and Liu’s lexicon for this purpose.

Now, we simply look up for the retained words from review bag of words in SentiWordNet giving “adjective” as the word’s part of speech (since we are dealing with adjectives). We take a count of positive and negative words along with calculating negative and positive scores city-wise.

2. Supervised Learning (Naïve Bayes):

Like, in previous step, we fetch the collections from MongoDB using its client library. We use the technique of TF-IDF over all the reviews to generate Global feature vector and city-wise reviews for local feature vectors.

All the reviews will be parsed twice for calculating term frequency and inverse document frequency. We begin by removing stop words using Lucene Standard analyzer’s stop word set and an exhaustive list of stop words from Google. We pick top 100, 200 and 500 words for global feature vector and compare performances. For local feature vector for each city, we stick to top 100 as there are not very many reviews.

Experiment:

* Weka 3.6.11 tool is used to train and test data using Naïve Bayes
* For task-2, unsupervised learning, we calculate the percentage (Pos/Pos+Neg) and compare it with average restaurant rating for that city. Observing the trend of two, we can infer if SentiWordNet is a good means to Sentiment analysis.
* For task-2, supervised learning, ground truth for a review is positive if it has more than 3star rating, else negative.
* Comparing accuracies for Global feature vector over 100, 200 and 500 reviews.
* City-wise reviews are mapped onto global and local feature vector and accuracies of both are compared.
* More emphasis is on the accuracy as we are dealing with predicting sentiment (which is either positive or negative).

**Results:**

Unsupervised Learning (SentiWordNet):

|  |  |  |  |
| --- | --- | --- | --- |
| **City** | **Number of reviews tested** | **Sentiment score scaled to 5**  **(Pos/Pos+Neg)** | **Average Restaurant reviews**  **Rating** |
| Madison | 472 | 3.68 | 3.85 |
| Glendale | 161 | 3.45 | 3.25 |
| Phoenix | 1111 | 3.69 | 3.3 |
| Las Vegas | 2828 | 3.7 | 3.5 |
| Edinburgh | 650 | 3.67 | 3.93 |
| Chandler | 322 | 3.42 | 3.45 |
| Tempe | 1268 | 3.67 | 3.3 |
| Scottsdale | 484 | 3.57 | 3.64 |

Red: Negative trend (predicted more negative sentiment than there is)

Green: Positive trend (predicted more positive sentiment than there is)

Yellow: non-conclusive

Analysis:

* Average restaurant review ratings are closely related to average positive sentiment value calculated from those same reviews.
* If a city has large number of reviews, the positive sentiment value increases. Reason being since we consider only unigrams, the negative sentiment of “not good” becomes positive in many reviews thus making it overpowering.
* In conclusion, there was no notable trend that can be applied to large scale sentiment analysis.

Supervised Learning (Naïve Bayes):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No of features**  **(Global)** | **Accuracy** | **Precision** | **Recall** | **F-measure** |
| 100 | 81.35% | 0.814 | 0.814 | 0.814 |
| 200 | 79.62% | 0.785 | 0.81 | 0.792 |
| 500 | 76.45% | 0.765 | 0.765 | 0.764 |

Comparing performance of Naïve Bayes over different dimensions of Global feature vector

Since a feature vector of 100 gives good accuracy, we consider that size for rest of training and test.

Comparing Global and Local Naïve Bayes performances over accuracy and other measures. Here we take top 3 results.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **City** | **Type** | **Precision** | **Recall** | **F-measure** |
| Phoenix | Global | 0.884 | 0.842 | 0.862 |
|  | Local | 0.85 | 0.84 | 0.849 |
| Las Vegas | Global | 0.84 | 0.825 | 0.829 |
|  | Local | 0.858 | 0.856 | 0.855 |
| Glendale | Global | 0.856 | 0.881 | 0.868 |
|  | Local | 0.799 | 0.806 | 0.802 |

Analysis:

* A feature vector of size 100 is good for about 8000 reviews.
* Local feature vector gives better accuracy than global feature vector for review of that particular city because

1. Local TF-IDF captures all the features that are more informative regarding reviews of that city.
2. Global TF-IDF creates a generic list of sentiment words that apply to any review but not very specific to task we are trying to accomplish.