PREDICTING STAR RATING FROM TEXT REVIEWS

Sneha Kadam[1], Neelasaranya Avudaiappan[1]
Clemson University, Clemson SC, USA.
skadam@g.clemson.edu, navudai@g.clemson.edu

ABSTRACT

The growth of various markets has given rise to innumerable options to choose from. One of the most common ways of deciding which business to choose is through reviews. In our project we propose a method of text summarization that negates the need to read full reviews to understand its contents. We simplify the process by providing star reviews that give an overall summary of how well accepted it is, and also a summarization technique to find the same which can be used to describe different aspects and performances the business or product. To illustrate this we use reviews of restaurants and predict the ratings.

1 INTRODUCTION

1.1 PROBLEM DEFINITION

Armed with the dataset from yelp.com, our task is to predict the star ratings from the review text alone. The result is a list of star ratings having values in the range 1 to 5, associated with every review text.

1.2 DESCRIPTION OF DATASET

The data set has five files containing business information, reviews, checking information, tip information and user information. The files are in json format which we parse and convert to csv for convenience. We use python json decoder to convert the data to the required format. For our purpose, we use Business information and reviews files to extract reviews only related to restaurants. Table 1 and Table 2 shows the format of data that was available in the business information and reviews respectively.

business_id	full_address	hours	city	review_count	categories	neighbourhood	attribute	type

Table1: Some features in business information file

business_id - A unique id used to associate a particular business
full_address - The address of the location of business
hours - The hours when the business is open
city - The city in which the business is present
review_count - The number of text reviews written
categories - The category the business falls into
neighbourhood - The location where it is present like downtown
attribute - Other attribute details like parking, wifi, ambience etc.
type - Describes if it business or non profit organization

votes	user_id	review_id	stars	date	text	type	business_id

Table 2: Some features in the Review file

votes - specifies the number of users who found the review useful, funny and cool
user_id - The unique id used to identify a user
review_id- The unique id used to identify a review
stars - The number of stars given, that is the star rating in the range 1 to 5
date - The date when the review was posted
text - The review text
type - The kind of text, for example review
business_id - The unique id used to identify the business

In our method, we are predicting star reviews for restaurants. So we first extract the business ids from the business file whose category is "restaurant". We then extract the reviews from the review file whose business ids are extracted. For our processing we need the review texts and the star ratings for training. From those set of reviews we try to avoid the sarcastic and funny posts since natural language processing becomes tricky on them. We do so by only choosing reviews that have more than a vote for 'useful' field in votes and not voted as funny. From a total of 200000 records, we choose 12000 records, 10000 for training and 2000 for testing.

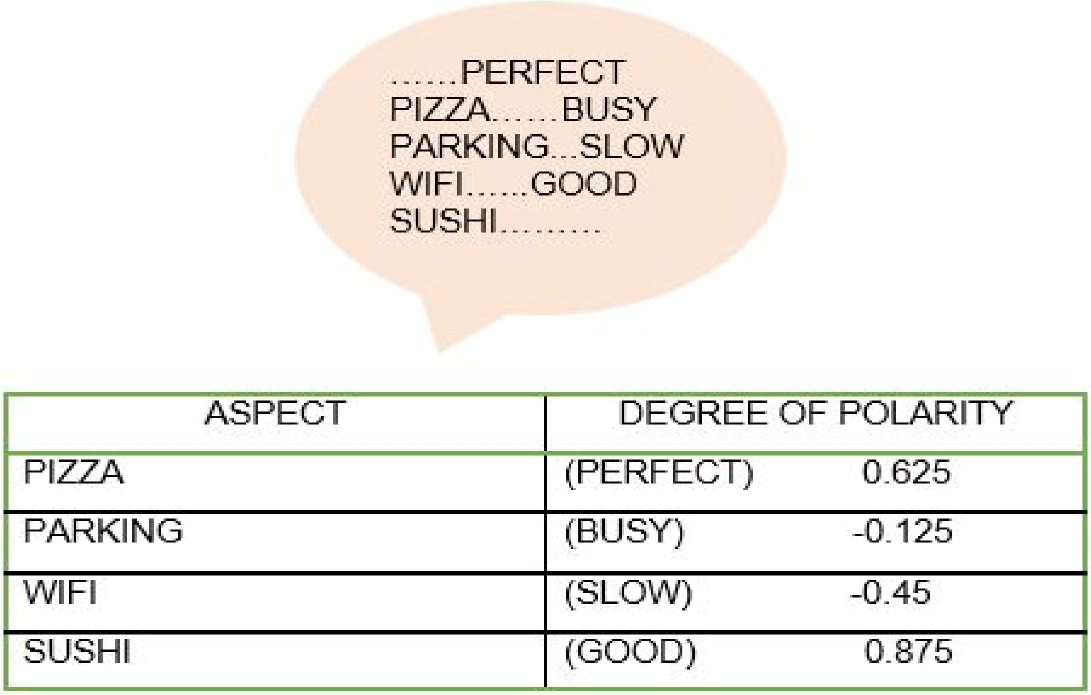
2 OUR APPROACH

2.1 ASPECT BASED SENTIMENT ANALYSIS (ABSA)

Sentiment analysis is extracting opinions, sentiments, evaluations and emotions from text. The general aim is to determine the attitude of the writer on a particular topic or the overall contextual polarity of a document[2]. In Aspect Based Sentiment Analysis the goal is to identify the aspects of given target entities and the sentiment expressed towards each aspect rather than finding the overall polarity. By assigning weights to these aspects based on sentiments we get aspect scores using which we predict the star ratings. Other than star ratings, aspect extraction can be used for a variety of applications as follows:

- Predicting what features a product is good/bad at
- What feature has improved/ deteriorated with time, if we have time series data
- What features should a product work on to get better ratings or customer satisfaction
- A metric to rank products based on different aspects

ABSA attempts to detect the main aspects (features) e.g., ‘pizza’, ‘parking’ , of an entity e.g., ‘restaurant’ and estimate the average sentiment of the aspect. For predicting the star ratings we use the ABSA approach. We Perform ABSA for feature extraction[3]. We label five features i.e. food, service, ambience, price and miscellaneous. These are the fixed categories into which every aspect will be categorized. After performing ABSA we get scores for these features based on review text. We then apply Support Vector Regression for predicting the star rating using these features.This is overall method for using ABSA in star rating prediction.


……PERFECT
PIZZA……BUSY
PARKING...SLOW
WIFI……GOOD
SUSHI………

ASPECT	DEGREE OF POLARITY
PIZZA	(PERFECT) 0.625
PARKING	(BUSY) -0.125
WIFI	(SLOW) -0.45
SUSHI	(GOOD) 0.875

Figure 1: Aspect Based Sentiment Analysis Example

2.2 ABSA STEPS

2.2.1 ASPECT TERM EXTRACTION

An aspect term names a particular aspect of the target entity. These are nothing but the nouns relevant to the predefined identity, in our case 'restaurants' are extracted. For example let us consider the following sentence "the fish was not good, but the staff greatly helped the perception" . The aspects here are 'fish', 'staff'.

All aspects are nouns but not all nouns are aspects. For instance in the sentence "My wife liked the food", wife is not an aspect, only food is. To identify nouns we used "Parts of speech" tagging. The relevant nouns were extracted after identifying the category they belonged to. That is initially all nouns are treated as aspects. Only after finding which category they belong to can we decide if it is related to restaurant. i.e whether or not it is an aspect. To do the part of speech tagging we used pos_tagger present in nltk library in python.

2.2.2 ASPECT CATEGORY DETECTION

After tagging the nouns, we next find the aspects and also categorize. In our approach, we detect the categories of the aspect terms from predefined categories : food, service, ambience, price and miscellaneous. To classify words into categories we use lin similarity. Lin similarity between two terms can be found in the wordnet module of nltk python. To categorize into a group, we it is not enough if we compare with one sentence. Similarity is measured with all words in every categories and the word goes to the category with word of maximum similarity. That is initially there are set of obvious words in each category and then the category expands as we process more data. Irrelevent aspects are not categorized which in turn leads to detecting appropriate aspects.

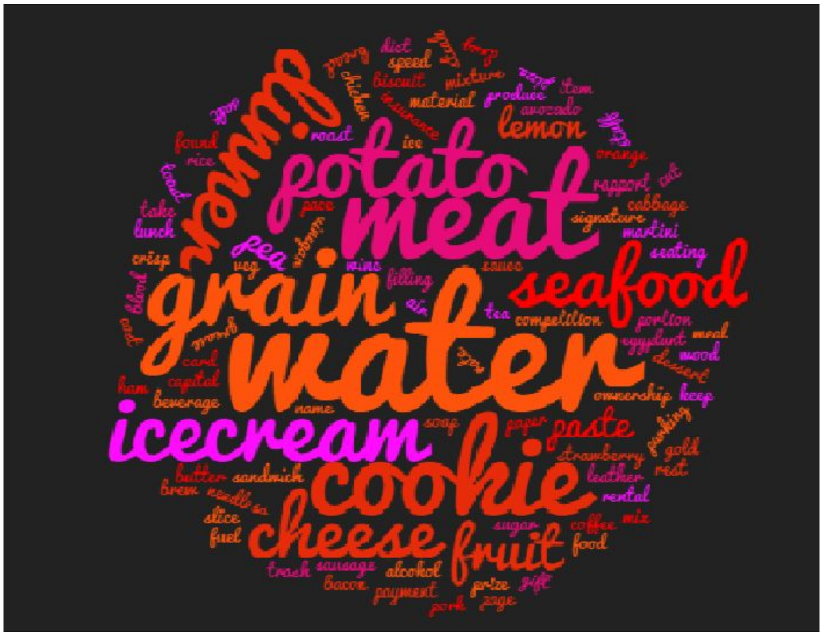


Figure 2: Feature: Food

Figure 3: Feature: Ambience

2.2.3 ASPECT SENTIMENT ANALYSIS

Now that we have the aspect term and their category, the next task is assigning scores. That is, for the aspect terms, we need to find the degree of positivity or negativity. Sentiwordnet in nltk python gives a positivity, negativity, objective score for all adjectives. We use that to find a value initially. We then detect values and scaled them to range -1 to 1 appropriately.

For example, “The manager was unprofessional” {unprofessional (manager): -0.675}

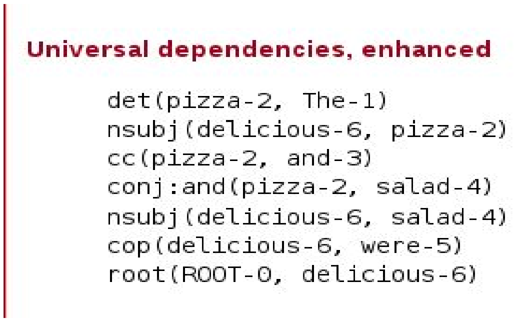
To improve upon the performance we used a list of positive and negative words in reviews by Hu and Liu, KDD-2004 to enhance performance. This is because words like “top”, “first-class” are neutral in sentiwordnet, but contained in positive list by Hu and Liu, KDD-2004. The list also contains some commonly misspelt words.

2.2.4 ASPECT CATEGORY SENTIMENT SCORES

Now that we have scores for adjectives, we need to associate scores with aspects. After the aspects are categorized and scores assigned to adjectives, the next step is to detect the aspects the adjectives describes. The most common methods used are distance metrics. That is measuring how far a noun is from the adjective and low level sentence structure analysis. Regular expressions are another set of common metrics used. While these methods produce considerable errors, they sometimes work well. For a better performance, we used Stanford CoreNLP[4].

Stanford typed dependencies representation provides a simple description of grammatical relationship by extracting textual relations.

Example : The pizza and salad were delicious



Universal dependencies, enhanced

```
det(pizza-2, The-1)
nsubj(delicious-6, pizza-2)
cc(pizza-2, and-3)
conj:and(pizza-2, salad-4)
nsubj(delicious-6, salad-4)
cop(delicious-6, were-5)
root(ROOT-0, delicious-6)
```

Figure 4: Universal Dependencies obtained using Stanford CoreNLP

2.3 PREDICTION MODELS

We Performed ABSA on 1000 records from entire training set. So we had 1000 records with 5 predictor variables Food, Category, Ambience, Price and Miscellaneous and 1 response variable i.e. star_rating. We divided it such that we used 900 of the records for training and 100 records for testing.

We have used 2 regression algorithms on our data. The first approach was Multiple Linear Regression. It takes set of features as input and finds the best-fitting straight line through the input data points. The second approach applied was Support Vector Regression. In that, the main idea is to minimize error, individualizing the hyperplane which maximizes the margin.

2.3.1 LINEAR REGRESSION

Linear regression is the most basic and commonly used predictive analysis. Regression estimates are used to describe data and to explain the relationship between one dependent variable and one or more independent variables. At the center of the regression analysis is the task of fitting a single line through a scatter plot. However linear regression analysis consists of more than just fitting a linear line through a cloud of data points. It consists of 3 stages – (1) analyzing the correlation and directionality of the data, (2) estimating the model, i.e., fitting the line, and (3) evaluating the validity and usefulness of the model.

To apply regression we performed feature scaling. Scaled values to range -1 to 1. The Linear Regression finds best fitting Linear model through data points. We implemented Linear Regression in R with the lm() function of R. After seeing the summary of the model we could see from regression coefficients that there is no strong linear relationship between features. As the p value was very high.

2.3.2 SUPPORT VECTOR REGRESSION

After Applying Linear Regression and getting a high error rate we applied a nonlinear function to the data. Support Vector Machines are very specific class of algorithms, characterized by usage of kernels, absence of local minima, sparseness of the solution and capacity control obtained by acting on the margin, or on number of support vectors. Support Vector Machine can be applied not only to classification problems but also to the case of regression. Still it contains all the main features that characterize maximum margin algorithm: a non-linear function is leaned by linear learning machine mapping into high dimensional kernel induced feature space.

The capacity of the system is controlled by parameters that do not depend on the dimensionality of feature space.In the same way as with classification approach there is motivation to seek and optimize the generalization bounds given for regression. They relied on defining the loss function that ignores errors, which are situated within the certain distance of the true value. This type of function is often called – epsilon intensive – loss function. The figure below shows an example of one-dimensional linear regression function with – epsilon intensive – band. The variables measure the cost of the errors on the training points. These are zero for all points that are inside the band.

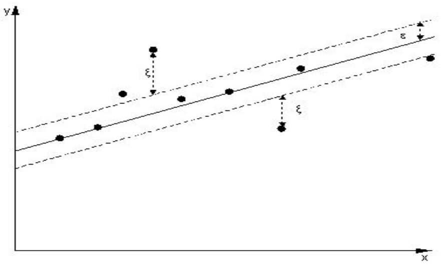


Figure 5: One-dimensional linear regression with epsilon intensive band.

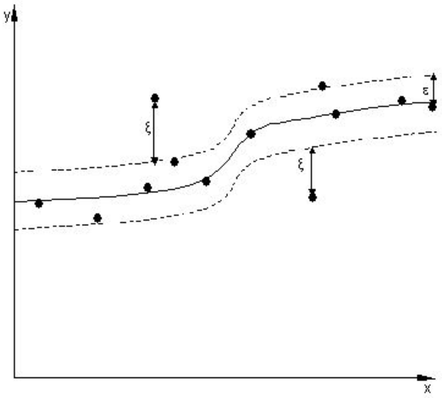


Figure 6: NonLinear Regression Function

The (**Gaussian**) **radial basis function kernel**, or **RBF kernel**, is a popular kernel function used in various kernelized learning algorithms. SVM bases its prediction on a hypothesis that “similar” points are likely to have similar values. The similarity between points mentioned here is basically the similarity determined by the kernel. The Gaussian/RBF kernel is given as:

$$K\left(x, y\right) = \exp\left(-\frac{||x-y||^2}{2\sigma^2}\right)$$

The SVM classifier with the Gaussian kernel is simply a weighted linear combination of the kernel function computed between a data point and each of the support vectors. The role of a support vector in the predicting value for a data point is tempered with α, the global prediction usefulness of the support vector, and K(x,y), the local influence of a support vector in prediction at a particular data point.

We used the e1071 package in R which has the svm function. We used the radial kernel for fitting the model. We tuned the model using 10 fold cross validation and found the best model with optimal value of cost function, gamma and epsilon.

3 RESULTS AND EVALUATIONS

We used 900 of the records for training and 100 records for testing. We Calculated Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE) for both the models.

The formulae for MAPE and RMSE:

1. RMSE:

The square root of the mean/average of the square of all of the error. The use of RMSE is very common and it makes an excellent general purpose error metric for numerical predictions. Compared to the similar Mean Absolute Error, RMSE amplifies and severely punishes large errors.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

2. MAPE:

The MAPE (Mean Absolute Percent Error) measures the size of the error in percentage terms. It is calculated as the average of the unsigned percentage error, as shown in the example below:

$$\mathbf{MAPE} = \left(\frac{1}{n}\sum\frac{|Actual - Forecast|}{|Actual|}\right)*100$$

	TRAINING ERROR	RMSE	MAPE
LINEAR REGRESSION	1.289	1.41	51
SUPPORT VECTOR REGRESSION	1.216164	1.16	40

Table 2. Comparison of Training and Test Error for Linear Regression and SVR Model

4 CONCLUSION

Our Aspect based sentiment analysis approach for star rating prediction performs better than normal sentiment analysis based approaches which do not consider the context of the sentiment. Considering the results and evaluation we have made so far, Support Vector Regressor is our best prediction model for the data set. It has managed to achieve up to 60% accuracy, compared to Linear Regression Model which had an average of about 49% accuracy. Also, the difference between training error and test error for Linear Regression is more than SVR. Thus, SVR is better fitted model than Linear Regression model.

5 REFERENCES

1. Opinion Mining, Sentiment Analysis, and Opinion Spam Detection
2. SemEval-2016 Task 5
3. Aspect-Based Sentiment Analysis of online reviews
4. Stanford CoreNLP