CMSC 122 Project: Quantifying crowd-sourced restaurant reviews published on Yelp

Group Name: print('what is our name')

Group Members: Rhedintza Audryna, Julianna Eng, Yunzhang Hu, Qing Yang Zhang

1. Project Description and Motivation:

While Yelp does not allow users to post reviews without an accompanying star rating (accurate as of Feb 2021), there is utility in being able to predict star reviews accurately.

The use case for Yelp and other crowdsource review platforms is obvious. By suggesting a star rating based on the textual review can introduce an element of objectivity in quantifying the quality of restaurants/services. It helps to anchor reviewers and allow them to be aware of what *other* reviewers would have rated given the situation.

More generally, this project could be applied to scenarios where there is no option for users to leave a 'star' or other numerical rating. For instance, many people post reviews of restaurants on platforms such as Instagram and Twitter. These reviews could be translated into star ratings and enhance the robustness of data available for users to consider before decision making.

2. Project Goals

By the end of the project, we should be able to accurately predict the star rating (out of 5) of a review using a trained model. The accuracy of the model can be measured via the Mean Squared Error between the actual and predicted star rating.

Users can interface with the final software (python file) by entering their review as input into a predetermined wrapper function. If time permits, the project will also explore regional and temporal variations in ratings and adjust for them accordingly when making ratings predictions.

3. Data sources:

We will be scraping reviews from Yelp.com using the urllib2 and BeautifulSoup libraries.

We aim to collect 10,000 reviews in total, of which 8,000 reviews will be randomly selected and allocated for training purposes. The remaining 2,000 will be used to evaluate the accuracy of the model created.

4. Project Breakdown and Milestones

	Task	**Due Date**	**Premorterm (expected difficulties)**
Scraping reviews from Yelp	Successfully requesting a Yelp url corresponding to a restaurant.	End of Week 4	Yelp detecting scraping attempts and preventing access via urllib
	Visit enough restaurants in the Chicago area to generate 10,000 reviews	End of Week 5	Navigating Yelp and moving on to next restaurant - how do we pick the next restaurant to scrape
	Parsing HTML file corresponding to restaurant and extracting out textual reviews along with rating	End of Week 5	
	Storing review in an accessible form (raw_text, rating)	By end of Week 5	
Cleaning and classifying data	Tokenization of raw_text, removing stop/common words, punctuations/symbols (Natural Language Toolkit library in Python)	End of Week 6	
	Stem and lemmatize tokens to identify root words	End of Week 6	Which library to use for this? Should we lemmatize tokens at all? Recommended vs 'do not recommend' has vastly different interpretations
	Remove 'poor quality' reviews - reviews with less than 5 tokens after cleaning	End of Week 6	
	Split remaining reviews into two groups - testing and training. Each group should have roughly the same proportion of 1-5 ratings	End of Week 6	
Training model	Training model using the Naive Bayes model and/or SVM model. If both models are used, choose the one that has better MSE value for user to interface with	End of Week 7	Which library is easy to use? Which model gives better results?
	Test with ngrams of different sizes	End of Week 7	
Designing user input	Wrapper function that takes user input and apply model to input	End of Week 7	How to make this process interactive
	Returns a rating and a confidence interval for the predicted rating	End of Week 7	What information would the user actually see in output?

II. Error Metric

We use precision and recall as the error metric.

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

where TP, FP, FN stand for the number of true positives, false positives and false negatives.

In terms of training error and test error, they are defined similarly as

$$\text{Training/ Test Error} = \frac{\sum_{i=1}^{m} 1\{y^{(i)} \neq y^{i}\}}{m}$$

where m is the total size of the training/test data, $y^{(i)}$ and y^{i} stand for the predicted sentiment and the true sentiment on the i-th example, respectively.