Predicting Yelp Restaurant Ratings from Review Text

Applying NLP and Machine Learning techniques to better understand the relationship between a restaurant's rating and its review text

Team **Yelp Explorers:** Parul Singh, Ilakya Palanisamy, Mukund Chillakanti, Abhinava Singh

parulsingh@berkeley.edu, ilakya.palanisamy@berkeley.edu, mukundc@berkeley.edu, asingh12@berkeley.edu

Motivation

- Restaurants are constantly getting feedback. Yelp is one channel, but many restaurants document verbal feedback, gather feedback from questionnaires, or even from other review sites.
- When faced with a huge amount of text, businesses need a way to objectively interpret their reviews.
- By building a way to relate text based feedback to a star rating, we can help these restaurants understand how they would rate on Yelp.

Problem Statement

- Can we guess a Yelp restaurant's rating from just the text of its reviews?
  - We want to learn the best mapping from a word vector to a business rating
  - We want to find the most essential and informative features for this classification
- Quality Metric
  - Accuracy on cross-validated data of predicting restaurant rating from review text

Data Preparation

Remove non-restaurants

Remove restaurants with no rating

Join restaurant and review datasets to get all review text for a restaurant

Add sentiment column for each business for binary classification

Featurization

Eliminate stop words

Stem and lemmatize words

Build bigram and trigram vocabulary

Generate matrix of feature vectors for each business.

2000 most frequent unigrams, 1000 most frequent bigrams, and 500 most frequent trigrams.

Learning Algorithms

- Binary Classification of Restaurant Sentiment
  - Predicting whether a business has a rating higher or lower than 3 as a baseline classification problem
- Rating Classification of Restaurant
  - Predicting 1-5 star rating of a restaurant
- Compare performance of Logistic Regression and SVM classifiers
- Use SVD and Chi-squared tests to identify most informative features and reduce feature set size

Tuning

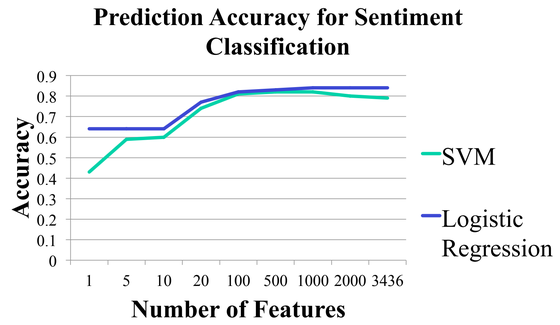
- Reduce feature size with ablation and singular value decomposition
- Find optimal C value for classifiers with 5-fold cross-validation

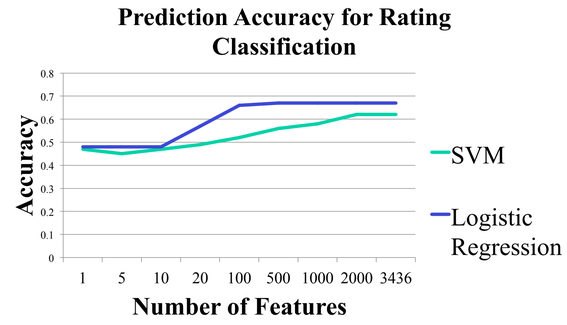
Tools Used

- NLTK SnowballStemmer, WordNetLemmatizer, Stop Words Corpus → For improving power of Ngrams Feature Model
- Scikit-learn → For SVM and Logistic Regression Classifiers as well as CountVectorizer for Ngrams
- Pickle → For serializing classifiers
- EC2 → For computing large feature matrix
- Pandas DataFrame → For data cleaning, preparation, and exploration on initial datasets

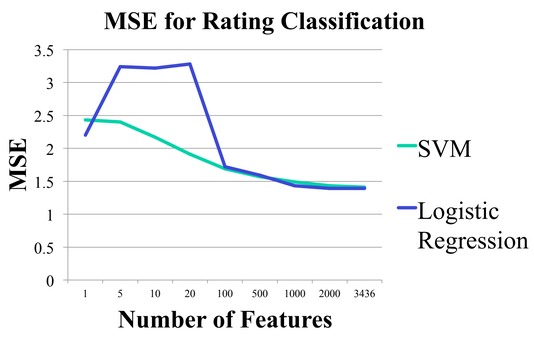
Results

These graphs show prediction accuracy as number of features increased for both classifiers we tried. Feature reduction was done using Singular Value Decomposition (SVD). Logistic regression consistently achieves better prediction accuracy than SVM.





We use MSE to quantify our error, as shown in the below graph. The best prediction accuracy for rating classification, within 0.5 of actual rating, was 84% using Logistic Regression with C = 1e5. The best RMSE was .35 using Logistic Regression with C = 1e5.



Best Prediction Accuracy

	Logistic Regression (C=1e5)	SVM (C=1)
Sentiment Classification	.86	.82
Rating Classification (within .5 of actual rating)	.84	.78

Best RMSE Accuracy

	Logistic Regression	SVM
Rating Classification (within .5 of actual rating)	.35	.46

Great	Good	Delicious	Best	Service	Amazing	One	Love	Time

Highest scoring unigram features from a chi-squared test

Great food	Go back	Love Place	Next time	Highly Recommend	Great Service	Come Back

Highest scoring bigram features

- These results were achieved after tuning our classifiers using 5-fold cross validation, to determine the best hyperparameter values, in this case the C value corresponding to regularization strength.
- As the number of features increases, we can see the MSE drops initially but stays around the same after 1000 features. Thus, using all 3500 features instead of just 1000 doesn't impact the quality of our classifier.

Lessons Learned

- We learned to how to work with a large dataset. For instance, we had to loop through subsections of reviews to compute features, constantly store data to avoid recomputation, and use EC2.
- We also learned to interpret results using an appropriate error metric like RSME. We found that just checking for exact prediction gave very low accuracy even for a good classifier.
- We learned to make use of regularization weights for final performance tuning.

Future Steps

- Exploring if Doc2Vec method for sentiment analysis improves our classification
- Build a web application to allow a restaurant to get a star rating for a given set of provided reviews
- Extract major topics existing in given reviews for a restaurant to use