Yelp Data Challenge
Final Project Report

Cole Stewart - Rafat Mahmud

Abstract — **Yelp.ca, provides an extensive dataset of business logistics, user reviews, user profiles, etc as part of an online competition, in order to incentivize innovative data-science research. Our project primarily consists of using this dataset, and performing sentiment analysis on user reviews by means of a multitude of classification, and regression algorithms, in an effort to predict the corresponding review ratings. In addition, we further attempt to extract interesting business specific trends by using clustering analysis on location specific, densely populated data points.**

I. Introduction

Sentiment is a subjective method of analyzing emotions behind human actions and language construction. While a statement may be made with a distinct sentiment in mind, it can be interpreted in many unique ways. This problem is compounded further in the absence of non-semantic cues such as body language and inflection, when analyzing statements only through text, or speech. Achieving realistic predictions for sentiment analysis problems are applicable to a multitude of sectors, from application specific integrations to corporate market analysis tools and environment specific IOT devices, to social media oriented businesses, and even artificial intelligence system platforms.

Yelp.ca, a website built to provide business reviews from a large social community, provides an extensive dataset of business logistics, reviews, user profile data, etc. from various locations across the globe, aiming to incentivize innovative data-science research, as part of a cash prize associated online competition. Our primary problem statement involves using the Yelp dataset, to initially perform sentiment analysis on user reviews, and predict corresponding review ratings. Specifically, given a review text corpus, we intend on successfully being able to predict a possible review rating in a discrete range of 1-5; 1 being the lowest, and 5 being the highest rating that a review can be associated with. We intend to achieve this by applying several classification and regression based algorithms, including Multinomial Naïve Bayes, Support Vector Machines, and several linear regression models.

In addition, we also apply Mean Shift clustering on densely populated location specific data points, in an effort to extract possibly interesting location specific business and user trends. This report will encompass the current related work on similar problems, structure of the data, methods used, and finally a discussion on the results obtained and potential future work that could be done.

II. Related Work

Most common and simplistic methods of establishing correlation between text and human emotions encompass applying the bag of words model, which is, to somewhat disregard sentence structure, and only assign significance values to influential words within the text. These significance scores can then be aggregated over vast training datasets, mapped to expected values that correspond to specific human emotions, and later be used to classify new test data based on posterior probability of the words belonging to the expected classes. Examples of algorithms that generally employ similar implementation models include Naïve Bayes Classifiers and Bayesian Networks.

According to Berger's [1] 1996 study on Natural Language processing, the Max Entropy classifier is another probabilistic method, which at times surpasses Naïve Bayes classifiers when classifying text for a variety of purposes. The Max Entropy Classifier unlike Naïve Bayes' is based on the principle of maximum entropy rather than assuming every feature of the dataset to be conditionally independent.

Apart from the probabilistic methods discussed above, there have also been several successful attempts of using linear classifiers such as Support Vectors Machine's (SVM) [2], and as well as unsupervised clustering algorithms [3] when tackling text classification and sentiment analysis. All of which indicate that there exists several diverse methods applicable and available for use to attempt to craft solutions for the specific chosen problem.

III. Data

Yelp.com, as part of the Yelp Dataset Challenge, provides the dataset that we used for this project. It contains:

- 2.7M reviews and 649K tips by 687K users for 86K businesses
- 566K business attributes, e.g., hours, parking availability, ambience.
- Social network of 687K users for a total of 4.2M social edges.
- Aggregated check-ins over time for each of the 86K businesses
- 200,000 pictures from the included businesses

Each file within the .tar archive provided contains unique json object types, which need to be exhaustively parsed to obtain the required data. However, with regards to our problem, we only intend on using the reviews dataset, and the business dataset. Several python modules are available to accommodate working with JSON data.

Examples of the data format can be seen below:

Review

```
{
  'type': 'review',
  'stars': (star rating, rounded to half-stars),
  'text': (review text),
  'date': (date, formatted like '2012-03-14'),
   ...
}
```

Business

```
{
  'type': 'business',
  'city': (city),
  'latitude': latitude,
  'longitude': longitude,
  'stars': (star rating, rounded to half-stars),
   ...
}
```

The only features we worked with in terms of the review prediction involved the actual review text, and the associated review rating. For spatial analysis of businesses, we used the business latitude, longitude, stars, and city. Other unique business specific features could definitely be used in future iterations of the project.

IV. Vectorization

Text data on its own cannot be fed directly into a classification or regression algorithm. The Bag of Words method is implemented to convert the text corpus into numerical feature vectors. However, prior to vectorization, all reviews were passed through a filter that excluded reviews less than 10 words in length. Shorter reviews generally tend to not contain as much relevant information, and be rather fickle in terms of ratings. As a result, generally contributing negatively towards prediction accuracy.

Tokenizing: Each document is tokenized into individual words by whitespace and punctuation. Each word is assigned a unique integer ID and a

mapping of seen words to their ID is stored in a vocabulary dictionary. All words are converted to lowercase to avoid creating duplicate entries. Words which appear very frequently but provide very little towards the overall sentiment of a corpus such as “a”, ”an”, ”or”, ”the”, more commonly known as “Stop Words”, are removed in an effort to reduce noise and simplify computation time.

Counting: Now that the corpus is represented as integers, each document is translated into a word count array. Each index of the array refers to a word in the generated vocabulary dictionary, and the frequency of each word within the document is stored at the corresponding index. The data that is generated through this process is rather sparse.

Normalization: Despite the removal of stop words, there could be words within the dataset that still occur very frequently. These common words are likely contribute very little to the overall sentiment of the document, and are a major source of noise that may hide less frequent but more influential words. To counteract this phenomenon, TF-IDF normalization is used. TF-IDF stands for term frequency times inverse document frequency, where term frequency is the number of times a word occurs within the given document, and inverse document frequency is the number of documents containing the given word. IDF allows for lowering the weight of more frequent terms, and increasing the weight of rare terms, by normalizing each word with the inverse in-corpus frequency. IDF is given as:

$$idf(t) = log\left(\frac{n_d + 1}{df(d,t) + 1}\right) + 1$$

where t is the term (word in this case), n_d the number of documents, and $df(d,t)$ the document frequency. One is added to the numerator and the denominator to avoid a zero divide, and an additional 1 is added to the logarithm so words that appear in every document are not completely ignored.

V. Applying Multinomial Naive Bayes (MNB)

In order to generate baseline results, scikit-learn's dummy classifier was used. This generates a random uniformly distributed set of classifications, which can then be scored against the test data. Having this data is a useful tool for comparing and quantifying actual results. The two tables in Figure 5.1 captures the performance of the random classifier.

Stars	Precision	Recall	F1-Score
1	0.12	0.20	0.15
2	0.08	0.20	0.12
3	0.12	0.20	0.15
4	0.25	0.20	0.22
5	0.42	0.20	0.27
Average	0.28	0.20	0.22

Accuracy	20%
Off By One Accuracy	49%
Off by +1 Accuracy	32%
Off by -1 Accuracy	38%
Variance	-1.034
Mean Absolute Error	1.702

Figure 5.1: Results from a random classifier

Both Bernoulli and Multinomial Naive Bayes were applied on the newly processed dataset. The range of values for star ratings for each review span between one to five stars, inclusive, incrementing by half stars. To limit the number of classes, each rating is rounded up to the nearest star. Multinomial Naive Bayes has been seen to provide better results, possibly because the frequency distribution of words such as “good” or “bad” helps determine the rating as well as whether or not they appear within

the given text, which Bernoulli Naïve Bayes ignores.

Figure 5.2 is a Naive Bayes learning curve generated using 3 fold cross validation, and shows sample accuracy for scoring the obtained results. Partial fitting was utilized to further reduce computation time. As we can see from the graph below, accuracy could improve significantly, given more training samples.

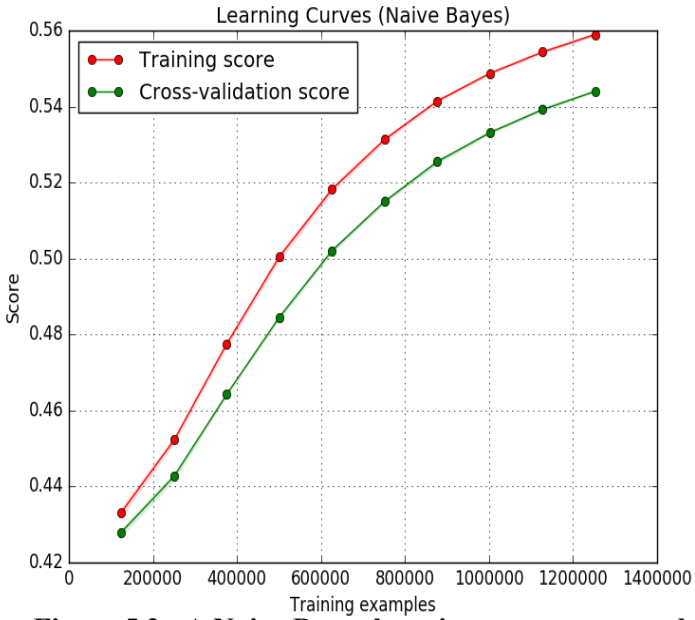

Figure 5.2: A Naive Bayes learning curve generated using 3 fold cross validation

The smoothing parameter, α, was determined using an exhaustive linear search (available from scikit-learn's GridSearchCV object) and 3 fold cross validation. All values from 0 to 1, incrementing by 0.01, were tested and scored using accuracy. The final value of α obtained with optimal accuracy, was 0.03. The exhaustive search took approximately 8 hours on 2.7 million reviews, which was rather surprising.

All classifier and regression models were trained on 70% of the data, and the remaining 30% being used for testing. After testing, the final accuracy score achieved was 58%. More metrics can be seen from the tables in Figure 5.3.

Stars	Precision	Recall	F1-Score
1	0.64	0.70	0.67
2	0.40	0.15	0.22
3	0.38	.12	0.18
4	0.43	0.52	0.47
5	0.68	0.80	0.74
Average	0.56	0.58	0.55

Accuracy	58%
Off By One Accuracy	89%
Off by +1 Accuracy	78%
Off by -1 Accuracy	68%
Variance	0.446
Mean Absolute Error	0.597

Figure 5.3: Results of applying a Naive Bayes classifier on the yelp dataset

An interesting fact to note from the results is that the classifier tends to have maximum precision predicting reviews that are rated 5 stars (68%) and 1 star (64%), and has minimum precision predicting mediocre reviews, which are rated around 3 stars (38%). Similar F1-scores for the highest and lowest rated stars also indicate similar trends for recall values as well. This is possibly the result of an even distribution of multiple words with both positive and negative connotations within the same review, as seen from the initial vocabulary dictionary.

Even though strict accuracy figures provided us with an estimate of the performance of our classifiers, we chose to define the metric "off by one accuracy", to further determine how close the classifier or regressor predictions were to actual values. In simple words, this metric tells us how many predictions were off by one, or within a magnitude difference of 1 of the actual rating, plus

the 100% accurate predictions. Similarly, the "off by +1 accuracy" metric only takes into account the predictions that the classifier overestimates by a magnitude of 1, in addition to the accurate predictions. Whereas the "off by -1 accuracy" metric takes into account the predictions that the classifier underestimates by a magnitude of 1, alongside the accurate predictions.

The off-by-one accuracy predictions for MNB seems to be very promising, with 89% of the predictions being off by one, or 100% accurate. The results of the random classifiers (20% Accuracy, 49% off-by-one accuracy) as seen from Table 5.1, are of no match to that obtained from the MNB classifier (58% Accuracy, 89% Off-by-one Accuracy).

VI. Applying Linear Regression, Perceptron, and SVM

A. Applying Linear Regression

Since review ratings can also be interpreted as continuous data, linear regression models can also be utilized for classification purposes. The results were interpreted with star ratings left as they were generated from the regressor, and further rounded to the nearest star to match the test results. As seen from Figure 6.1, an interesting point to note would be that rounding stars had very little effect on the accuracy of the results.

An ordinary linear regression model proved to be the best performing model for our particular problem. It obtained an accuracy of about 50% with an off-by-one accuracy of 93%. It had a mean absolute error value of 0.676, which is also an indication of the performance of the model. Just like the Multinomial Naïve Bayes classifier, the linear regression model also significantly outperformed the random classifier.

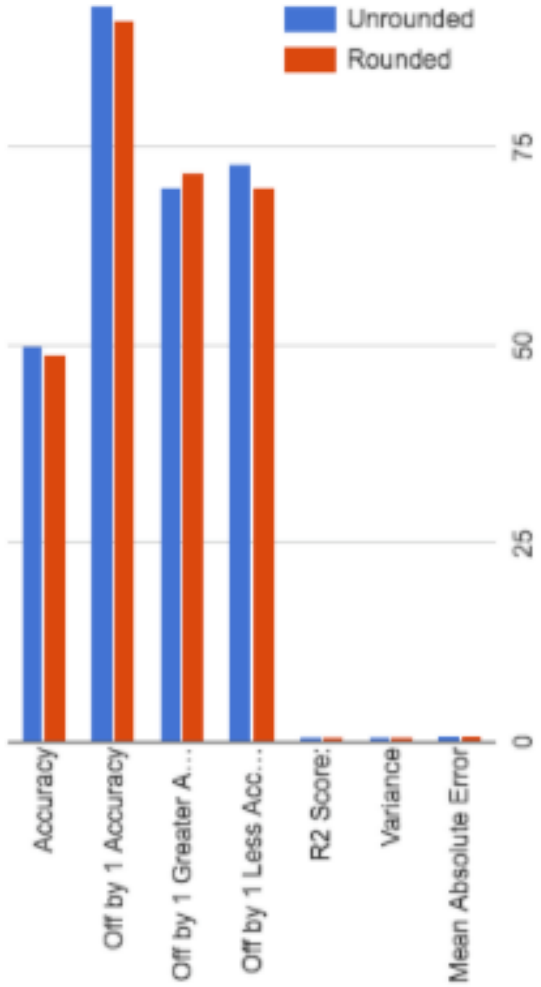

Figure 6.1: Comparison of Ordinary Least Squares Linear Regression model on rounded and unrounded stars

Accuracy	49%
Off By One Accuracy	91%
Off by + 1 Accuracy	72%
Off by - 1 Accuracy	70%
R2	0.532
Variance	0.532
Mean Absolute Error	0.676

Figure 6.2: Performance metrics of Ordinary Least Squares Linear Regression model on unrounded stars

B. Applying Perceptron and SVM

Due to the success of the linear regression model, we further tested the data by applying a Perceptron, as well a Support Vector Machine, with Stochastic Gradient Descent learning. L2 Regularization was applied on both models. And unsurprisingly, both models produced results with rather high off-by-1

accuracy values. Both models also produced similar looking precision, recall, and F1-Score graphs as seen from Figure 6.3(b) and 6.4(b). However, when compared, the SVM slightly outperformed the Perceptron.

Accuracy	47%
Off By One Accuracy	84%
Off by + 1 Accuracy	62%
Off by - 1 Accuracy	68%
Variance	0.272
Mean Absolute Error	0.779

Figure 6.3(a): Performance metrics of Perceptron

Accuracy	54%
Off By One Accuracy	83%
Off by + 1 Accuracy	78%
Off by - 1 Accuracy	59%
Variance	0.322
Mean Absolute Error	0.744

Figure 6.4(a): Precision, Recall, F1-Score comparison per star rating for SVM Model.

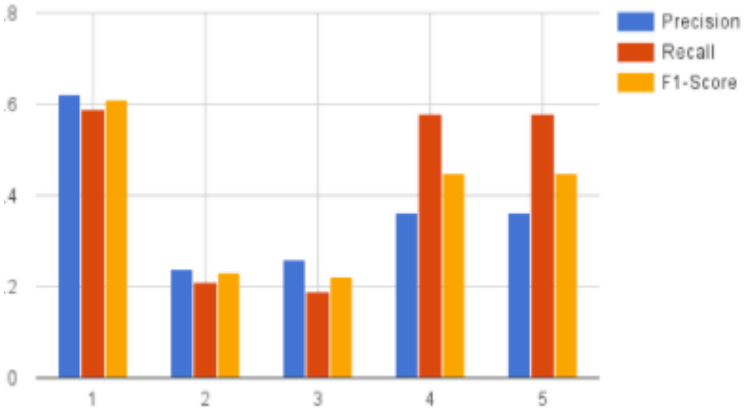

Figure 6.4(b): Precision, Recall, F1-Score comparison per star rating for Perceptron Model.

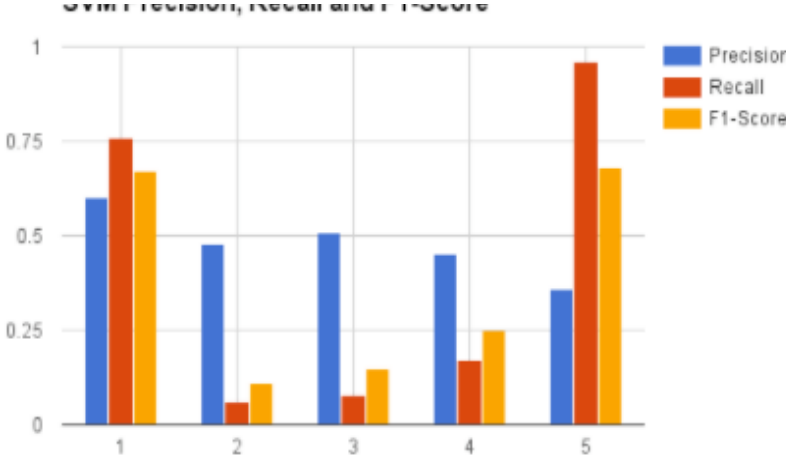

Figure 6.3(b): Precision, Recall, F1-Score comparison per star rating for SVM Model.

VII. Spatial Analysis Using Clustering

The idea for this part of the project was to choose a densely populated region with business reviews, use Mean Shift clustering to cluster businesses based on location and ratings, and possibly identify useful information such as, the part a city with the best restaurants, etc. The location information was obtained from the business json archive, and scikit-learn's mean shift clustering implementation was used.

Mean shift was chosen because it does not require a predefined number of clusters (unlike K-means) and deals well with clusters of various sizes. The initial goal was to determine areas of businesses with different rating classes, then infer possibly useful business specific information. The map below displays the results using the Google Maps API. The dots represent businesses and the color of the dot determines its cluster membership.

Figure 7.1: Mean-Shift Clustering of Businesses in Phoenix, Arizona based on user ratings

As seen from Figure 7.1, there is not much distinction between the dotted points, which represent the uniquely clustered businesses; therefore a 3D plot was attempted as seen in Figure 7.2.

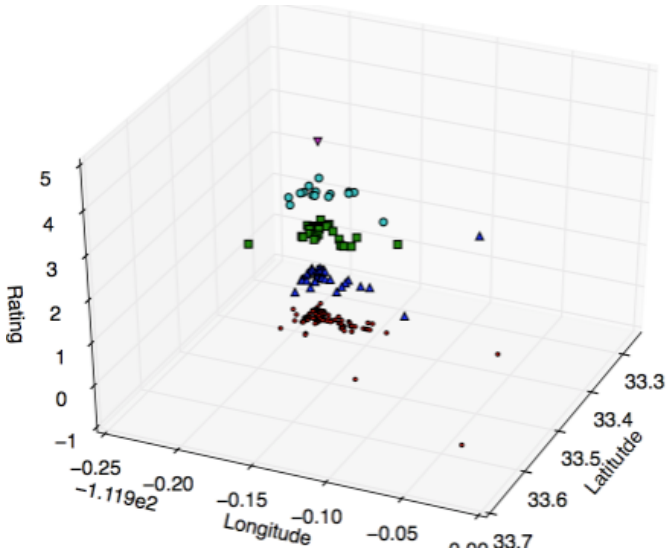

Figure 7.2: 3D plot of Mean-Shift Clustering of Businesses in Phoenix, Arizona based on user ratings

As seen from Figure 7.2, the 3D plot suggests that the clustering algorithm was successful to distinguish the different clusters, as they've been plotted and marked in the varying spatial (z) axis. But raw latitude, and longitude positions do not help in any way to infer neighborhood related business information from a 3D plot, visually. Unfortunately, due to the time constraints of the project, further analysis could not be done, but even though not much information could be extracted at the moment, there remains to be a lot to explore about this idea.

VIII. Discussion

Ultimately, Multinomial Naive Bayes proved to be the best algorithm for our chosen problem specification, with linear regression approaching asymptotically towards it. However, the linear regression model did have a higher off-by-1 accuracy. MNB resulted in a final accuracy of 58%, which is a large increase compared to the 20% accuracy of the random classifier. The mean absolute error was also reduced to 0.597 stars compared to the 1.702 stars of the random classifier. Below is a chart displaying a comparison of the results obtained from all classifiers and regression models evaluated.

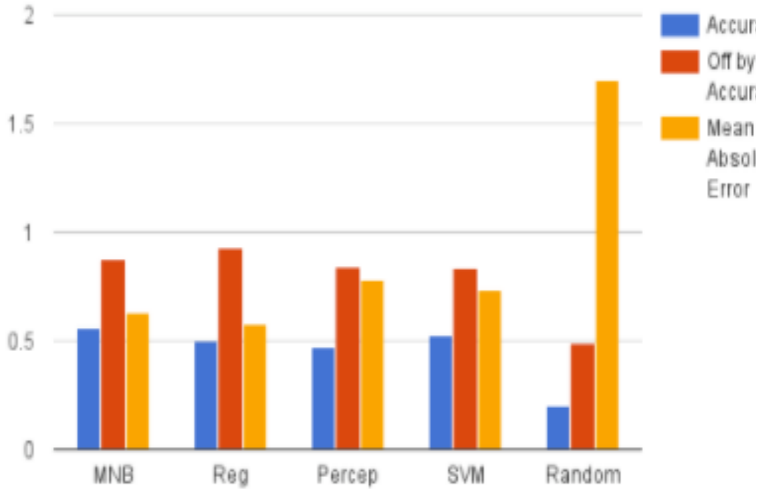

Figure 8.1: Comparison between performances of models

Except for the Random classifier used, no particular model performed too poorly on the given dataset. More training, and testing could further improve confidence on the current results.

A. Clustering Efforts

The attempts to infer business specific information from spatially clustering similarly rated businesses within a densely populated region were not particularly successful. As expected, the algorithm simply grouped together businesses of similar ratings, providing no new information. As this was not the main focus of our project, due to time constraints, not a lot of time could be spent exploring other methods. Weighting the ratings lower than the location, clustering each ratings group independently, or possibly plotting the similar clusters on a 3D map of a particular city could potentially elicit better results. However, the idea still responds to us, and further time spent analyzing the data could result in interesting outcomes.

B. Sentiment Analysis

Analyzing the raw text of the reviews, while comparing the predicted and actual star ratings provided insight into what the bag of words approach does and does not do well. Since the documents are being analyzed on a word-to-word basis, it is only the individual words that contribute towards the predicted sentiment of a specific document, and not phrases or sentences. This heuristic method works well when there are words within the text that possibly correspond to strong emotions, but results in poor classifications when more general, or confusing language is used. A recurring example of misclassifications involved descriptions of customer experiences. Each individual word forming the review often did not correlate to particularly strong emotions, even though the overall content of the review could clearly be classified by a human as either positive or negative.

The distribution of the ratings and the performance of all employed algorithms on individual star classifications are also rather interesting. Five and four star reviews were the most numerous, followed by one star reviews. This may be due to the fact that people are more inclined to review a business when they have either had a very good or an extremely poor experience, and are much less likely to write a review if their experience was mediocre. Some people may also not be objective with their ratings, and consistently provide one star reviews for a generally bad experience, or five star reviews for a generally good experience. The performance of the algorithms was also consistently higher on one and five star reviews (Figure - 6.4(b), Figure - 6.3(b), Figure 5.3). Part of this is the result of having more samples to train on, but even when the distribution of samples was seen to be uniform, predictions of one and five star reviews seemed to be more accurate. A possible hypothesis could be that one and five star reviews contain a higher frequency of sentimental words. People are likely to use stronger language when describing an “amazing” or a “horrible” experience.

C. Future Work

As mentioned earlier, the methods in this project primarily deal with individual words as n-grams. The addition of phrase and possibly sentence n-grams may further improve performance. However, this would be accompanied by a significant rise in computational cost; as this would increase the number of features almost exponentially, especially if individual words are still counted as n-grams. Another possible method to explore could be to implement a form of cumulative sentiment scoring over each individual sentence within a review. An algorithm such as Naive Bayes could be sequentially applied to each constituent sentence within the review, and the overall sentiment being determined by summing each of the individual calculated sentiment scores.

Another interesting relationship to know would be how the length of a review relates to the accuracy or error of the prediction. This could be achieved by applying something such like a Pearson correlation to the size and error of predicted reviews.

The data supplied by Yelp consists of a significant number of features that went unutilized during our analysis. Something that could be of definite use for our specific problem is the field “votes” within a review object. This field essentially represents the approval of other Yelp users of a particular review, and could potentially be used to weight the rating value of a review. A recommender system would also suit this data well. Since user IDs and business IDs are provided within the review object, similarity matrices could be easily crafted, allowing for a simple recommender to be built without much discomfort.

IX. Contributions

Cole Stewart

- Project Proposal
- Research
- Midterm Report
- Final Report
- Data Loader
- Naive Bayes Classification
- Regression
- Clustering - Data
- Clustering - 2D google maps
- Presentation Slides

Rafat Mahmud

- Project proposal
- Research
- Midterm report
- Final report
- Regression
- Perceptron
- SVM
- Neural Nets
- Predictor tool
- Custering - 3D plot
- Presentation Slides

X. References

[1] Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996, “A maximum entropy approach to natural language processing.” *Computational Linguistics*, 22(1):39–71.

[2] Thorsten Joachims, “Text Categorization with Support Vector Machines: Learning with Many Relevant Features,” *Machine Learning: ECML-98*

[3] M. Farhadloo and E. Rolland, "Multi-Class Sentiment Analysis with Clustering and Score Representation," *2013 IEEE 13th International Conference on Data Mining Workshops*, Dallas, TX, 2013, pp. 904-912.

.