6.867 Final Project

Ayesha Bajwa, Zareen Choudhury, Katy Muhlrad

Abstract

We present our methods and findings in using Yelp review data to predict the star ratings of restaurants in Arizona and Nevada. We compare occurrence and frequency featurization of textual reviews. We show that additional user and business features result in nominal improvements in test accuracy with supervised classification models, while unsupervised clustering models generally fail with standard similarity metrics. We also observe a large variation in users' rating scores by comparing the assignment of integer star ratings to positive vs. negative sentiments.

1. Introduction

We use the publicly available Yelp dataset to explore prediction of business star ratings from user reviews. The dataset consists of 4.7 million reviews of 156,000 business from 12 major US metropolitan areas, subdivided into reviews, business metadata, check-in information, tips, and photos. Our investigation focuses on a subset of the review data. Each review entry contains metadata, textual contents, and a rating between 1 and 5 stars. Our goals are to explore binary predictions of review ratings (positive or negative) and multiclass predictions of review ratings (1-5).

We also investigate whether providing additional information about the business or user (e.g. business location, average rating awarded by user) influences rating prediction. Each classifier takes the default feature matrix consisting of the binary occurrence or term frequency vectors, with the option of augmenting the matrix with additional business or user features. For example, we may want to consider that a particular user gives, on average, low-scoring reviews when using her review to predict a business's star rating. Similarly, knowing the city in which a business is located may influence our prediction of its star rating. Additionally, we would like to determine how assigning positive vs. negative binary class labels to reviews with 3-star ratings might affect performance of the binary classifiers. We also attempt unsupervised learning such as k-means and spectral clustering on review data to see whether the reviews can separate into clusters representative of their star ratings.

This paper is organized as follows: Section 2 provides an overview of our methods, Section 3 goes into further details about our dataset construction and preprocessing, Section 4 presents the results of our investigations, and Section 5 summarizes our findings.

2. Methods

2.1. Data Preprocessing

The first step is to preprocess the Yelp dataset and featurize reviews to be used by our classifiers and clustering methods. Preprocessing involves the following:

- Downsampling from the review corpus of the *Yelp Dataset Challenge* [1]
- Converting the provided JSON files into CSV files
- Filtering the reviews in the CSV files based on attributes we want to investigate (e.g. restaurants in Arizona)
- Splitting the filtered reviews into training, validation, and test sets (60/20/20 split)

We next featurize the reviews into binary occurrence or term frequency representations by:

- Cleaning up reviews through lowercasing, stemming, and removing punctuation and stopwords, using the `nltk` toolkit (Loper & Bird, 2002)
- Creating a dictionary of words from the training set
- Using the created dictionary to featurize training, validation, and test data as either binary occurrence vectors or term frequency vectors with

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

[1] https://www.yelp.com/dataset/challenge

TFIDF regularization (Sparck Jones, 1988), (Pedregosa et al., 2011)

- Optionally augmenting feature vector with additional information about the *business* or *user* of a particular review. To use additional features, we concatenate these feature vectors to the original feature vector.
  - Features such as city use one-hot vectors
  - Numerical features such as the user's average star rating are represented as floating point values
- Assigning binary and multiclass labels to reviews
  - Multiclass labels are equivalent to the reviews' star ratings (1-5)
  - Binary labels for reviews with star ratings of 1-2 are negative, 4-5 are positive
  - We investigate assignment of star rating 3 to be positive or negative

2.2. Binary Rating Prediction

For binary rating prediction, we investigate three classification techniques: perceptron, binary logistic regression, and binary SVM. We adapt these methods from existing implementations provided by the Python `scikit-learn` toolkit (Pedregosa et al., 2011). These supervised classification techniques take the featurized reviews and corresponding binary labels generated during preprocessing (Section 2.1) as their input. We vary the learning parameters for each method to optimize classification performance based on validation sets. For perceptron, we test both L1 and L2 regularization. For SVM, we test both linear and RBF kernels with varying C and γ. For logistic regression, we test both L1 and L2 regularization with varying λ.

2.3. Multiclass Rating Prediction

We investigate multiclass rating prediction using both supervised and unsupervised methods: logistic regression, SVM, and k-means and spectral clustering. While logistic regression and SVM take the featurized reviews and multiclass labels as their input, the clustering methods take only the featurized reviews. We again adapt implementations from `scikit-learn` (Pedregosa et al., 2011). As in Section 2.3, we test L1 and L2 regularization with varying λ for logistic regression. SVM uses linear and RBF kernels with varying C and γ. We use $k = 5$ for k-means clustering - investigating both the default euclidean and angular cosine similarity distance functions - as well as spectral clustering with nearest-neighbors affinity.

3. Dataset

3.1. Dataset Construction

The whole Yelp dataset has over 3 million reviews. In order to focus our efforts on a smaller subset of that data, we only consider restaurant reviews. We choose the subset of restaurant reviews from the states of Arizona (AZ) and Nevada (NV), totaling about 1 million reviews. To construct smaller sets of data on which to perform classification, we segregate the reviews by state and construct 3 different datasets for each state that we perform all of our classification on.

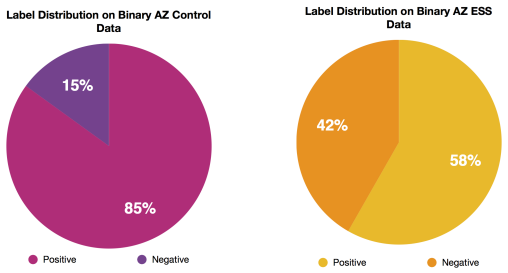

Figure 1. **Comparison between the Control and ESS binary training set label distributions.**

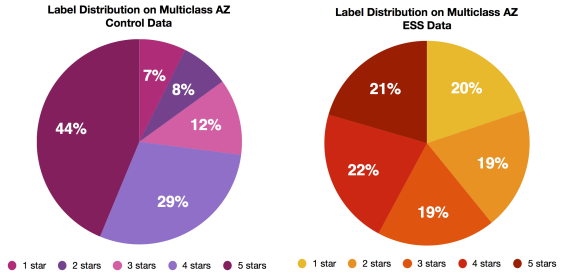

Figure 2. **Comparison between the Control and ESS multiclass training set label distributions.**

The review data is sorted by business, and we want multiple reviews from each business for consistency. Our first data collection takes the first 1000 reviews from each state and sequentially divides the file into train, validation, and test sets with a 60%-20%-20% split. This is the *control dataset*.

We randomly shuffle the control data to form new train, validation, and test sets so that each individual subset is not ordered with the same businesses. This is the *shuffled dataset*.

The control dataset lacks a diverse set of review ratings, enabling binary classifiers to achieve an almost 90% accuracy by simply predicting every review as positive. In order to balance the star ratings but keep multiple reviews from the same business in our dataset,

we take the first X number of reviews of each class (either binary or number of star ratings) from all of that state's reviews. All our datasets are made up of 1000 reviews, so binary data has 500 positive and 500 negative reviews, while multiclass data has 200 reviews of each star rating. Before splitting into train, validation, and test data, we randomly shuffle the set of 1000 reviews so each subset will have a similar but not equal distribution of classes. This is the *evenly split and shuffled (ESS) dataset*.

Figures 1 and 2 show the label distributions of the training subset of the control and ESS datasets for AZ reviews. The ESS sets are much more evenly-distributed than the the control sets.

Note that although we constructed both AZ and NV datasets, we mostly analyze classification results only on the AZ dataset throughout this paper, which had consistently better results.

3.2. Data Preprocessing

We illustrate in Table 1 that our preprocessing cleanup steps of lowercasing, stemming, and removing punctuation and stopwords generally increase validation accuracies. We use these cleanup procedures on training, validation, and test data in all subsequent reported results.

	training accuracy	validation accuracy	test accuracy
Perceptron			
with cleanup	99.2%	**74%**	72%
without	99%	**71%**	73%
Binary LR			
with cleanup	100%	**76%**	71.5%
without	99%	**73.5%**	77%
Bin. Linear SVM			
with cleanup	91%	**77%**	75%
without	100%	**71%**	73%
Bin. RBF SVM			
with cleanup	100%	**75%**	73.5%
without	100%	**67.5%**	74%

Table 1. **Preprocessing cleanup improves validation accuracy on ESS binary NV data.** We show training, validation, and test accuracy with and without cleanup (includes lowercasing, stemming, and removing punctuation and stopwords) for the highest achieved validation accuracies.

4. Analysis

In this section we present the results of our investigation of feature augmentation, featurization using occurrence and TFIDF, and alternate classifications of 3-star ratings. We also analyze the results of cross validation and unsupervised clustering methods.

4.1. Feature Augmentation

4.1.1. Binary Classification

We first try to classify restaurant reviews as only positive or negative. To create these labels, we treat reviews with a star rating of 3-5 as positive and 1-2 as negative. In Section 4.2, we explore treating 3 stars as a negative label.

Figures 3 - 5 show comparisons of perceptron, linear regression, and SVM on datasets with different features. The baseline dataset is featurized using occurrence on the review text, the City dataset is augmented only with the city the business is in, the ASR (average star rating) dataset is only augmented with the user's average star rating, and the City + ASR dataset is augmented with both.

In all cases, augmenting the data with both the city and ASR information improves the test set accuracy. Adding just the city data increases all classifiers except for logistic regression, where the accuracy decreases by 1%. Conversely, adding just the ASR data decreases the accuracies of all the classifiers except SVM.

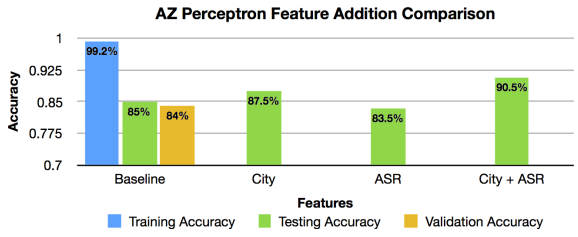

Figure 3. **Comparing augmented features using perceptron.**

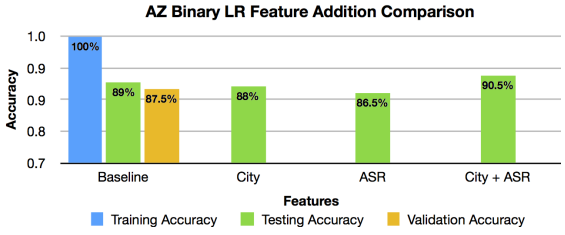

Figure 4. **Comparing augmented features using binary logistic regression.**

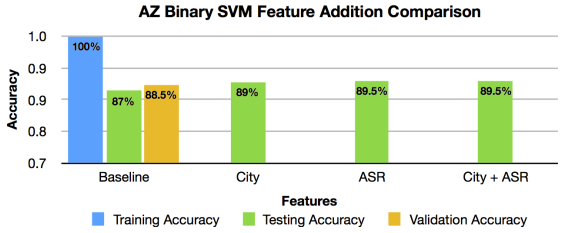

Figure 5. **Comparing augmented features using binary SVM.**

4.1.2. Multiclass Classification

Next we try to predict the integer star rating of each restaurant from 1-5 using multiclass logistic regression and SVM. The datasets are the same as those discussed in Section 4.1.1.

Figures 6 - 7 show comparisons of linear regression and SVM on the datasets with different feature augmentations.

Using the augmented datasets does not significantly change the classification accuracy of logistic regression. All augmented datasets do slightly improve SVM's test accuracy.

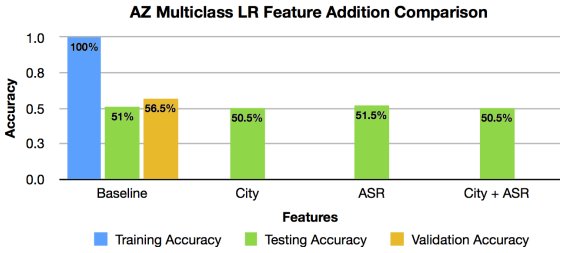

Figure 6. **Comparing augmented features using multiclass logistic regression.**

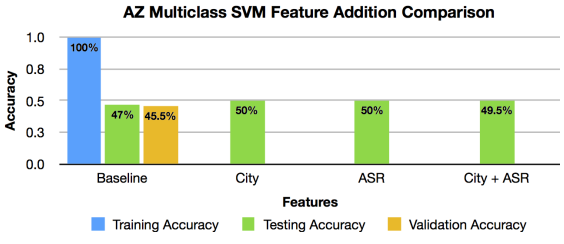

Figure 7. **Comparing augmented features using multiclass SVM.**

4.2. Alternate Binary Classification of 3-Star Ratings

For the binary classification task, we run our classifiers on ESS sets of data and compare treating a score of 3 stars as a positive vs. as a negative rating. Table 2 summarizes the results.

	test accuracy (3 is positive)	test accuracy (3 is negative)
LR (AZ)	**89%**	79.5%
Linear SVM (AZ)	**87%**	85%
RBF SVM (AZ)	**84%**	77%
Perceptron (AZ)	**85%**	**85%**
LR (NV)	71.5%	**72.5%**
Linear SVM (NV)	**75%**	74.5%
RBF SVM (NV)	73.5%	**77.5%**
Perceptron (NV)	**74%**	71.5%

Table 2. **Comparing test accuracies using 3 stars as a positive vs. negative label.**

For the AZ data, with the exception of perceptron, the accuracy with 3 stars as a positive label is higher than the accuracy with 3 stars as a negative label. Based on this data, we hypothesize that reviewers of AZ restaurants consider restaurants they rate 3 stars to still be decent restaurants.

The results from the NV data are less consistent. Since all the test accuracies are close, we hypothesize that reviewers in NV consider a 3 star rating to be more neutral than generally good or bad.

4.3. Occurrence vs. TFIDF

We explore constructing bag-of-words feature vectors with both occurrence and TFIDF. We expect that bag-of-words with TFIDF should produce higher accuracy because it normalizes term frequency by the size of the document, and gives less weight to common words that occur frequently. However, we find that nearly all the classifiers, both binary and multiclass, display no significant difference in accuracy when featurized using TFIDF instead of occurrence. This can be explained by the fact that our data preprocessing already removes commonly occurring stopwords (Section 2.1).

One anomalous case is perceptron, as shown in Figure 8. When augmented with average star rating information, perceptron performs significantly worse with TFIDF than with occurrence. This may occur because the average star rating feature is not normalized like the rest of the feature vector. Since the perceptron update rule includes a multiplication by the feature

vector (Bishop, 2006), this unnormalized feature could be greatly skewing classification results and overall accuracy.

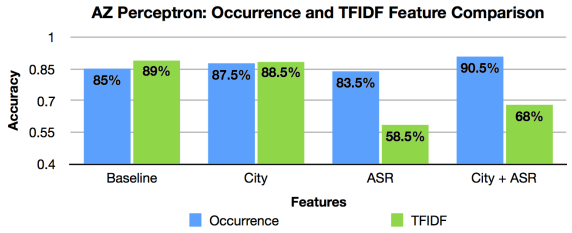

Figure 8. **Comparison of bag-of-words featurization with occurrence vs. TFIDF for perceptron on AZ data.**

4.4. Classification Cross Validation

After running various classification algorithms, we perform k-fold cross validation on 80% of our total datasets for each state, which are combinations of the states' respective training and validation sets. According to (Kohavi, 1995), a value of $k = 20$ achieves an estimate with low bias, however we observe using such a large value of k leads to slow estimations. To speed up the estimation, we use a value of $k = 5$, which does not report significantly different estimates from $k = 20$, but does significantly speed up the runtime.

Tables 3 and 4 summarize the differences between our previous classification results and the cross validation accuracy estimations. This data suggests that our test accuracies so far have been reasonable given our data.

	Test Accuracy	5-fold CV Accuracy	|Test-CV|
Perceptron	85%	84.1%	**0.9%**
Bin. LR	89%	87.3%	**1.7%**
Mul. LR	51.5%	48.9%	**2.6%**
Bin. Lin. SVM	87%	88%	**1%**
Bin. RBF SVM	84%	83.5%	**0.5%**
Mul. Lin. SVM	46%	47.1%	**1.1%**
Mul. RBF SVM	47%	44.5%	**2.5%**

Table 3. **AZ cross validation results.** The third column is the absolute value of the difference of test accuracy on the ESS dataset and 5-fold cross validation accuracy. All accuracies are very close, differing at most by 2.6%

	Test Accuracy	5-fold CV Accuracy	|Test-CV|
Perceptron	72%	75.9%	**3.9%**
Bin. LR	71.5%	80%	**8.5%**
Mul. LR	46.5%	45.8%	**0.7%**
Bin. Lin. SVM	75%	79.3%	**4.3%**
Bin. RBF SVM	73.5%	76.1%	**2.6%**
Mul. Lin. SVM	46%	43.1%	**2.9%**
Mul. RBF SVM	42%	41.4%	**0.6%**

Table 4. **NV cross validation results.** Unlike in the AZ cross validation results, the differences between the actual and predicted test accuracies on the NV ESS dataset are more spread apart. The largest difference is greater than 3 times the largest difference in the AZ data at 8.5%

4.5. Clustering

4.5.1. k-means Clustering

We run k-means clustering on the review data to see whether we can achieve cluster separation by business rating. We consistently use the binary occurrence featurization (using term frequency does not show any consistent difference) along with investigating other features. The distance function used by the `scikit-learn` implementation of k-means is by default Euclidean distance:

$$||v|| = \sqrt{v_1^2 + v_2^2 + ... + v_n^2} \tag{1}$$

We use an indirect approach involving the cluster statistics (i.e. mean, standard deviation, median, and mode) to judge whether a clustering is useful for the intended application (Feldman & Sanger, 2006), in this case to multiclass prediction of star ratings.

	mean	std.dev	median	mode
A	2.8	1.2	3	3
B	4	-	4	4
C	4	-	4	4
D	5	-	5	5
E	3.1	1.4	3	5

Table 5. **k-means with angular cosine similarity on ESS AZ training data with cities.** There appears to be some separation of clusters based on the means, but this may not be conclusive since only 1 point is assigned to clusters B, C, and D.

	mean	std.dev	median	mode
A	4.6	0.5	5	5
B	3.0	1.4	3	4
C	3	2.3	3	-
D	3.1	1.3	3	2
E	5	0.0	5	5

Table 6. **Cluster statistics of spectral clustering on ESS AZ training data with with Cities and Nearest-Neighbors affinity.** There is some separation of clusters based on the means, but again not be conclusive since so few points are assigned to most clusters.

	mean	std.dev	median	mode
A	2.8	1.4	3	2
B	2.4	1.5	2	2
C	2.5	0.7	2.5	-
D	2.7	2.1	2	-
E	3.3	1.7	3.5	-

Table 7. **Cluster statistics of spectral clustering on ESS AZ test data with with Cities and Nearest-Neighbors affinity.** Any separation of test cluster means is much less apparent than in the training clusters.

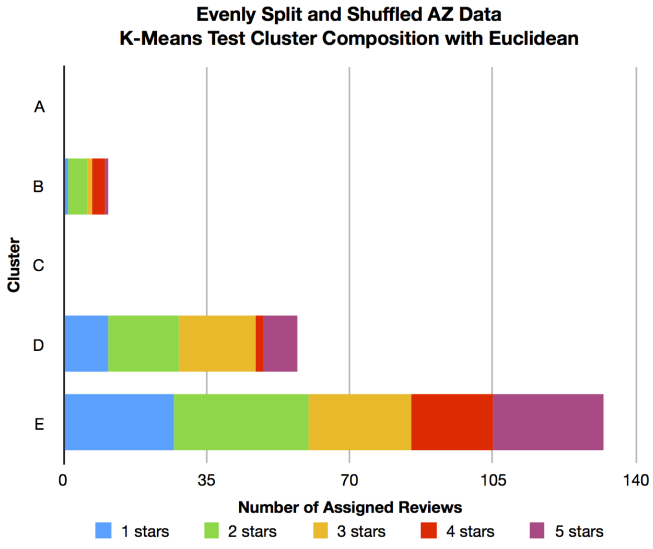

Figure 9. **Test cluster composition of k-means on ESS AZ Test Data with Euclidean distance and no additional features.**

We find that the clusters have similar compositions and cluster statistics to each other and across training, validation, and test data. We show cluster composition of test data in Figure 9. No consistent change occurs when additional features are used. We then investigate angular cosine similarity (Bishop, 2006), another valid distance function bounded between 0 and 1, for k-means clustering:

$$distance = \frac{cos^{-1}(\frac{AB}{||A||_2||B||_2})}{\pi} \quad (2)$$

$$similarity = 1 - distance \quad (3)$$

Using angular cosine similarity gives similar results in Figure 10, showing some separation by associated star ratings in the cluster statistics (see Table 5). Adding other features appears to consistently cause the training data to utilize all k clusters. However, the generalization to validation and test data remains poor: not all clusters are utilized and the cluster statistics on validation and test data do not show any separation.

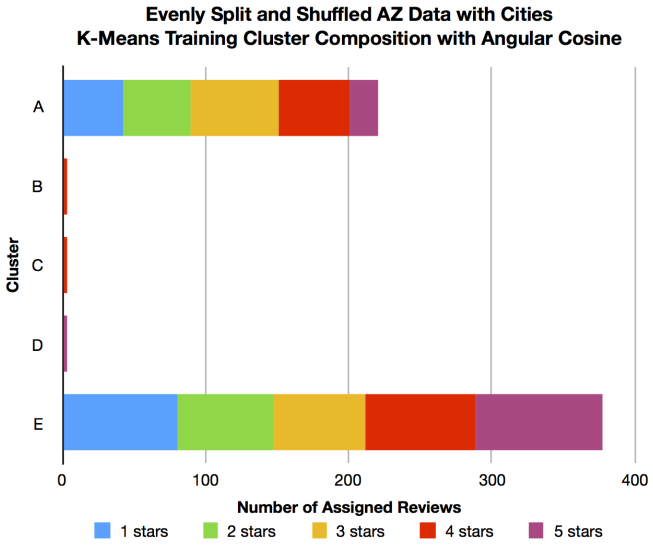

Figure 10. **k-means on ESS AZ Training Data with Angular Cosine and Cities feature.** Cluster statistics given in Table 5.

4.5.2. Spectral Clustering

The likely issue with k-means is that neither distance function works well to separate the review data. We attempt spectral clustering, which is able to discover other types of cluster configurations (e.g. encircling rings, which k-means cannot), and achieve similar results.

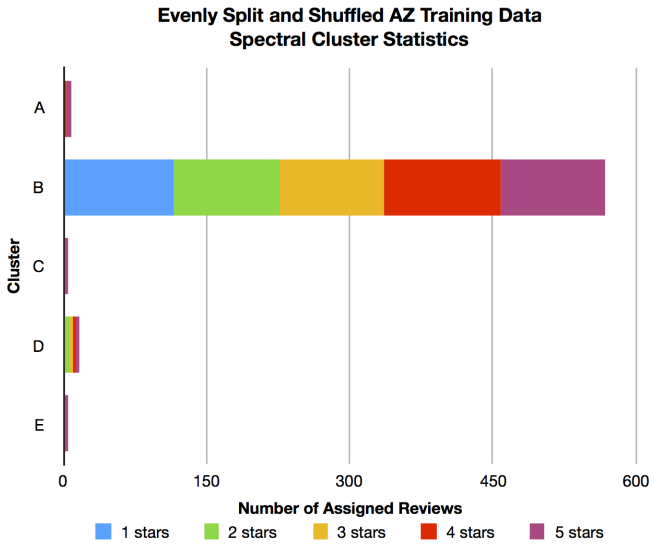

Figure 11. **Spectral clustering on ESS AZ training data with Cities and Nearest-Neighbors affinity.**

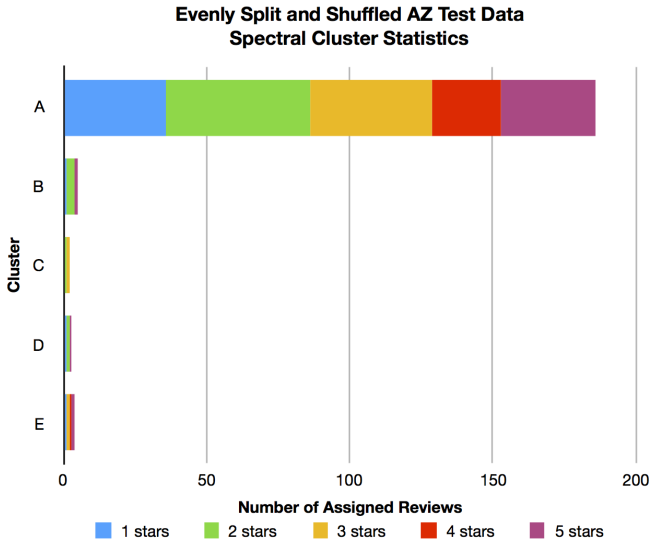

Figure 12. **Spectral clustering on ESS AZ test data with Cities and Nearest-Neighbors affinity.**

We use an affinity of nearest-neighbors, which is more complex and does not have to conform to the constraints of k-means distance functions, for spectral clustering's similarity metric. One difference is that spectral clustering with nearest-neighbors consistently utilizes all k clusters across training, validation, and test sets. While the cluster compositions do not change much and most points are assigned to the same cluster, we do observe the cluster statistics, particularly the mean, suggest some separation in the training data, as in Figure 11 and Table 6. However, we do not have a good way to prove this is not merely noise, since so few points are assigned to most clusters. This suggested separation is again lost in the generalization to the validation and test data, as in Figure 12 and Table 7, so spectral clustering is not necessarily better than k-means.

5. Discussion

We observe a few general trends in increasing classification accuracies of the binary and multiclass prediction problems. First, we note that the occurrence featurization appears to work better for the binary problem (see Table 13), while the term frequency or frequency featurization works slightly better for the multiclass problem (see Table 14).

However, the improvements are minimal, which is likely due to the fact that our preprocessing already removes commonly occurring words, and that the feature vectors are extremely sparse to begin with. As a result, normalizing and weighting with TDIDF does not have much of an impact on featurization.

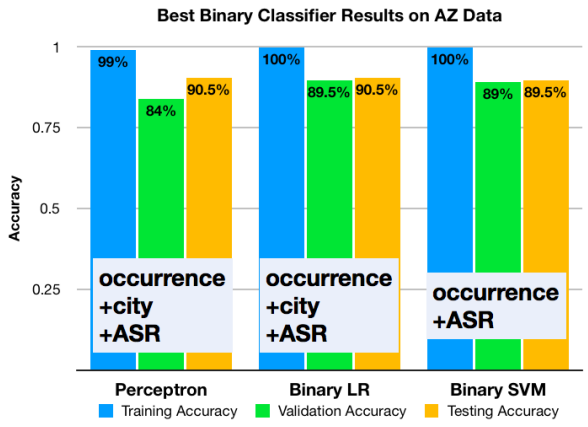

Figure 13. **Best achieved Test Accuracies for Binary Classifiers on ESS AZ Data.** Observe that all binary classifiers exhibit best performance with occurrence featurization.

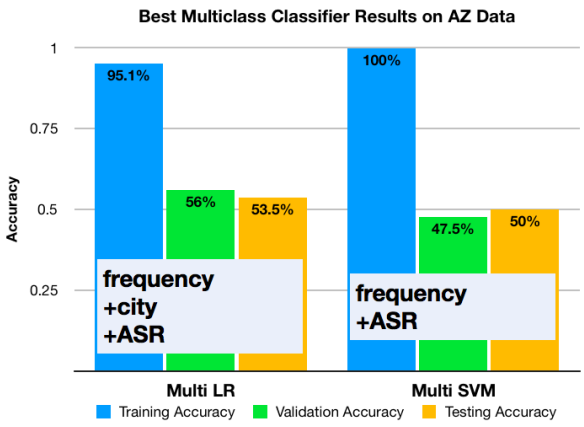

Figure 14. **Best achieved Test Accuracies for Multiclass Classifiers on ESS AZ Data.** In contrast to the binary classifiers, the multiclass classifiers exhibit best performance with frequency featurization (and *TFIDF* regularization).

From Figures 13 and 14, we see that for both the binary and multiclass SVMs, the addition of the user's average star rating produces the best results, but the improvements from augmentation are again very nominal. In fact, we only see significantly improved classification results on perceptron. This may be because the information we chose to augment the feature vectors with (city, average star rating) may not actually be strong indicators of a user's rating, if there exists a lot of variation in rating within a single user or city, for instance. Alternatively, there may not have been enough data for a given city or user for the classifiers to learn trends based on these features.

As seen in Figure 13, all three binary classification techniques consistently achieve close to 90% accuracy. On the other hand, multiclass classification displays poor accuracy at only around 50%. One possibility, which was hinted at during the discussion of treating a 3-star rating as a positive or negative review, is that there is a very large variance across how people describe a specific star rating. For example, two different users might both rate a restaurant with 4 stars, but one could describe it similar to how another user might describe another restaurant they gave 5 stars, whereas the other user might describe it similar to how someone else would describe a 3-star restaurant.

For binary classification, language people use to describe good or bad experiences is probably much more similar than the language that differentiates a specific star rating. This observation is corroborated by the fact that none of the clustering algorithms we tested were able to clearly distinguish among the five classes of star ratings, even with data augmentations. Another possibility is that we are not testing on a large enough dataset.

6. Future Work

In the future, we would like to explore further variations in featurizing the review text. This includes augmenting the bag-of-words representation with more informative features, such as restaurant category, number of reviews written by user, and elite status of user. We can try to normalize the augmented feature vectors so that they do not skew results as observed with perceptron.

In addition, it would be interesting to investigate other featurization techniques besides bag-of-words, such as `doc2vec`. This would involve training a model on a large representative corpus of text, in order to learn a more fitting feature vector for the review document. The limitation of the current bag-of-words representation is that the vectors are extremely sparse and do not encode much semantic meaning about the distance between review documents.

Finally, we would like to run our classifiers on much larger datasets. This would give us more insight into the variation in individual users' rating standards.

7. Source Code and Division of Labor

Our repository is available at `https://github.com/zareenc/6.867-project`. We did not upload Yelp's data since we are not authorized distributors, but our full pipeline is available. The work was divided as follows:

- Ayesha: wrote the cleanup and featurizing (basic and with additional features) code, analysis code for perceptron, and analysis and statistics code for clustering (kmeans and spectral).
- Katy: wrote most of the code for filtering and splitting the data, binary and multiclass SVM, cross-validation, and treating 3 stars as a positive and negative class label.
- Zareen: wrote binary and multiclass logistic regression, pipeline for augmenting features with user data, and data file type conversion.

Additionally, we all worked together on running the classifiers on the various datasets, producing tables and figures, and writing this report. This project was not used for any other classes.

8. Acknowledgements

We would like to thank Alice Zhan, our 6.867 TA, for her guidance in this project.

References

Asghar, Nabiha. Yelp dataset challenge: Review rating prediction. *CoRR*, abs/1605.05362, 2016. URL http://arxiv.org/abs/1605.05362.

Bishop, Christopher M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

Feldman, Ronen and Sanger, James. *Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, New York, NY, USA, 2006. ISBN 0521836573, 9780521836579.

Kohavi, Ron. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, pp. 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-363-8. URL http://dl.acm.org/citation.cfm?id=1643031.1643047.

Loper, Edward and Bird, Steven. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pp. 63–70, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1118108.1118117. URL https://doi.org/10.3115/1118108.1118117.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Sparck Jones, Karen. Document retrieval systems. chapter A Statistical Interpretation of Term Specificity and Its Application in Retrieval, pp. 132–142. Taylor Graham Publishing, London, UK, UK, 1988. ISBN 0-947568-21-2. URL http://dl.acm.org/citation.cfm?id=106765.106782.