Applications of Machine Learning to Predict Yelp Ratings

Kyle Carbon
Aeronautics and Astronautics
Stanford University
Stanford, CA
kcarbon@stanford.edu

Kacyn Fujii
Electrical Engineering
Stanford University
Stanford, CA
khfujii@stanford.edu

Prasanth Veerina
Computer Science
Stanford University
Stanford, CA
pveerina@stanford.edu

***Abstract*—In this project, we investigate potential factors that may affect business performance on Yelp. We use a mix of features already available in the Yelp dataset as well as generating our own features using location clustering and sentiment analysis of reviews for businesses in Phoenix, AZ. After preprocessing the data to handle missing values, we ran various feature-selection techniques to evaluate which features might have the greatest importance. Multi-class classification (logistic regression, SVM, tree and random forest classifiers, Naive Bayes, GDA) was then run on these feature subsets with an accuracy of $\sim 45\%$, significantly higher than random chance (16.7% for 6-class classification). Regression models (linear regression, SVR) were also tested but achieved lower accuracy. In addition, the accuracy was approximately the same across different feature sets. We found that across feature selection techniques, the important features included positive/negative sentiment of reviews, number of reviews, location, whether a restaurant takes reservations, and cluster size (which represents the number of businesses in the surrounding neighborhood). However, sentiment seemed to have the largest predictive power. Thus, in order to improve business performance, a future step would be to conduct additional analysis of review text to determine new features that might help the business to achieve a higher number of positive reviews.**

I. Introduction

Yelp, founded in 2004, is a multinational corporation that publishes crowd-sourced online reviews on local businesses. As of 2014, Yelp.com had 57 million reviews and 132 million monthly visitors [1]. A portion of their large dataset is available on the Yelp Dataset Challenge homepage, which includes data on 42,153 businesses, 252,898 users, and 1,125,458 reviews from the cities of Phoenix, Las Vegas, Madison, Waterloo, and Edinburgh [2]. For businesses, the dataset includes business name, neighborhood, city, latitude and longitude, average review rating, number of reviews, and categories such as "good for lunch". The dataset also includes review text and rating.

Our goal was to analyze what factors may affect the performance of a business on Yelp. Specifically, we wanted to investigate the effect of location and other business attributes vs. quality of food and service (measured indirectly through review sentiment) on the business' rating. In this paper, we investigate several models to determine which factors have the greatest effect on a business' Yelp rating, which can then be used to help them improve their Yelp rating. To increase the tractability of the problem at hand, we limit ourselves to data from businesses and reviews in Phoenix, AZ.

To determine potential features of interest, we explore clustering of business location (as represented by latitude and longitude), as well as sentiment analysis of reviews. We then use methods of feature selection to determine which features best predict business performance, as represented by star rating. Finally, we implement and evaluate different prediction models.

II. Data

The dataset we used is available on the Yelp Dataset Challenge homepage. We limited ourselves to businesses and reviews from Phoenix, AZ, which included 7,499 business records and 164,890 reviews. Business and review data were available in two separate files, each with a single object type, one json-object per-line.

The business records had the following format:

```
business
{
    'type': 'business',
    'business_id': (encrypted business id),
    'name': (business name),
    'neighborhoods': [(hood names)],
    'full_address': (localized address),
    'city': (city),
    'state': (state),
    'latitude': latitude,
    'longitude': longitude,
    'stars': (star rating, rounded to half-
        stars),
    'review_count': review count,
    'categories': [(localized category names)]
    'open': True / False (corresponds to
        closed, not business hours),
    'hours': {
        (day_of_week): {
            'open': (HH:MM),
            'close': (HH:MM)
        },
        ...
    },
    'attributes': {
        (attribute_name): (attribute_value),
        ...
    },
}
```

The reviews had the format:

```
review
{
    ’type’: ’review’,
    ’business_id’: (encrypted business id),
    ’user_id’: (encrypted user id),
    ’stars’: (star rating, rounded to half-
        stars),
    ’text’: (review text),
    ’date’: (date, formatted like
        ’2012-03-14’),
    ’votes’: {(vote type): (count)},
}
```

We imported these json files into Python arrays using NumPy [3] for our analysis.

III. Feature Generation

A number of the features we evaluated came directly from the Yelp dataset, such as the number of reviews the business has. Specifically, the ones we chose were:

- latitude
- longitude
- review count
- price range
- accepts credit cards
- has take-out
- has delivery
- is wheelchair accessible
- good for lunch
- good for dinner
- good for brunch
- good for breakfast
- takes reservations

In addition to these, we generated additional features using clustering and sentiment analysis to determine if there were other features that may be useful in predicting business performance.

A. *K-means Clustering for Location*

K-means clustering can be used to create additional labels for each business which correlate to its location in a city. For example, clustering could reveal downtown locations, shopping malls, and other popular gathering places. This gives additional information not apparent from just latitude and longitude, such as the number of nearby businesses (determined by cluster size). This also makes it easier to implement as a feature when compared to the business’ neighborhood (formatted as a string). We ran the clustering algorithm for $k = 15$, which returned results which seem to overlap with some neighborhoods. A visualization of the clustering results is shown in Figure 1.

The features we used from k-Means are the cluster labels and sizes, which represent the neighborhood and the size of the neighborhood, respectively.

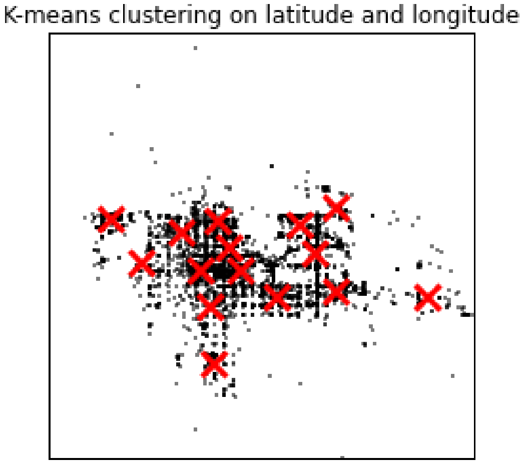

Fig. 1. Results from k-Means clustering on latitude and longitude for businesses in Phoenix, AZ with k=15

B. *Naive Bayes Classifier for Review Sentiment Analysis*

We trained a Naive Bayes Classifier to classify business reviews as either positive or negative. We used 20,000 reviews from the Yelp dataset, with 75% of the data as the training set and 25% held out as a development set.

In order to make this a binary classification problem, we defined the negative label to be star rating less than or equal to 3 and positive label to be star rating greater than 3 (reviews are labeled with a 0-5 star rating). We began with a simple bag of words as our feature set. Each review was processed into a set of indicator variables for each word in the review. This simple bag of words did not produce a very accurate model (Accuracy of 0.477). Although the overall bag of words introduces a huge number of features, we saw that the most important features were actually very few in number.

Most Informative Features	Neg:Pos Ratio
Horrible	58.4 : 1.0
Awful	55.0 : 1.0
Disappointing	40.9 : 1.0
Waited	38.1 : 1.0
Worst	34.5 : 1.0
Disgusting	32.5 : 1.0
Terrible	30.9 : 1.0
Wrong	29.6 : 1.0
WORST	27.9 : 1.0
Overpriced	27.9 : 1.0

We tried to address this problem of having too many uninformative features by only using the top performing features. This made a significant difference in our results. The table below summarizes the accuracies achieved by culling the number of features down to the n-most important.

# feat.	Accuracy	Pos Prec., Recall	Neg Prec., Recall
All	.477	.988, .472	.303, .975
10000	.568	.991, .506	.319, .982
5000	.597	.982, .562	.340, .958
500	.638	.982, .562	.340, .958
100	.765	.946, .753	.439, .817

Note that when using all the word features we classify too many reviews as negative (high negative recall, low negative precision), but by culling the number of features we get better overall classification because we don't classify negative reviews as aggressively (lower negative recall, higher negative precision and higher positive recall).

Next we added bigram features with the intuition being words like "awful" and "great" might be negated in cases like "awful good" or "not great". This actually decreased our classifier accuracy to 0.688. Our hypothesis is that adding a huge number of new bigram features diluted the predictive power of the unigram features.

We used the trained unigram classifier with 100 features (76.5% accuracy) to generate a sentiment feature for the businesses in our training, development, and test sets. This was done by computing the ratio of positive reviews to total reviews for each business. This sentiment feature is our indirect way of measuring a businesses food and customer service quality, since most of our sentiment features deal with qualitative factors.

C. Data Pre-processing

Before running our feature selection algorithm, we scaled the data to have zero mean and unit variance. To handle missing data, we performed imputation using scikit's Imputer. We selected the mean strategy after observing that none of the mean, median, and most frequent strategies significantly outperformed the others. Categorical variables were represented by 1 for true and 0 for false. It is worth noting that for the multinomial Naive Bayes classifier, the data is unscaled since the scikit implementation required positive-valued features.

IV. Feature Selection

After generating features, we explored various methods of feature selection to determine which features might be most useful in predicting business performance.

A. Univariate Feature Selection

Univariate feature selection works by conducting various univariate statistical tests on the feature set, then selecting the features that performed best [4]. We used the Anova F-value as the scoring function for the feature set, then selected the top two features with the highest score since the F-values for these features were significantly higher than the rest.

B. Recursive Feature Elimination

We used scikit's recursive feature elimination implementation [5], which assigns weights to each feature and prunes the feature with the smallest weight at each iteration. This was performed in a cross-validation loop to find the optimal number of features. Recursive feature elimination was implemented for SVM, logistic regression, Naive Bayes, and GDA. Due to time constraints, we were not able to implement our own version of recursive feature elimination for regression algorithms.

C. Tree-Based Feature Selection

We used a tree-based estimator to compute feature importances. Specifically, we used the Extra Trees Classifier from scikit [6], which fits ten randomized decision trees on subsamples of the dataset and averages them to improve accuracy and reduce over-fitting. We used the default "gini" criterion to evaluate splits and selected those features with the highest importance score.

V. Prediction Models

We implemented the following models using scikit libraries: support vector machines (SVM), logistic regression, Multinomial Naive Bayes, Gaussian discriminant analysis (GDA), decision trees and random forest classifiers, linear regression with regularization and support vector regression (SVR). Most algorithms were implemented with the defaults from scikit, which can be found in their user guide [7]. However, we investigated the types of kernels for SVM and the regularization parameters for logistic and linear regression.

A. Support Vector Machine

Keeping all input features, we experimented using scikit's SVM [8] with linear, polynomial and Gaussian kernels. Our results are shown in Table I.

Kernel	Training Accuracy	Test Accuracy
Linear	.4676	.4619
Gaussian	.4672	.4582
Polynomial, $d = 2$	.4395	.4121
Polynomial, $d = 3$	.5137	.4618
Polynomial, $d = 4$	.5089	.4407

TABLE I
Classification results with regularization for logistic regression.

Based on these results, we used linear kernels moving forward.

B. Multinomial Logistic Regression

We implemented multinomial logistic regression [9] with regularization using the entire feature set. scikit implements regularization using parameter C, which is inversely proportional to the strength of regularization.

While regularization did have an effect on testing accuracy, it was essentially negligible, so we used $C = 1.0$.

C. Multinomial Naive Bayes

We used scikit's implementation of a Multinomial Naive Bayes classifier [10] with a Laplace smoothing parameter of $\alpha = 1.0$.

C	Training Accuracy	Test Accuracy
10^2	0.4618	0.4562
10^1	0.4618	0.4563
10^0	0.4617	0.4565
10^{-1}	0.4615	0.4567
10^{-2}	0.4610	0.4573

TABLE II
CLASSIFICATION RESULTS WITH REGULARIZATION FOR LOGISTIC REGRESSION.

D. Gaussian Discriminant Analysis

We used the LDA classifier available with scikit [11] for our 6-class classification problem with all of the default settings.

E. Decision Trees and Random Forest Classifier

For decision trees and the random forest classifier, we used the scikit implementation with the "gini" criteria to pick the best split [12] [13]. Our random forest classifier averaged over ten trees.

F. Linear Regression with Regularization

Using the ridge regression implementation in scikit [14], we implemented linear regression with L2 regularization, parameterized by α. Once again, we saw little effect of the regularization on the accuracies (Table III).

α	Training Accuracy	Test Accuracy
10^4	0.3394	0.3392
10^2	0.3883	0.3867
10^0	0.3894	0.3884
10^{-2}	0.3895	0.3883
10^{-4}	0.3883	0.3883

TABLE III
PREDICTION RESULTS WITH REGULARIZATION FOR LINEAR REGRESSION.

Thus, in our final model, we selected the default parameter of $\alpha = 1.0$.

G. Support Vector Regression

Similarly to SVM, we used the scikit implementation of SVR with a linear kernel [15].

VI. RESULTS

A. Prediction Results

When making predictions using classification models, we floored the business' rating $[0, 5] \cap \mathbb{Z}$ as part of the pre-processing. When using a regression model, we floored the business' rating only when comparing the prediction to the actual rating. For all models, we implemented cross validation over ten iterations, withholding 30% of the data as the test set.

In Tables IV–VII, we present our prediction results using the entire feature set, recursive feature selection, univariate feature selection, and tree-based feature selection.

Model	Training Accuracy	Test Accuracy
SVM	0.4558	0.4483
Logistic Regression	0.4505	0.4469
Naive Bayes	0.2854	0.2847
GDA	0.4642	0.4598
Decision Tree	0.9007	0.4227
Random Forest	0.8835	0.4562
Linear Regression w/ regularization	0.3907	0.3904
SVR	0.3932	0.3931

TABLE IV
PREDICTION RESULTS USING THE ENTIRE FEATURE SET.

Model	Training Accuracy	Test Accuracy
SVM	0.4514	0.4462
Logistic Regression	0.4540	0.4515
Naive Bayes	0.3886	0.3880
GDA	0.4620	0.4609

TABLE V
PREDICTION RESULTS WITH RECURSIVE FEATURE SELECTION.

Model	Training Accuracy	Test Accuracy
SVM	0.4410	0.4402
Logistic Regression	0.4328	0.4325
Naive Bayes	0.3340	0.3322
GDA	0.4475	0.4467
Decision Tree	0.6065	0.4315
Random Forest	0.5979	0.4391
Linear Regression w/ regularization	0.3838	0.3833
SVR	0.3953	0.3951

TABLE VI
PREDICTION RESULTS WITH UNIVARIATE FEATURE SELECTION.

Model	Training Accuracy	Test Accuracy
SVM	0.4408	0.4406
Logistic Regression	0.4367	0.4355
Naive Bayes	0.2870	0.2867
GDA	0.4558	0.4562
Decision Tree	0.7528	0.3886
Random Forest	0.5974	0.4362
Linear Regression w/ regularization	0.3827	0.3821
SVR	0.3931	0.3921

TABLE VII
PREDICTION RESULTS WITH TREE-BASED FEATURE SELECTION.

B. Feature Selection Results

This section details the most important features selected. Recursive feature selection with logistic regression selected the following eight features: sentiment, reservations, latitude, longitude, review count, good for lunch, good for dinner, and cluster size. The accuracy versus number of features selected is shown in Fig. 2. Meanwhile, univariate feature selection determines that the features in Table VIII are the most important, which is corroborated by tree-based feature selection in Table IX.

VII. DISCUSSION

Across feature selection techniques and multi-class classification models, we observed an accuracy of $\sim 45\%$. Regres-

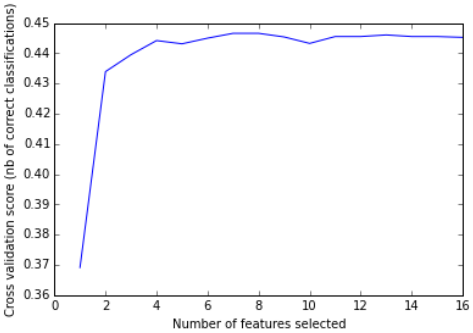

Fig. 2. Cross validation accuracy vs. number of features for recursive feature selection using logistic regression.

Feature	Anova F-Value
Sentiment	584.0
No. of reviews	75.66
Cluster size	18.04
Reservations	9.416
Longitude	9.303

TABLE VIII
THE TOP 5 MOST IMPORTANT FEATURES ACCORDING TO UNIVARIATE FEATURE SELECTION USING A SVM WITH A LINEAR KERNEL.

Feature	Importance Value
Sentiment	0.2072
Latitude	0.1894
Longitude	0.1892
No. of reviews	0.1570
Cluster size	0.0428

TABLE IX
THE TOP 5 MOST IMPORTANT FEATURES ACCORDING TO TREE-BASED FEATURE SELECTION USING A SVM WITH A LINEAR KERNEL.

sion techniques and Multinomial Naive Bayes achieved lower accuracy. For 6-class classification, where random chance is 16.7%, our accuracy results indicate that features such as location, price range, and the option of take-out have significant predictive power. This means if a business improves upon these features, they should be able to improve their business rating.

As corroborated by multiple feature selection techniques, the most important feature was the review's sentiment, which reflects upon the business' quality of service. This ultimately drives its rating on Yelp. Unsurprisingly, while other features like location can help, a business looking to improve should first focus on its service. As such, future work should involve detailed sentiment analysis of review text, which is essentially improving the feature set. Additionally, the maximum test accuracy achieved was 46.19% across all models. Given that a large variety of algorithms were used, new features are most likely needed to further improve accuracy. Future work should not only improve the sentiment analysis but also examine other possible features to use in rating prediction.

It should also be noted that not all businesses have the same goals. There are restaurants that aim for Michelin stars, while others are clearly in the take-out category. It is likely that customers take this into account when reviewing these businesses. As such, clustering can be used to determine certain common categories of businesses. Predictions can then be made within each category for the Yelp star rating, which may improve the accuracy of predictions. Conceivably, there are competing features for certain businesses, as speed is valued much more at a fast-food option, whereas quality is much more valued at a high-end restaurant.

VIII. Conclusion

Feature selection methods found our sentiment classifier to be an important feature, followed by location, number of reviews, cluster size, and whether or not the restaurant takes reservations. We observed accuracies of $\sim 45\%$ across 6-class classification and feature selection models, suggesting that review sentiment had the most power in predicting business rating on Yelp.

IX. Future Work

Future work will include investigating the generation of new features based on review text and user data that may hold predictive power. In particular, analyzing review text to develop new features or stronger models for sentiment analysis may be a promising direction. In addition to sentiment, the review text and user data could also be analyzed to gather other features about the business that can be improved. Sorting businesses into categories before running predictions may also help to improve accuracy.

Another future area of interest might be to pair Yelp data with sources of other data, like Walkscore or weather information for the location.

References

[1] Yelp - Wikipedia [Online]. Available: http://en.wikipedia.org/wiki/Yelp

[2] Yelp Dataset Challenge [Online]. Available: http://www.yelp.com/dataset_challenge

[3] NumPy [Online]. Available: http://www.numpy.org/

[4] sklearn Select Percentile [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectPercentile.html

[5] sklearn RFECV [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFECV.html

[6] sklearn RFECV [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html

[7] scikit learn user guide [Online]. Available: http://scikit-learn.org/stable/user_guide.html

[8] sklearn SVC [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[9] sklearn Logistic Regression [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[10] sklearn Multinomial Naive Bayes [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html

[11] sklearn LDA [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.lda.LDA.html

[12] sklearn Decision Tree Classifier [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

[13] sklearn Random Forest Classifier [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[14] sklearn Ridge Regression [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html

[15] sklearn SVR [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html