PREDICTING HEALTH INSPECTION GRADES OF RESTAURANTS

Team: SALSA

Antonio Robayo (amr1059), Laureano Nisenbaum (lvn218), Sammie Kim (sk7327), Shreyas Chandrakaladharan (sc6957)

1. Motivation

Every restaurant undergoes regular health and hygiene inspections by government agencies in the United States. The results of these health inspections are letter grades based on the amount of health code violations issued. In certain cities, it is required for restaurants to post grade cards where they can easily be seen by potential customers. In addition, previous inspection results are available online to the general public. These inspection grades inform customers of restaurant hygiene and have shown to substantially affect the revenue of businesses (e.g., Jin and Leslie (2005)).

In this project, we aim to predict the inspection grades of restaurants using data from previous inspections. Moreover, various studies have shown that valuable insights into the hygiene status of a restaurant can be gleaned from user reviews and ratings made about the restaurant on Yelp. Therefore, we also aim to show that contemporaneous user reviews on Yelp can help predict the next inspection grade. We believe that the resulting insight will benefit both restaurant owners as well as government agencies. Based on our predictions, restaurant owners can proactively improve conditions of the restaurant, if needed. Government agencies, on the other hand, can properly staff and prioritize their inspection schedules based on our predictions.

2. About the Data

2.1 Health Inspection Data

Our primary data source is the dataset containing health inspection records of food establishments in Las Vegas, Nevada. These records are collected when the Southern Nevada Health District makes unannounced inspections of food establishments in Las Vegas. Each record provides a snapshot view of the hygiene status of the restaurant at the time and day of the inspection. Inspection records are posted online approximately 5 days following the inspection and are made freely available online for the public.

The dataset had a total of 166k records and 23 columns. Two health inspection records have been given in Fig. 1 as a sample. Only the most important columns have been listed.

Restaurant Name	Barley's Brewer's Cafe	Mirchi
Category Name	Restaurant	Bar
Address	4552 SPRING MOUNTAIN	1950 N Rainbow Blvd
Zip	89122-6010	89149-4574
Inspection Time	2011 Mar 10 12:30:00 PM	2011 Mar 13 12:50:00 PM
Current Demerits	2	0
Current Grade	A	A
Inspection Demerits	3	12
Inspection Grade	A	B
Inspection Type	Routine Inspection	Re-Inspection
Violations	204,203,209	10,11,12,13,14,15,18,19,20,21,22,23
Location	(36.1214517, 115.1696112)	(36.0519048, 115.1716222)

Figure 1. Example of two health inspection records

In Fig. 1, **Current Demerits** and **Current Grade** represent number of the violations and letter grade of the restaurant respectively before the start of the inspection. **Inspection Demerits** represents the number of violations by the restaurant found during the inspection. **Inspection Grade** represents the new letter grade assigned to the restaurant after the inspection. **Violations** represents the list of violations found in the inspection.

We initially trained our classifier on this data to predict Inspection Grades. Later we used Yelp data to see if we could improve the performance of the classifier.

2.2 Yelp Data

Our supplementary source was the Yelp Data. We used Yelp data to see if we could improve the performance of our classifier.

Choosing Yelp data as a way to improve our performance was a rather intuitive choice because Yelp ratings and reviews convey the customers' sentiment about a food establishment. We hypothesized that the customers' sentiment correlates with the hygiene levels of the establishment. For example, a customer who steps into an unclean restaurant is likely to report the same on Yelp if she is an active user of Yelp.

Yelp's data was freely available in JSON format as part of its Dataset Challenge. It was provided in two files:

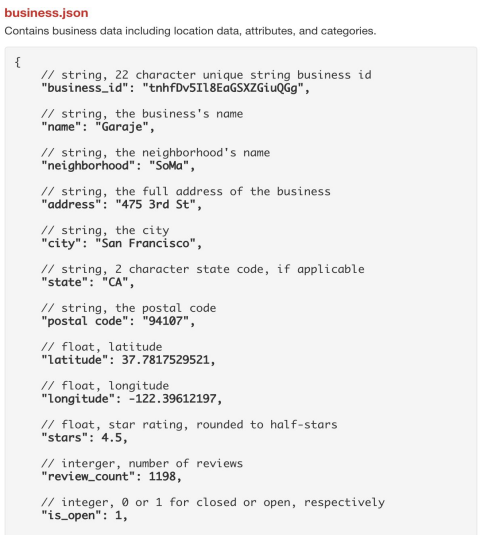


Figure 2. ***yelp_business.json****: Business data describing establishments, e.g name, category, overall rating, ambience.*

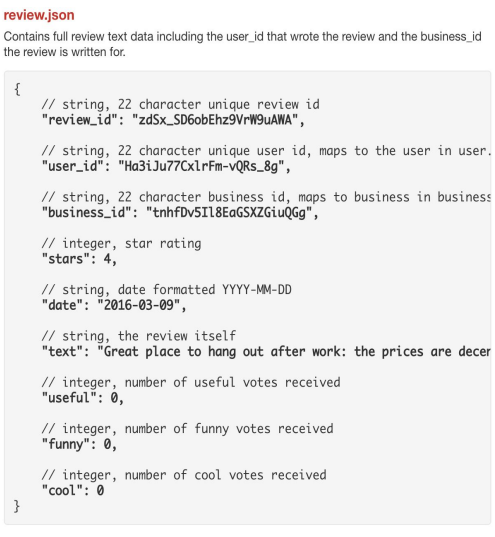


Figure 3. ***yelp_review.json:*** *Review data containing full text written by a given user and attributes measured in integer values that describe how many times users voted a review as either 'cool', 'useful' or 'funny'.*

Note: We made the decision to use the health inspection data of restaurants in Las Vegas particularly because the Yelp data had maximum matches for Las Vegas restaurants. Initially, we wanted to use the health inspection data and Yelp data to predict inspection grades for restaurants in New York. However, the number of NYC restaurants we could match was only 19. With LV, we could match 35,688 entries. We determined this by taking a list of NYC and LV zip codes[1] and subsetting Yelp's business data.

2.3 Target Variable

The target variable is the column '**Inspection Grade**' in Fig. 1. It corresponds to a letter grade that is related to the number of health code violations recorded in a restaurant's most recent inspection. A letter grade of 'A' indicates that a restaurant received 13 or fewer health code violations in their last inspection. A letter grade of 'B' indicates 14 to 27 recorded violations at the restaurant. A letter grade of 'C' indicates more than 28 health code violations.

There were other inspection grades documented in the health inspection records (e.g., X for "closed with fees", O for "approved, and P for "passed). However, only grades 'A', 'B' and 'C' were taken into consideration as these are the most critical indicators of hygiene status. We discarded the records with inspection grades other than 'A','B' or 'C'.

2.4 Inherent Skew in Data

As observed, there is an inherent skew in the data with most of the records being present for 'A':0 (88%) and very few for 'B':2 (5%) and 'C':1 (7%) comparatively.

[1] https://www.health.ny.gov/statistics/cancer/registry/appendix/neighborhoods.htm

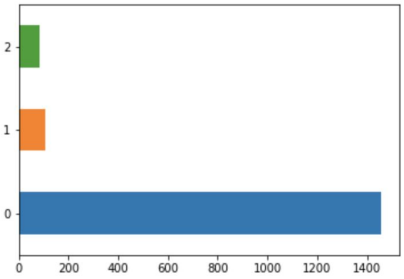


Figure 4. Distribution of the Target Variable Inspection Grade

2.5 Selection Bias

Yelp has created an environment where a common person can become a food critic and his/her opinions can be easily shared with and even rated by readers. Due to this social networking aspect, there are more reviews and comments made on establishments that are geared towards younger customers. As a result, there's a scarcity of review data around restaurants that are less popular or target older generations.

3 Overview of Methodology

We followed an agile approach to ensure we move quickly and fail fast in case of any obstacles. Following the iterative/agile approach, we built our project in increments. We started with basic features and built a baseline model. We engineered more features and built more sophisticated models to improve our classifier as we progressed further.

Baseline: First, to implement a baseline model, we used data only from previous health inspection data (e.g. category name, inspection type, zip, permit status, etc.). We attempted to draw insights from past inspections alone to see whether these would be good indicators of the outcome of future inspections. We used a Decision Tree (DT) for our baseline model as it is simple and robust. Additionally, we could also get a sense of the feature importance.

Increment 1: Next, we augmented our DT model with features from the Yelp data such as the restaurant's overall rating on Yelp, characteristics (e.g. neighborhood, price range, etc.) to see if these variables enhanced the predictive power of our model.

Increment 2: Lastly, we focused on the reviews about a restaurant on Yelp. Using this customer input, we hypothesized that negative reviews will tend to correlate with a higher number of violations which will affect the outcome of future inspections. The review data has important variables like 'text; (body of the review), 'review rating', etc. So we engineered new features using the review data and passed it to our DT model to test the effectiveness of using review data.

As we built more features, we simultaneously started experimenting with models other than DT such as Logistic Regression (LR) and Random Forests (RF).

4. Data Preparation

4.1 Data Integration

Part of the challenge of using health inspection grades in combination with Yelp data was finding a way to link the two. Although Las Vegas' publicly available restaurant data did have longitude and latitude features, it was not of the same degree of precision as Yelp's longitude and latitude features. From Fig. 5 we see this discrepancy in Las Vegas' restaurant coordinates (**feature name: location 1**) compared to the coordinates provided by Yelp (**feature name: latitude, longitude**). Our initial attempt involved rounding the coordinate values up to a certain decimal point. This, however, resulted in many spurious matches.

	location 1	latitude	longitude
26	(36.2388477, 115.232952)	36.238454	-115.232391
133	(36.2322239, 115.2509803)	36.232224	-115.250980
287	(36.0673016, 115.1722227)	36.068218	-115.175222
319	(36.272201, 115.260982)	36.271474	-115.260613
335	(36.143173, 115.207894)	36.144000	-115.206161
394	(36.1657583, 115.1329346)	36.165758	-115.132935
430	(36.158457, 115.126522)	36.158475	-115.126549

Figure 5. Longitude and latitude precision in data

Afterwards, we tried to use restaurant names and addresses to join the two datasets. We proceeded by normalizing these two features through removing whitespace and converting all letters to lowercase. Following which, we were able to successfully match 388 restaurants. Once we were able accurately link the two datasets, we began eliminating restaurant instances that were incomplete (e.g missing or pending health grade) or that did not have an associated Yelp counterpart (e.g empty name and address field).

Once we knew which restaurants we were working with, it was fairly simple to join the Yelp reviews as we could merge on **business_id**, a unique key generated by Yelp for each restaurant.

4.2 Feature engineering

Due to the fact that inspection grades and yelp reviews are a function of time, we needed to choose an appropriate interval in which to consider yelp reviews for predicting restaurant grades. After having observed the relative frequency of recorded health inspections, we determined it would be best to look at health grades in monthly intervals. Without an interval, there would have been no appropriate way to link Yelp reviews to health inspection results. To overcome the challenge of multiple inspections per month, we decided to take the latest awarded grade in each interval. We accomplished this by overwriting **inspection date** such that each date consisted of only month and year. Regardless of what day of the month an inspection occurred, we rewrote the date as the first of said month. Then, using **inspection time**,

which is a timestamp object describing the date and time of inspection, we were able to extract the most recent restaurant grade for that month.

Given that there were multiple Yelp reviews for each restaurant at any given month, we had several features which we needed to combine—**review_rating, funny, cool, useful,** and **text.** For all of the aforementioned features, except **text**, we took the average. We combined **text** by concatenating Yelp reviews for each month per restaurant. In hopes of identifying words associated with different healthgrades, we built a Latent Dirichlet Allocation (LDA) topic model. To avoid overfitting, we exposed our LDA model to documents only in our training set. In this way, our topic model would have a chance to encounter previously unseen words, thus mirroring the real world scenario of deploying this model. We then used our LDA model to create the numeric feature **topics**--an integer between 0 and 9 that was assigned based on which topic dominated each Yelp review.

4.3 Data Cleaning

Since both of the datasets we worked with were pretty structured datasets, we did not have to clean the data too much.

Dealing with Missing data

The health inspection data did not contain any missing data. The Yelp dataset did have a lot of missing data in certain attributes. However, we did not use any of these attributes so it did not matter.

The major source of missing data was when we aggregated the reviews by month per restaurant. Some restaurants did not have reviews for all months. Thus, we could not calculate the averages of **review_rating, funny, cool** and **useful** for those months. To deal with this, we populated the missing

values with the mean value of the corresponding feature. We used this approach, as using dummy variables for missing variables proved less effective and we did not have enough data to build sub-predictors for predicting the missing values.

Also when there were no reviews in a month, the **text** column was empty, so we could not generate **topics** for this month. We assigned a default topic number **10** to all such instances. Essentially making **10** a dummy value to indicate that **topics** is missing.

String Cleaning

To build our LDA model, we had to preprocess the strings we pass. We lemmatized the words passed to their root forms. We removed stopwords and any word less than 3 characters in length.

4.4 Feature Transformations

Feature Scaling

The models we tried are DT, LR and RF. DT and RF do not require scaled features as they are not distance based models. They work on information gain/ gini index. Both are metrics which do not change with scale of features. LR, though a distance based model, has the characteristic ability to scale the weights correspondingly to the scale of the features to ensure that highly scaled features do not dominate the predictions.

One-Hot Encoding

Since some of our features are categorical, a natural way to encode those features is One-Hot Encoding. As we cannot pass categorical features as strings to our models, we factorized them into numbers and passed them to our models. Example: Our target variable **category_name** which could take values such as ‘Restaurant’, ‘Bar’, ‘Tavern’, etc. was factored into ‘0’, ‘1’, ‘2’, etc.. By converting the strings into

numbers, we were able to pass it to our model. A disadvantage of this method is that, numerical feature would imply that there is an order/rank in the values i.e. 1 is better 0 because 1 > 0, although that is not the case. To overcome this, we decided to one-hot encode these values as vectors. Pandas gives an easy way to one-hot encode the features via the get_dummies function. We used it as a preprocessing step to test whether One-Hot encoding improves the performance of our model.

5. Modeling & Evaluation

5.1 Choices of Data Mining Algorithm

Our task at hand was a multi label classification problem. We had a total of three classes to predict. With this in mind we decided to explore data with the following algorithms:

- **Multinomial Logistic Regression:** The main advantage of using Logistic Regression is the fact that it is a robust model which works on small datasets. Thereby, it is well suited as a baseline model for an exploratory project of this nature. Though, the logistic regression biggest disadvantage is also its simplicity. It is not able to account for complicated relationships between features. It assumes a linear relationship between features and also independence between the observations. In many cases, this might not be true. In our case, from our knowledge of the data collection mechanisms we assume samples are independent. Nevertheless, since we did not collect the data ourselves we cannot be sure of the independence of the samples.
- **Decision Trees:** Decision tree classifiers were a natural choice for our problem because they were useful for ranking features and to get a sense of feature importance, which is crucial in an exploratory project of this nature. However, as decision trees tend to overfit training data, we tried our best to reduce overfitting by parameter tuning with grid search and cross validation.

- **Random Forests:** After using Decision Trees, a natural improvement is to test Random Forests. From a bias variance perspective, the decision tree has low bias but high variance. Running multiple decision trees on the bootstrap samples of the same data and aggregating the results reduces the total variance of the model and acts as a regularization mechanism. This is why we were motivated to use the same in our exploration.

We chose the three models mentioned above because of the nature of our problem. We did not have a lot of data to deal with. We were exploring the possibility of making good predictions.

Complex alternatives such as Neural Networks and Boltzmann Machines were not tried because they require much more data to train on for effective results. Support Vector Machines and Gradient Boosting Trees are other algorithms which we would have tried, however we were unsure whether they would be significantly better than the algorithms we tried. Given more time, we would have tested those algorithms as well.

5.2 Baseline Model and Its Performance

The baseline model consisted of two basic classifiers:

(1) A multinomial logistic regression classifier with default parameters.

(2) A decision tree classifier with 'entropy' as criterion and default parameters.

For these basic classifiers only features from the restaurant's past inspection records were used. The initial list of variables included: 'Category name', 'Zip', 'Inspection Type', 'Permit Status'. Many variables were initially discarded because they were not providing any insight on the target variable (i.e., Business ID, Employee ID, Inspection Time, etc). Moreover, during feature selection some of them were

discarded because they showed a direct relationship with our target variable 'Inspection Grade': 'Current Demerits', 'Inspection Results', 'Current Grades'.

All categorical variables where transformed into numeric with pandas function factorize to be able to feed them to the logistic regression model. The test/training split was made in a 0.25/0.75 proportion respectively and accuracy was measured as initial metric.

Identifying Data Leakage

The initial results showed almost perfect accuracy which aroused our suspicion. Thus, in order to identify whether data leakage was present in our model, we incorporated a feature importance analysis with our decision tree classifier. This analysis allowed us to identify that the variable 'Permit status' was a really good predictor of our target variable, in fact, the values contained in 'Permit Status' were almost a duplicate of our target variable but with some missing values. Therefore, we identified this as data leakage and proceeded to drop it from the feature list.

Dealing with Imbalanced Samples

As mentioned previously, an important characteristic of our data was the innate imbalance in the class labels. The distribution of grades across the restaurants was highly skewed (88% for 'A', 5% for 'B' and 7% for 'C'). To deal with this, some of the approaches we tried were:

1. Upsampling the classes which had lesser number of samples
2. Downsampling the class with more samples
3. Giving different class weights to the sample

Our original training data distribution is given in Fig. 6. We tried upsampling to various degrees and the best distribution is given in Fig. 7

Training Data Counts

Class Label	Number of Records
A	1102
B	82
C	62
Total	1246

Figure 6

Balanced Training Data Counts

Class Label	Number of Records
A	1102
B	1102
C	1102
Total	3306

Figure 7

Testing Data Counts

Class Label	Number of Records
A	369
B	23
C	21
Total	413

Figure 8

Our testing distribution is given in Fig. 8. We did not upsample/downsample the testing distribution because it has to reflect the original data distribution.

Evaluation Metric

Due to the imbalanced nature of the data, accuracy is not an effective evaluation metric. If our model just predicts class 'A' every time, we would get an accuracy of 89%. But it does not mean the classifier has learned anything useful.

We had to resort to using other classification metrics such as precision, recall or f1-score to evaluate the performance of our model. We chose **precision** over other metrics in particular because we wanted to penalize false positives more (Type 1 error). This decision was taken considering our domain and type of problem. It is more dangerous to call an unhealthy restaurant as healthy ('A') than calling a healthy restaurant unhealthy ('B', 'C'). Precision signifies how many restaurants we label as healthy ('A') are actually healthy and vice versa. Thus, we decided to optimize all our models to increase precision over recall/f1-score.

Model Selection

Grid Search

A key mechanism to ensure we have selected optimal parameter configuration is Grid Search. We implemented the same using the sklearn package to ensure we tested across a wide variety of parameters. For GridSearch with Decision Trees, the main parameters are **min_split_size** and **min_leaf_size.** **min_leaf_size** is especially important because it can be used to control overfitting in Decision Trees. **min_leaf_size = 1** would imply the Decision Tree has just memorized the samples. Very low values of **min_leaf_size** indicate overfitting. We varied it from (5, 15). We varied **min_split_size** from (10,30). For Logistic Regression, the only parameter we searched was '**C**' from (10^{-3}, 10^{3}). In addition to GridSearch, we also used the sklearn pipeline to introduce scaling and one-hot encoding as preprocessing steps to see if they would improve performance. However, both methods did not significantly alter the performance. From the Grid Search, for Decision Trees, the best parameter configuration was **min_leaf_size** = **5** and **min_split_size =10.** For Logistic Regression, the best C was 1.0 which was the default value.

Cross Validation

Cross validation is a great mechanism to improve performance of models in cases of having low training/testing data. By splitting the data into multiple train/test combinations, cross validation ensures the model has been trained on almost all the samples and tested equally well. Interestingly, with the configurations from Grid Search, cross validation did not improve the performance in our case. Cross validation with 4 folds was just as effective doing a train/test split of 75/25. The number of folds was determined to be 4 by empirically testing multiple fold values. We can only surmise from this that our model does not have a high variance. Cross validation is a method to lower variance of a low bias/ high variance model. However, since our model did not have much variance to begin with, cross validation did not result in a substantial improvement.

Results

After running a decision tree and a logistic regression on our training set with the best parameters, we obtained the following results. Fig. 9 shows the feature importances obtained from the built decision tree. Fig. 10. shows the classification metrics and also the confusion matrix of the decision tree. Fig 11. shows the classification metrics and the confusion matrix of the multinomial logistic regression.

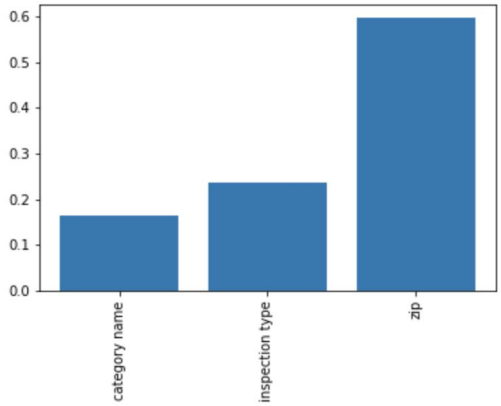


Figure 9. Feature Importance on Baseline Model

	precision	recall	f1-score	support
0	0.35	0.96	0.51	133
1	0.30	0.06	0.10	113
2	0.48	0.06	0.11	167
avg / total	0.39	0.35	0.24	413

```
[[128   3   2]
 [ 97   7   9]
 [144  13  10]]
```

Figure 10. Decision Tree Baseline Model Confusion Matrix Result

	precision	recall	f1-score	support
0	0.00	0.00	0.00	0
1	1.00	0.06	0.11	413
2	0.00	0.00	0.00	0
avg / total	1.00	0.06	0.11	413

```
[[  0   0   0]
 [369  23  21]
 [  0   0   0]]
```

Figure 11. Logistic Regression Baseline Model Confusion Matrix Result

As observed (Fig. 10) the baseline decision tree averages around 0.35 precision for all the classes. The logistic regression essentially predicts '1' always.

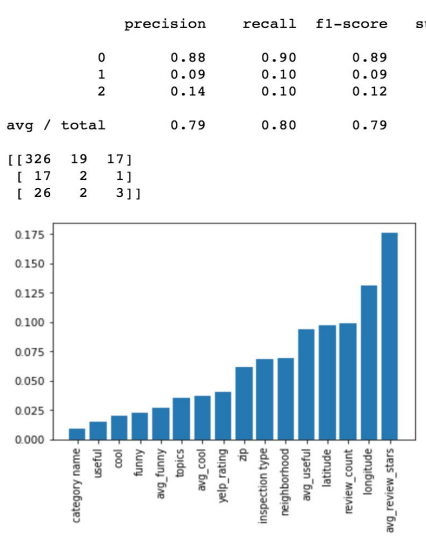
Intuitively, we can see that the decision tree is barely better than random guessing. The logistic regression has a better accuracy than random guessing due to sample skew however, it has not learned anything useful.

Keeping these as the baseline models, we would like to investigate whether adding more features from Yelp data would improve precision, recall and overall accuracy preferred in that order.

5.3 Incremental Models

We built our incremental models over our baseline models by including more informative features from Yelp Data like **'latitude', 'longitude', 'neighborhood', 'review_count', 'yelp_rating'**. We also included the new features we engineered by aggregating individual reviews per month per restaurant such as **'avg_review_stars', 'avg_cool', 'avg_funny', 'avg_useful', 'topics'**.

We built a new decision tree augmented these features but with the same previous parameters. Our new Decision Tree model was successful in predicting the dominant class-- health grade 'A'. Below is the performance as described by the classification metrics and corresponding feature importance.



	precision	recall	f1-score	support
0	0.88	0.90	0.89	362
1	0.09	0.10	0.09	20
2	0.14	0.10	0.12	31
avg / total	0.79	0.80	0.79	413

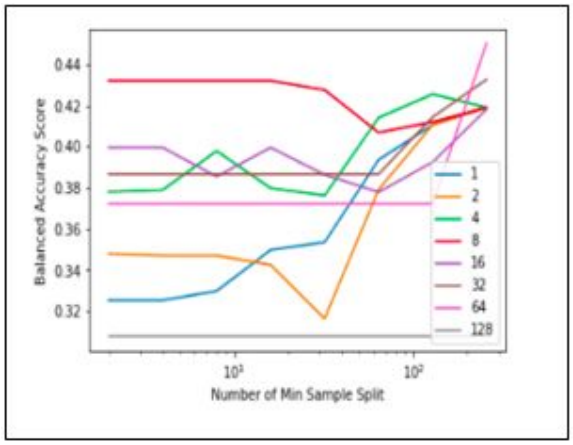
```
[[326  19  17]
 [ 17   2   1]
 [ 26   2   3]]
```


	feature_importance
category name	0.009346
useful	0.014726
cool	0.019895
funny	0.022094
avg_funny	0.026590
topics	0.035331
avg_cool	0.036600
yelp_rating	0.040103
zip	0.061497
inspection type	0.067919
neighborhood	0.069444
avg_useful	0.093805
latitude	0.097107
review_count	0.098519
longitude	0.131450
avg_review_stars	0.175572

Figure 9. Decision Tree and Feature Importance

From the confusion matrix, we can observe that our Decision Tree is not just predicting 'A' always and has high precision / recall. However, when compared to our baseline model, it loses out on performance in predicting 'B' and 'C'. One might argue the high precision and recall are due to huge number of 'A' samples in the test set.

Considering that, to improve our model, we tried a fresh GridSearch [**min_sample_split_size** from (2^1,2^8) and **min_sample_leaf_size** from (2^1,2^8)]. We used the Balanced Accuracy Score metric in order to address the inflated performance estimates (for health grade 'A') on our imbalanced datasets. As seen below, we obtained the **min_sample_leaf_size** = 8 and **min_sample_split_size** = 2, which resulted in the best Balanced Accuracy Score. With the updated hyperparameters, the precision for 'A' decreased from 0.88 to 0.68, but it improved for 'B' and 'C' (from 0.09 to 0.39 and from 0.14 to 0.19, respectively). As a result, we improved the precision for B and C at the cost of A.



	precision	recall	f1-score	support
0	0.68	0.91	0.78	275
1	0.39	0.12	0.19	73
2	0.19	0.06	0.09	65
micro avg	0.64	0.64	0.64	413
macro avg	0.42	0.36	0.35	413
weighted avg	0.55	0.64	0.56	413

```
[[250  10  15]
 [ 62   9   2]
 [ 57   4   4]]
0.6368038740920097
```

Figure 11. Balanced Accuracy Score and Number of Min Sample leaf

We trained a Logistic Regression model with the new features which was better able to identify minority classes. While fitting the LR model, different 'solver' parameters were tested to use in the optimization problem including 'newton-cg', 'sag' (Stochastic Average Gradient), 'lbfgs' (Limited-memory BFGS), all yielding similar results.

```
                precision    recall  f1-score   support

             0       0.04      1.00      0.07        14
             1       0.70      0.06      0.12       255
             2       0.57      0.08      0.15       144

avg / total       0.63      0.10      0.12       413

[[ 14   0   0]
 [230  16   9]
 [125   7  12]]
```

Figure 10. Logistic Regression

From the precision scores for 'B' and 'C' we can observe that the LR model learns to predict 'B' and 'C' well but not 'A'. We have seen that Decision Tree works better for health grade 'A'.

Although neither model can identify every class to a high degree, considering the business aspects of the problem we are hoping to solve, we would choose Logistic Regression as model to deploy in production. In the context of restaurant grades/health inspections it is more costly to miscategorize classes B, C as A than the converse. In view of that, our choice is justified by the precision scores for B, C shown above. Moreover, it might be worthwhile to consider deploying an ensemble model where we split A vs. B/C. We use the DT model to decide whether it is A/ not A. Then use the LR model to decide B/C if not A. As an alternative to the aforementioned, we could build a regression model that returns probabilities of class membership as opposed to class labels themselves. The probability of belonging to B, C classes could then be utilized as risk levels. Users, in turn, decide their threshold for risk and act accordingly. This opens up the possibility for different models, such as Generalized Additive Models, which allow us to maximize the overall fit of the model by minimizing the overall likelihood of the data given the model.

We also tried using Random Forests. We did a grid search across the parameters **min_leaf_size, min_split_size** and **n_estimators.** The Random Forest resulted in the same performance as Decision Tree. This can be attributed to two reasons

1. From our feature distribution from Decision Trees we can observe that some features dominate. This leads to poorly performing Random Forests.
2. Our model does not have high variance and generalizes well. Random Forests try to lower the high variance in Decision Trees. However, if our DT model did not have high variance to begin with, Random Forests would not increase the performance much.

6. Deployment

6.1 Deployment Use Cases

1. **Customers are Restaurant owners**: We can develop an online membership-based service. A restaurant owner can create a profile for his/her restaurant and he/she can view a daily snapshot, a monthly report with more details such as anonymized statistics of peer restaurants in terms of a similar type of cuisine, neighborhood, etc. to see how his/her restaurant compares against other restaurants. The restaurant owner can also receive an alert when a predefined threshold has been exceeded (e.g., predicted number of violations for the upcoming inspection exceeds 10, etc.).
2. **Customers are Health Inspectors**: On a monthly, etc. basis, government health inspectors can use the prioritized list of restaurants in order of potential health inspection grades as a supplementary guide to their inspection schedules.

6.2 Monitoring and Evaluation in Production

Once deployed to production, on a monthly basis, we will collect new Yelp data and health inspection data, and predict the inspection grades. After two/three months, when we get the actual inspection grades, we will compare our predictions to the actual values to evaluate the model. If the performance is significantly different from what is expected then further analysis has to be done to ensure elements like seasonal variation, etc. have not resulted in Concept Drift. If not, using more data further investigation has

to be done to find any underlying patterns of weakness in the model. If determined that some system issues caused deviances from expected behavior, care has to be taken to fix the system issue without affecting the existing system. Furthermore, to ensure we have the right predictions, a rough estimate could be plotting the probability distribution of the predictions made in live environment and see if it matches the probability distribution of our training/testing data. If large deviations begin to occur, then it can be deduced that there is something wrong with the model.

6.3 Potential Ethical Considerations

We have seen that the feature importance graph ranked longitude/latitude and neighborhood among the top. Therefore these two features are informative when predicting the future inspection grades. An analysis could be done to see if our models show that restaurants in low-income neighborhoods tend to get lower inspection grades. Also, we could include new features from Yelp data such as cuisine and establishment owner information to check if there are any unfair biases against specific cuisines/ owners. This would imply these establishment receive lower grade predictions and hence more frequent inspections by government agencies. This is unethical as cuisine/ owner information/ location should ideally not affect the prediction of health grades.

6.4 Associated Risks and Mitigation Plan

Yelp does not allow any "scraping" of the site. In other words, Yelp does not allow taking content for display or sale. It's also not permissible to record, process, or mine information about users. This means, our ability to improve the models is limited to the amount of dataset available through Yelp Dataset Challenge data.

Another challenge could come from quality of user comments. There have been many cases where competitors left fake negative reviews as one can be anonymous on Yelp. On the flip side, business

owners can also hire someone to leave positive reviews to boost their own ratings. These artificially inflated negative or positive reviews can affect the accuracy of the features utilized in our models.

In an attempt to mitigate both risks, we can develop a model to identify expected distributions in a dataset and detect fake reviews. Since Yelp is motivated to keep its services free of fake reviews and false data, we could establish a business deal with Yelp. We can inform Yelp of our findings and in return get permission to scrape data we need. Alternatively, we can try recreating our models using different restaurant/ social media data sources, such as Trip Advisors, Grubhub, Seamless, Instagram, etc.

7. References

[1] Ginger Zhe Jin and Phillip Leslie, 2005. The Case in Support of Restaurant Hygiene Grade Cards.

[2] Eric Goldman, 2014. Does Yelp Have The 'Most Trusted Reviews'? A Court Wants To Know More.

8. Future Work

Two possible avenues for further exploration are:

Combining data from multiple cities

We used data for only Las Vegas restaurants, similar data is available for many cities across the US. Combining data from these cities and training a model is potentially a great way to improve the model and the usefulness of the model. Additional data helps the model generalize better. Furthermore, with a large dataset, we can also think about building sub-predictors to predict missing data in the dataset.

Using more attributes from Yelp data

We used only a subset of the most important and relevant attributes from Yelp data. Exploring the effects of other attributes is also a viable option to test.

9. Contributions

- **Antonio** - data preparation (cleaning and processing Yelp/LV health grade into dataframes), LDA model; contributed to writing S2,S4, & generating feature importance plots
- **Laureano** - Helped identifying data leakage and with the study of feature importance. Modeling Multinomial Logistic Regression. Contributed to writing sections 2 and 5.
- **Sammie** - Section 6 writing. Modeling of Decision Tree using GridSearch, Contributed to writing sections 1, 2, and 5.
- **Shreyas** - Helped with data aggregation, feature engineering and transformation, Random Forests, and model selection while developing the code. Contributed to writing sections 2,4,5 & 8 editing and formatting of the entire report.