New York City Restaurant Inspection Analysis and Forecasting

Jialing Xu(jx1047), Lin Jiang(lj1194), Rui Jiang(rj1407), Yuxuan He(yh2857)
Team Expecto Patronum, Center for Data Science, New York University

Abstract: In this paper, we propose the complete data science pipeline on tackling restaurant inspection prediction problem. We aim to predict the likelihood of failing health inspection for a restaurant in order to offer insight to NYC health department. This project contains five parts, involving Business understanding, Data understanding, Data Preparation and Analysis, Model selection and Evaluation and Deployment. The models we selected include Logistic Regression, Random Forest, and LightGBM. The results show that the lightgbm outperforms all the rest machine learning models.

1 Business Understanding

Each year, the New York City Health Department randomly inspects over 30,000 restaurants multiple times in an effort to improve the quality and safety of restaurants. These inspections resulted in over 100,000 violations ranging from employees not washing their hands to rodent infestation. Each violation is assigned a certain number of points, which then is calculated to form an inspection score indicating the restaurant's overall inspection score. However, such project consumes a huge amount of resources in terms of manpower, money, and materials. Thus, we started thinking: can we take advantage of the data science techniques and abundant restaurant data on the internet to simplify and improve this process?

To be more specific, in this project, we aim to design a prediction model based on publicly available restaurant data to forecasting the potential probability for a restaurant instance to fail the inspection. The primary usage scenario of the model is to generate insight for the New York City Department of Health and Mental Hygiene. Instead of randomly performing inspection, the Department of Health and Mental Hygiene can perform the inspection to restaurants with high potential of failing. A secondary usage scenario is to offer insight to restaurant owners: whether they, as the manager of the restaurants, should make some improvements for its environment to avoid failing the inspection test. Moreover, this model could then be applied to other cities in the world and improve the restaurant inspection efficiency on a greater scale.

2 Data Understanding

2.1 Collecting Data

To accurately predict restaurant inspection failures, after careful discussion, we decided to collect data in the following four aspects including past inspection results data, restaurant data, location data and time data.

The primary resource we used is 'DOHMH New York City Restaurant Inspection Results' dataset. It contains every violation citation from every inspection conducted up to three years prior to the most recent inspection for restaurants. Since the data is indexed by one restaurant per inspection per violation, we aggregated the data by combining the violations for each restaurant during each inspection and the instances look like the following. Very intuitively, we choose to use 'Critical Flag' variable as an indicator to generate the target variable. (If Critical Flag is greater than a threshold, we set the Target variable to 1, else set it to 0). It is because 'Critical Flag' indicates the number of critical violations such as those contributed to food-borne illness that this restaurant committed in the current inspection.

CAMIS	DATE	DBA	ZIPC ODE	CUISINE DESCRIPTION	ACTION	VIOLATION CODE	VIOLATION DESCRIPTION	CRITICAL FLAG	SCORE	GRADE
30075445	2016-02-18	MORRIS PARK BAKE SHOP	10462	Bakery	Violations were cited in the following area(s).Violations were cited in the following area(s).	04L08A	Evidence of mice or live mice present in facility's food and/or non-food areas.Facility not vermin proof. Harborage or conditions conducive to attracting vermin to the premises and/or allowing vermin to exist.	1	10	A
	2017-05-18	MORRIS PARK BAKE SHOP	10462	Bakery	Violations were cited in the following area(s).Violations were cited in the following area(s).	06D10F	Food contact surface not properly washed, rinsed and sanitized after each use and following any activity when contamination may have occurred.Non-food contact surface improperly constructed. Unacceptable material used. Non-food contact surface or equipment improperly maintained and/or not properly sealed, raised, spaced or movable to allow accessibility for cleaning on all sides, above and underneath the unit.	1	7	A
	2018-05-11	MORRIS PARK BAKE SHOP	10462	Bakery	Violations were cited in the following area(s).Violations were cited in the following area(s).	10F08C	Non-food contact surface improperly constructed. Unacceptable material used. Non-food contact surface or equipment improperly maintained and/or not properly sealed, raised, spaced or movable to allow accessibility for cleaning on all sides, above and underneath the unit.Pesticide use not in accordance with label or applicable laws. Prohibited chemical used/stored. Open bait station used.	1	5	A
30112340	2015-05-07	WENDY'S	11225	Hamburgers	Violations were cited in the following area(s).Violations were cited in the following area(s).	10B04A	Plumbing not properly installed or maintained; anti-siphonage or backflow prevention device not provided where required; equipment or floor not properly drained; sewage disposal system in disrepair or not functioning properly.Food Protection Certificate not held by supervisor of food operations.	1	12	A
	2016-04-12	WENDY'S	11225	Hamburgers	No violations were recorded at the time of this inspection.		N/A	0	0	N/A

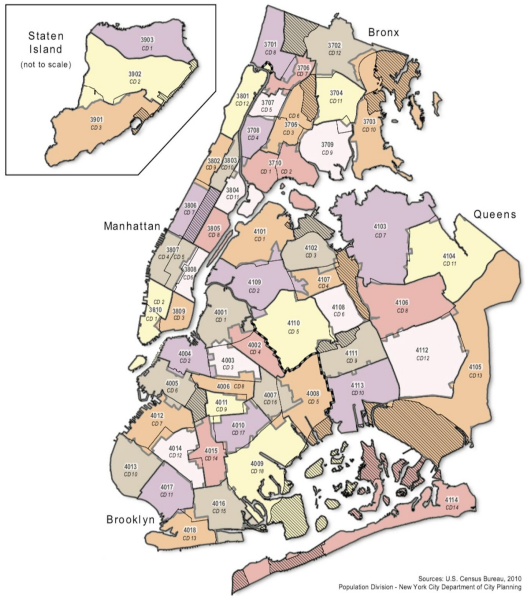
Table 1: Inspection Dataset Overview

The secondary data is restaurants data which we generated using Yelp Fusion API and Google Map API. For each unique CAMIS value in the inspection table above, we extract its DBA (name of the restaurant) and Address as parameters and send HTTP request to Yelp Fusion API's business Match endpoint to get BusinessID. Then we use BusinessID as parameters to query its corresponding reviews and business details. Similarly, we send HTTP requests to Google Map API to retrieve restaurant reviews given each restaurant's name and location. The

features we extracted includes yelp_price, yelp_rating, yelp_review_counts, latitude, longitude, opening_days, combined_reviews.

CAMIS	yelp_price	yelp_rating	review_count	latitude	longitude	day0_open	day1_open	day2_open	day3_open	day4_open	day5_open	day6_open	combined_reviews
40362098	2	3	138	40.787	-73.976	1	1	1	1	1	1	0	Delicious burgers New York style one coolest owners around You want taste city Then head Harriets Don share Delivery disaster Called food min late Assured would moments I live within blocks minutes This place shut health department I lived area since always passed place put handwrittenWe ordered
40362274	2	3.5	459	40.726	-73.997	1	1	1	1	1	0	0	The Angelika I go way back In fact way back early days I daresay pretty common run indie celebrities Vincent Gallo say I theater showing Free Solo I like cinemas mainly showing independent films I know I would return reasons I love old vintage vibe theater managed hold onto years No updated reclining
40363098	1	2	11	40.692	-73.991	1	1	1	1	0	1	1	I love hate relationship place Over I came know dishes really worth highway robbery style pricing whether My aunt lives th Street Lexington Avenue feeling well lately I promised dinner I cook anything As year Yelp Elite reviews I must begin I never heard restaurant half whose reviews refer ownerSimply
40362715	1	2.5	20	40.706	-74.009	1	1	1	1	1	1	1	I walked past place like hundred times Decided stop lunch I preordered salad chicken mozz They ready go Cute signage suggests cozy little place grab salad coffee maybe sandwich Then sit one tables huge public space The Country Cafe pretty bare bones It basically small corner deli located
40363744	NaN	5	41	40.643	-73.907	1	1	1	1	1	1	1	I craving sandwich coworker swore Sonny thee place get I Canarsie area yrs Ding dong Doorbell rings wan na take walk Sonnys nThis childhood Taking walk Sonny hero delicious experience This best place Canarsie Brooklyn get hero sandwich made boars head cold cuts made pride live

Table 2: Yelp Data Overview (fusion API)



The third portion of data contains mainly geological information. NYU Furman Center offers rich Neighborhood Data Profiles including crime rate, population, unemployment rate, average income and racial diversity index for each neighborhood. Thus, we write code to map each restaurant to its belonging neighborhood base on zip code and geographic coordinates and then merged Neighborhood Data Profiles to each restaurant instance. And using similar methodology, we mapped rodent complaints data from NYC OpenData - Rodent Complaints in NYC to each restaurant instance since rodent complaints are highly correlated to restaurant inspection failures.

Last but not least, inspection time provided critical information to the prediction, too. On the one hand, for each restaurant instance, we append its historical inspection data including last inspection time, score, number of violations etc in history as features. On the other hand, since food is in general sensitive to temperature and humidity, we incorporated local temperature and humidity for inspection day as features, too. These data are generated from NOAA - Local Climatological Data.

2.2 Target Distribution

One significant problem we discovered is that the dataset is imbalanced. We have in total 132147 data instance and only 17987 data instance has 'Critical Flag' variable greater than one. That is, if we choose to set target variable to represent whether a restaurant has committed at least critical violation in the inspection as a target variable, then only 13.6% of the data would be label as 0.

The initial approach we choose to deal with this issue is to relax our target variable threshold. We chose to set the target variable to represent whether 'Critical Flag' variable is greater than 1, which means whether the restaurant has committed multiple critical violations. This makes sense because we do want to predict the higher risk restaurants. In this way, we are able to shift our target variable to a 7:3 ratio.

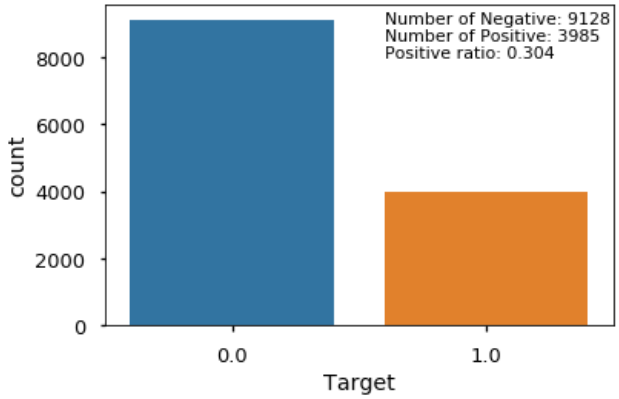


Figure 1: Distribution of Target Variable

2.3 Selection bias

According to the described inspection process from NYC government website, every restaurant in New York City is scheduled for an unannounced inspection at least once a year. So theoretically, our initial inspection dataset should cover the whole restaurant population. However, after we merged inspection data with yelp data, we dropped rows which we could not gather enough restaurant information. For those dropped rows, we performed statistical tests and discovered that distribution of critical flag is approximately the same as that in the original datasets. Hence, we believe that the selection bias is quite limited.

3 Data Preparation and Analysis

3.1 Data Processing and Feature Engineering

In the overall inspection dataset, we clean the data by deleting mismatched or unfound records. We then replace NaN with zero or mean based on the variable nature and added indicator variables indicating whether zero has been set to any of its row and. We also perform feature engineering and binning based on the domain knowledge of the data and generated some new features which are nonlinear functions of the existing ones. Our domain knowledge indicates the resulting feature will be meaningful and informative.

Source	Existing Features	New Features
DOHMN	**Open**: The detailed opening hours of each day in a week.	**DaysSinceLastInsp**: days elapsed between the last two inspections **TotalOpenDays**: days elapsed between the first and last inspections **InitialInsp**: 1 if the restaurant has no previous inspection, 0 otherwise
	CriticalFlag: Indicator of critical violation **InspectionDate**: the date of inspection	**HistCrit**: total number of historical critical violations **NoHistCrit**: 1 if there is no previous critical violation, 0 otherwise **CritRate**: historical critical violation times over total inspection times **LastCritOverLastInsp**: days elapsed since last critical violation over days elapsed since last inspection
	Zipcode: Zip code of restaurant location	**SubBoro**: mapped to corresponding sub-borough areas
	Cuisine: cuisine type associated with this restaurant.	**FoodType**: re-grouped to smaller number of categories
YELP	**Open**: The detailed opening hours of each day in a week.	**WeeklyOpenHr**: total opening hours in a week **WeeklyOpenDays**: total opening days in a week **WeeklyOvernightTimes**: number of times a business opens overnight
	Price: Price level of the restaurant **Rating**: Rating for this restaurant **ReviewCount**: Number of reviews for this restaurant.	**PriceOverRating**: price-rating ratio **RatingOverPrice**: rating-price ratio **RatingCountProd**: product of restaurant rating and review count
NOAA	**HourlyTempF**: hourly temperature (degrees Fahrenheit) **HourlyRelativeHumidity**: relative humidity (percent)	**AveTemp3dayMax**: max of average daily temperature in latest 3 days **AveTemp3dayAve**: average of average daily temperature in latest 3 days **AveTemp3dayRange**: range of average daily temperature in latest 3 days **MaxTemp3dayMax**: max of highest daily temperature in latest 3 days **MaxTemp3dayAve**: average of highest daily temperature in latest 3 days **MaxTemp3dayRange**: range of highest daily temperature in latest 3 days **AveHum3dayMax**: max of average daily humidity in latest 3 days **AveHum3dayAve**: max of average daily humidity in latest 3 days **AveHum3dayRange**: range of average daily humidity in latest 3 days
NYC Open Data	**ComplaintDescriptor**: sighting of rats or mice, as well as conditions that might attract rodents. **IncidentZip**: zip code of incident location **CreatedDate**: date and time of the complaint .	**RatSighting**: number of rat sighting complaints per zip code **MouseSighting**: number of mouse sighting complaints per zip code **CondAttractRodents**: number of conditions attracting rodents complaints per zip code **RodentSigns**: number of signs of rodents complaints per zip code **RodentScore**: weighted average of RatSighting, MouseSighting, CondAttractRodents and RodentSigns

Table 3: Feature Engineering Summary

3.2 Descriptive Data Analysis and Data Visualization

We made several analysis plots on the data and below are the ones that generate insightful information. For example, Figure 2 shows the positive/total ratio for restaurants within each food type. The outlier score for Brazilian and Turkish may be caused by a small number of samples. No food type shows significantly higher relative possibility of being in positive class than others.

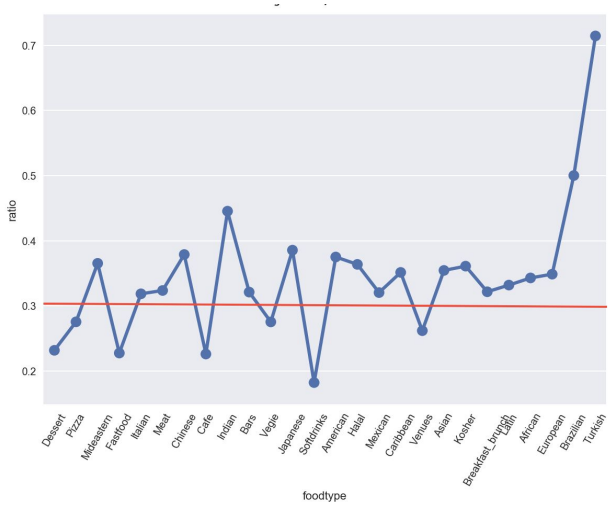


Figure 2: Positive Ratio Across Food Type

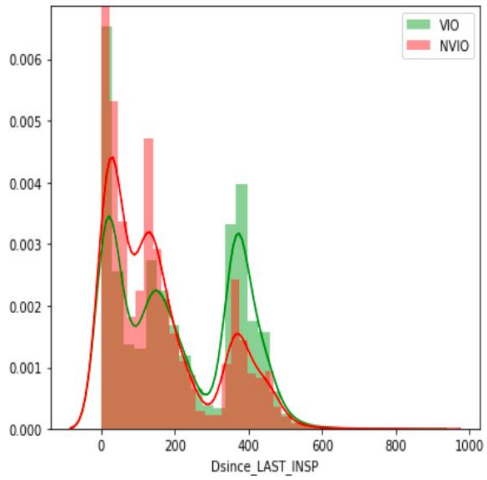


Figure 3: Day Since Last Inspection Distribution

Figure 3 shows high volume of non-violation instance with small Dsince_Last_INSP and high volume of violation instance with high Dsince_Last_INSP. It makes sense since if a restaurant has been inspected recently, it would probably pass another inspection in the near future. The longer the time that a restaurant has not been checked, the higher probability that it will fail the inspection since the restaurant may lose its hygiene standards as days goes by. And another large part of critical restaurants inspected frequently are indeed restaurants with high internal risk to have violations, which partly justifies for DOHMH's current inspection cycle, though it may be better if they shorten the period more.

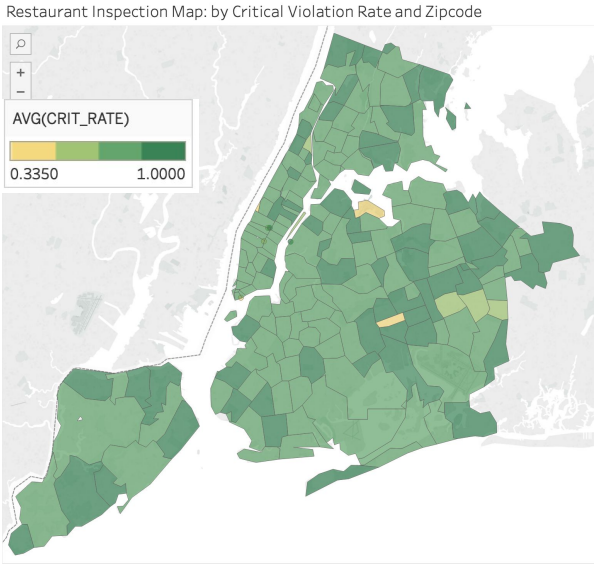


Figure 4: Average Critical Rate Of Zip Code Area

Furthermore, Figure 4 shows a visualization of geographically display the distribution of restaurant inspection with Tableau. In the figure, we can see that central Queens and outer area of New York City seems to have higher critical rate.

Last but not the least, by looking at our final feature importance lists (Figure 10,11), top factors behind critical violations are their historical performance, Yelp reflection , weather conditions and goelocal influence like rodents, median income etc.

4 Modeling and Evaluation

4.1 Empirical Set Up and Baseline Model

Since our data has skewed class distribution of 30% (positive), we chose AUC as our evaluation metrics since it is invariant to base rate. We used random sampling to split data into training and testing (80/20) sets. With a relatively small data size we expected high variance, so we need to use regularization to control model complexity and avoid overfitting.

We used three raw features, LastGrade, LastAction and LastScore from the DOHMH inspection results and apply one hot encoding to build the baseline logistic regression model. The result AUC score of 0.61 which successfully beats the random classifier. We chose Logistic Regression because it is robust in small dataset as well as skewed classes. Besides, it has an advantage of returning a probability score.

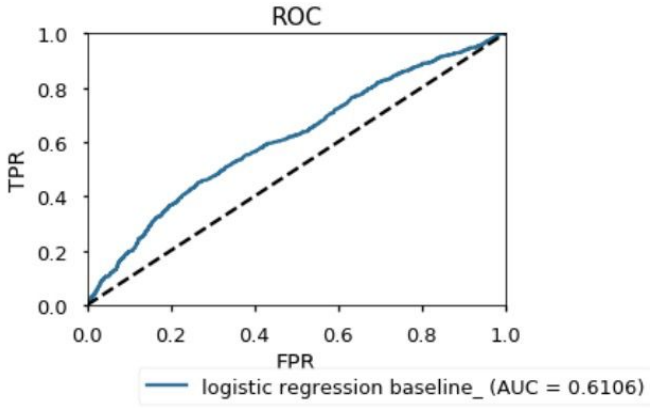


Figure 5: ROC of Logistic Regression (Baseline Model)

We sought to improve upon the baseline model above in the following way:

- **Incorporate more features.** We progressively added evidence from assessments from customers(Yelp), climatological factors(temperature and humidity) and neighborhood influence(rodent,economics, crime rate etc.).
- **Find the best algorithms that fit our problem.** Exploratory analysis shows that there is no linear correlation among features and target variables. Thus we choose to experiments those that can fit nonlinear structures such as tree based methods, neural networks and non-linear support vector machine.
- **Feature Selection.** We experimented several feature selection methods to understand feature importance and extract useful features also to lower model variance.

- **Tuning hyperparameters for each model.** We tried out of bag method for random forest, k folds cross validation to do grid search on multiple variables and finally automatic GridsearchCV in Sklearn.
- **Try different encoding methods.** The state of the art one hot encoding on categorical features with high cardinality like zipcode, cuisine type generate sparse matrix. To improve model efficiency, we test other encoding method that starts to gain popularity among Kagglers called target encoding, which encodes the category using some representation of its relevance to the target.
- **Feature engineering.** Transforming numeric features, rebinning categorical features etc.

4.2 The Algorithms We Used and Why We Use Them

Besides logistic regression, we chose to use random forest and lightgbm. We choose Random Forest because it is simple but produces a great out-of-box performance. Besides, we include a wide range of features from different aspects in our model, some of which may be of little importance. In this case, since RF searches for the best feature among a random subset of features, it results in a wide diversity that generally returns a better model. LightGBM is another tree-based learning algorithms designed to have faster training speed, higher efficiency and better accuracy. Despite its hyper-parameter tuning complexity, we chose to try LightGBM to further improve our prediction accuracy.

4.3 Results

Choosing hyperparameters:

We learned that two crucial parameters for random forest are n_estimators and max_features. So we first used OOB to get a sense of effective range. Later, we used GridSearchCV to tune more parameters, such as max_depth, min_samples_split and min_samples_leaf.Using only the training set, we did grid search for only these two parameters and get mean AUC scores of 5 folds cross validation. From the figure we see max_feature = 0.2 is definitely better while different n_estimators don't give much variance under this metric.The n_estimators chosen by GridSearchCV is 1200.

For LightGBM, num_leaves and min_data_in_leaf are two main parameters to control the model complexity and prevent overfitting. We firsted tuned the key parameters manually, then continue to fine-tune them and other hyper-parameters by using GridSearchCV.

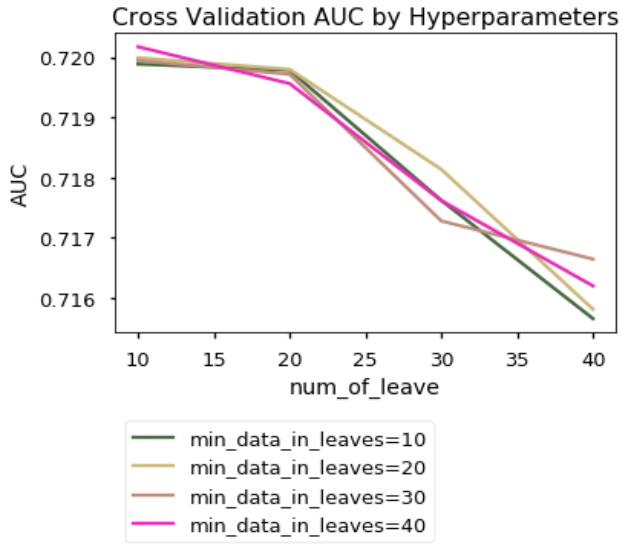


Figure 6: max_feature & n_estimators for RF

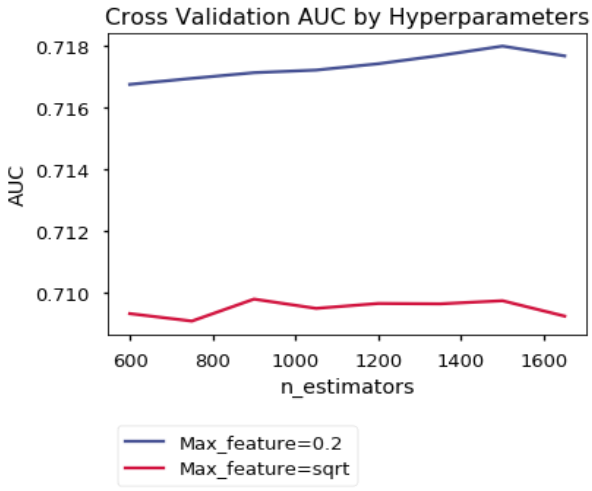


Figure 7: min_data_in_leaves&num_of_leaves for LightGBM

We found that The optimal parameters chosen by the three approaches above are quite close.The final parameters we are using for the three models are:

Logistic Regression: {C = 0.1, penalty = 'l1'}

Random Forest:

{criterion='entropy', bootstrap=True, max_depth=30, n_estimators=1200, max_features='0.2', min_samples_split=20, min_samples_leaf=10}

LightGBM:

{'min_data_in_leaves' = 20, 'boosting_type': 'gbdt', 'objective': 'binary', 'metric': 'binary_logloss', 'bagging_fraction': 0.8, 'bagging_freq': 5, 'n_estimators': 500, 'verbose': 0}

- **Final optimal test set AUC results after all the experiments.**

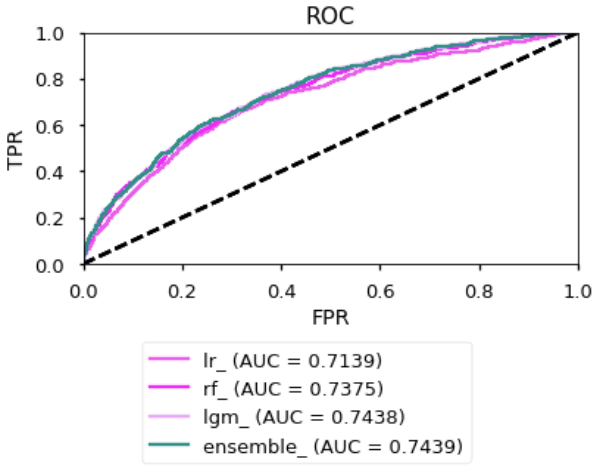


Figure 8: ROC of Best Models

The ensemble model above is a simple one. We take weighted average of the probability prediction of logistic regression, random forest and light GBM with weight (0.2,0.3,0.5). No matter how we adjusted the weight, ensemble this way can not beat lightgbm significantly.

- **Using different feature selection methods may get us slightly different feature sets and feature importance, but they don't impact final accuracy score.**

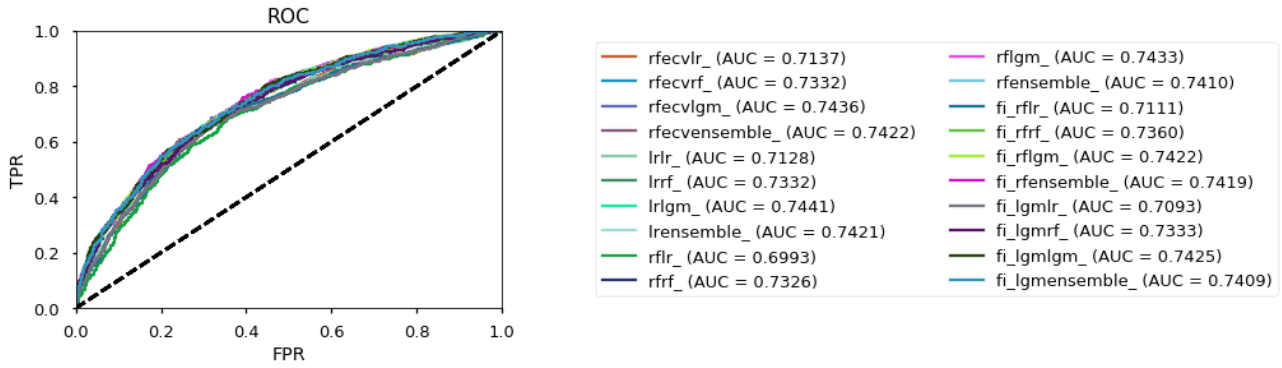


Figure 9: ROC after Using Feature Selection Methods

The feature selection frameworks we tried here are three automated approaches from sklearn: recursive feature elimination and cross-validation, SelectFromModel attributes on logistic regression and random forest as well as two feature importance based methods where we keep features that count for the top 95% of feature importance generated by our random forest and light GBM classifiers. We see that with or without feature selection, no matter what selection approaches we take, AUC scores have no particular distinction. And the order of performance the four models under discussion preserves across feature selection methods.

- **The AUC score of models is quite stable. We are unlikely to overfit**. The three pictures below show AUC score of each fold, the red line being the mean level. We split the training set into 10 folds and ran cross validation. The variance is small for all three models.

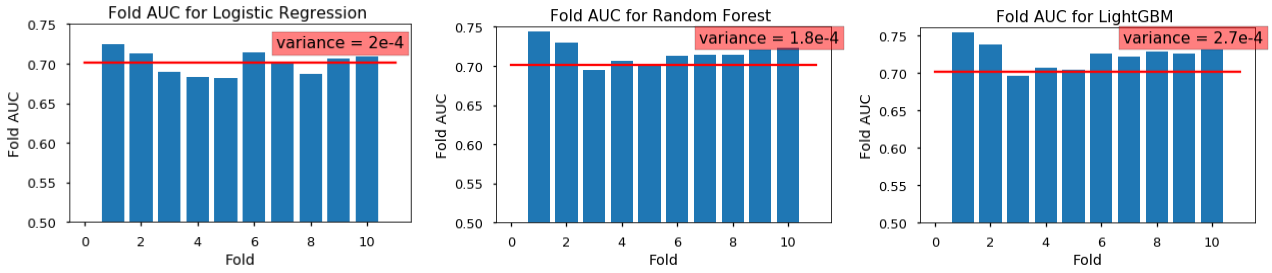


Figure 10: Fold AUC for Cross Validation on LR, RF and LightGBM Models

- **One hot encoding does not have any bad influence for this problem.** In comparing one hot encoding and target encoding, we tried different combinations of encoding methods on multiple categorical features with high cardinality. One hot encoding actually performs slightly better. Note: we carefully avoided data leakage with target encoding. For example, in training data, we mapped each zipcode to the mean target value of this zipcode group. And then we applied the same mapping onto the zip code in test data.

5. Other Attempts

5.1 Text Mining

We requested user review data from yelp fusion API and Google Place API. However, we can only request 3 reviews per restaurant from the former and 5 from the latter. Possibly due to the limited amount of reviews we collected, the result of text mining using TF-IDF was not very informative. It has negligible effect on our model performance.

5.2 Keras Simple Neural Network

This model from Keras library is simple to use and it allows us to build Neural Networks in just a few lines of code. Yet, fine-tuning the hyperparameters is highly challenging. We failed to find hyperparameters that generate better or even average performance.

5.3 SMOTE

Given our dataset has a base rate of 30%, we tried oversampling methods such as bootstrapping and SMOTE. Because binary decision rule generally works best when classes are roughly equal and tree-based model may have difficulty learning in skewed target distribution.

Synthetic Minority Over-sampling Technique (SMOTE) is a common technique to oversample a dataset used in a typical classification problem. We applied this method on minority class after splitting test and train data. However, the AUC didn't change much. One

thing we noticed is that some variables' feature importances changed dramatically after adopting SMOTE (see below chart).

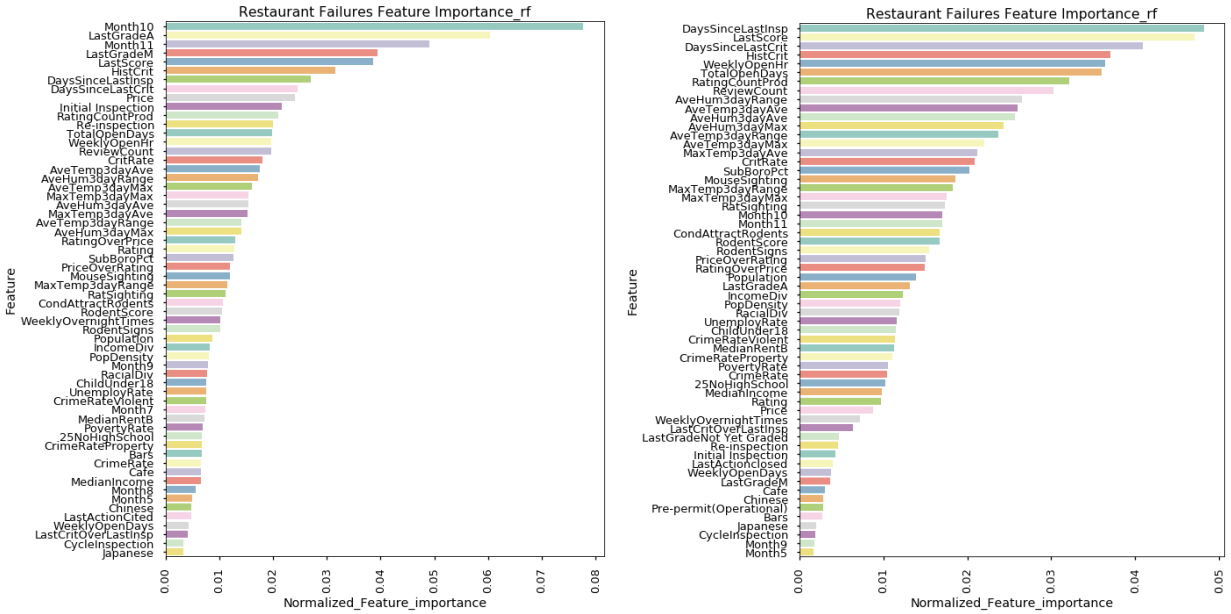


Figure 11: Feature Importance of RF Before(Right) and After(Left) SMOTE

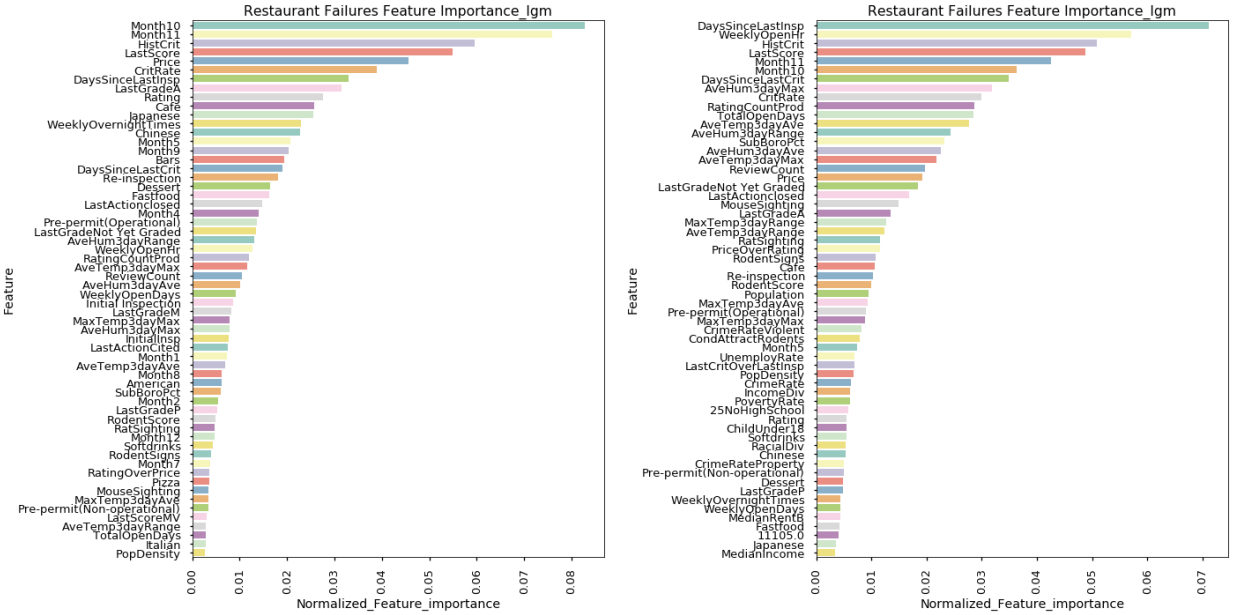


Figure 12: Feature Importance of LightGBM Before(Right) and After(Left) SMOTE

Our guess is that the variations within the minority class is relatively high while the difference between the two classes is relatively low. Since SMOTE uses KNN to create new samples, if the variations within the minority class is high, then SMOTE may end up using samples that are not real neighbors.

5.4 More about Dealing with Imbalanced Data

In the later stage, when considering new attempts to improve our model, we came up with a different approach to deal with imbalanced data. Rather than aggregating all inspection information to build one single instance for each restaurant, we decided to keep all inspections records for each restaurant and update each inspection to include only one latest information before its date. This way, our dataset got 10 times more data instances. A larger dataset should lower the risk of overfitting. We choose to manually separate test set by randomly selecting 1000 unique records from their latest inspection records with 13.6% of the instance with 'Critical Flag' equals 0 and the rest greater than 0. Then we select equal sizes of data instance with different target variables from the rest records to build a balanced training set. For the rest features, we only have their latest static values. So we dropped most of them, such as majority of neighborhood profile, yelp data such as reviewcount and rating, and weather details. Although the sample size was increased and test set was balanced, due to the dramatic decrease of available features, the model perform on this new dataset is not good, failing to result in even average performance.

6. Deployment

6.1 Business Deployment

We could deploy our code onto AWS service and build pipeline to make forecasting for each restaurant at runtime. We could also apply the same pipeline to datas in different Cities to generalize our model to the world.

6.2 Deployment Issue

We need to periodically update our model since the yelp/google API related data including reviews, would change over time.

One constraint of our model is that our dataset combines information from yelp API, which may not be 100% accurate. For instance, during the data cleaning process, we found some records with highly suspicious opening time, such as only operating half an hour in a week. Although we tried to clean the data as best we can, we cannot guarantee there is no omission. The rating and review counts can also be biased or fake, which will lower the predictability of our model.

For future forecasting, all future instances should have almost the same features as engaged in our models and the methods of preprocessing data should be identical to those we applied to our training dataset.

6.3 Risk Identified

Since our models is based on features that depend on historical inspection database, it may incur some problems since those with better performance in the past are more likely to be predicted as having very low potential to commit multiple violations. With relatively low frequency to be inspected, those may in turn become risky; However, those having multiple critical violations in the history will have more motivation or under pressure to get completely rectified. Based on those conditions, our model need to be adjusted.

7. Reference

1. NYC Health: The Inspection Process
2. Imblearn
3. LightGBM
4. Datasource:

- NYC Open Data - DOHMH New York City Restaurant Inspection Results :
  This is our core data containing the target variable, which contains all violation citations from the Department of Health and Mental Hygiene (DOHMH) inspection conducted up to three years prior to the most recent inspection for restaurants in New York City.
- Yelp Fusion API:
  The Yelp Fusion API allows us to get the restaurant profile and user reviews from businesses of interest.
- Google Places API:
  This API allows us to request user reviews about the indicated restaurant.
- NOAA - Local Climatological Data (LCD):
  We get New York City hourly weather data from this online data tools. This contains temperature and humidity data that is relevant to our problem, since fresh food, thawed foods are sensitive to temperatures. Improper temperature control for food, are most susceptible to accommodating the start or spread of food borne illnesses.
- NYC Open Data - Rodent Complaints in NYC:
  It is fair to say that the worst possible scenario for any restaurant is a rodent problem. We use this dataset to identify which area's restaurants are more likely to suffer from rodent infestation, therefore scoring lower in inspection.
- NYU Furman Center - New York City Neighborhood Data Profiles :
  The William and Anita Newman Library - NYC Geographies:
  Neighborhood data is critical for understanding local demographic and identifying community needs. We are interested to know whether neighborhood profiles would have a influence on restaurants' performances in inspection.

8. Contribution

- Jialing Xu
    - Data Cleaning
    - Feature Engineering And Feature Selection
    - Try Different Encoding Method
    - Build Weighted Average Ensemble Result
    - Grid Search Cross Validation for Important Parameters
    - Try SVM and Simple Neural Network on Keras
    - Try Text Mining Yelp & Google Map Reviews
    - Writeup: Modeling And Evaluation
- Lin Jiang
    - Data Cleaning
    - Find Sub-datasets (Neighborhood Profile, Weather Data, Rodent Complaint)
    - Feature Engineering
    - Multithreading Yelp Query Restaurant Data
    - Build and Tune Random Forest Model
    - Tune LightGBM Model
    - Tableau Visualization
    - Try SMOTE, Non-linear SVM
    - Writeup: Data Preparation And Analysis, Other Attempts, Reference
- Rui Jiang
    - Data Cleaning
    - Background Research
    - Performs Descriptive Analysis
    - Build LightGBM Model
    - Visualization
    - Writeup: Business Understanding, Data Analysis, Deployment
- Yuxuan He
    - Data Cleaning
    - Multithreading Call To Google Map Api And Yelp Fusion API To Get Review Data
    - Perform Embedding Method On Balanced Dataset
    - Try Build New Dataset (One Instance Per Inspection) To Deal With Imbalanced Data
    - Writeup: Business Understanding And Data Understanding, Deployment, Overall Polishing The Paper