Are Chicago and NYC Facilities Compliant with Public Health Regulations?

Omkar R Sunkersett

University of Michigan

December 18, 2017

Abstract

Public Health is a fundamental aspect of a big city. People travel to big cities to avail the opportunities they offer. One important part of the city life is its food. It is crucial that big cities offer a variety of food items since they bring people from different walks of life together. Food items are typically made, stored and sold at eateries in such cities. People love to visit a number of places to eat within a big city. Such places appear to be clean while purchasing and consuming food items, but their kitchens have a number of irregularities that are never reported. My report uses data mining tools and techniques to unravel the mysteries that stay hidden from us. It reveals regulatory non-compliances that are not readily available to the public. Such non-compliances pose as a hazard to customers who purchase and consume food items at such facilities. This report helps Public Health officials keep a track of non-compliant facilities from Chicago & New York City and then decide about the course of action in addition to making this information available to the people of these cities.

1 Introduction

People in big cities often have a busy lifestyle. They work for a large number of hours. Some work at startups whilst others work at either medium or large sized companies. Some of them run their own businesses. Many of them are even parents of a number of children. The children live in their own world. They go to school to study, do their homework, play outdoor sports and participate in other activities that let them enjoy their childhood to the maximum extent. Parents have a number of responsibilities. They have to ensure that their children go to school daily, do their homework, stay happy, and return home safely amongst many other responsibilities. Therefore, parents lead a very busy lifestyle. Further, they have many household duties. They clean the house or apartment and do the laundry amongst other chores. They have to also cook food for themselves and their children. This leaves them tired and exhausted by the weekends. They prefer to eat outside with their children in order to avoid the burden of making meals on weekends.

This has also been a trend amongst the youth in their 20s and 30s. Many young professionals work for 50+ hours a week for a livelihood. They have a number of duties similar to most parents and prefer to eat outside during weekdays and weekends in order to reduce their overall burden. They visit a number of great eateries that offer a variety of food items. Such eateries have a number of staff who help maintain them. Staff members generally clean the place in addition to packing food items. Chefs work in the kitchen of such eateries and prepare food items for customers. Such food items are made in bulk and kept on the shelf or prepared spontaneously upon an order of receipt. Though such facilities are kept clean by the staff members, their kitchens are often not taken care of properly. Whilst some of them have minor problems related to non-food contact surfaces, the other ones have major problems regarding restaurant safety and food sanitation.

My paper focuses on addressing these challenges that lie outside the purview of the customer. It presents both quantitative and qualitative results to the people and public health department officials. It is a beginning step to make information about facilities in Chicago and New York City accessible to the people living in their neighborhoods. It will help the personnel from the Department of Public Health to keep a track on the regulatory violations concerning eatery safety, especially maintaining the safety and sanitation of food bought, sold, or prepared at such facilities. It can aid in predicting restaurant closings, violations more prominent by certain neighborhoods or cuisines and seasonal or time-based anomalies.

2 Related Work

Similar ideas have been pursued in the past by researchers using different techniques. For instance, the paper entitled "nEmesis: Which Restaurants Should You Avoid Today?" [1] addressed the problem of restaurant hygiene and safety by analyzing the tweets of Twitter users suffering from foodborne illnesses and collecting their GPS data to identify the restaurants they visited. It then used statistical modeling to assign "health scores" to various restaurants based on the proportion of users who fell ill after visiting those restaurants. Similarly, the paper entitled "Where Not to Eat? Improving Public Policy by Predicting Hygiene Inspections Using Online Reviews" [2] used statistical modeling techniques on online reviews and public health records to differentiate between violating and non-violating restaurants across the United States. Further, the paper titled "Sanitary Conditions of Food Vending Sites and Food Handling Practices of Street Food Vendors in Benin City, Nigeria: Implication for Food Hygiene and Safety" [3] used surveying and interviewing techniques to ascertain the sanitary conditions of food distribution sites and their food handling practices in order to identify statistically significant associations between the educational and hygiene statuses of the food premises in Benin City, Nigeria. On similar grounds, the paper entitled "Current knowledge, attitude and behavior of hand and food hygiene in a developed residential community of Singapore: a cross-sectional survey" [4] determined the knowledge, attitude and behavior of hand and food hygiene and the potential risk factors of diseases such as diarrhea using descriptive, univariate and multivariate statistical analyses. It should be noted that all these researchers employed techniques that were based either partially or completely on the subjectivity of the users, whereas my project uses techniques that eliminate the subjectivity of the user with the goal of performing an exclusively objective analysis using official records from the public health departments of two major US cities: Chicago and New York City, for a totally unbiased outcome. This can help in accurately predicting the ground truth.

3 Problem Definition and Data

This was a data mining problem based on frequent item set mining [5]. In this problem, I identified frequent patterns of items within a list of transactions in a database based upon a predefined condition called the minimum support threshold [5]. I had two datasets available from Kaggle for this purpose [6, 7]. Each dataset represented a database consisting of a list of transactions. Each row of the dataset represented a transaction in the database. I used comma-separated value (CSV) files, so each item in a transaction was separated by a comma and each transaction was separated by the new-line character "\n". The total number of times a given item appeared in the database defined the absolute support of that item [8]. The number of transactions in which a given item appeared in the database defined the relative support (s) of that item [8]. Further, the transactions had a number of items in them, so multiple items could appear together in the database as a pattern of items called an item set [5]. The order of these items within the item set did not matter. Instead, what was more important was the frequency of the item sets in the list of transactions of the database. An item set was said to be frequent if its support (s) was greater than or equal to the minimum support threshold [8].

This gave rise to another question: How were item sets dependent on each other? It introduced the concept of association rules, which defined the conditional probability between two item sets in the database [9, 10]. If I had two item sets, X and Y, and I wanted to determine the likelihood that a transaction containing X also contained Y, then I could calculate the conditional probability of Y given X, $P(Y \mid X)$. This conditional probability was called the confidence (c) of the association rule for that list of transactions in the database [11, 12]. The higher the confidence of the association rule, the higher was the likelihood of the presence of Y given that X was present in the database. For the definition of my problem, I considered the minimum support threshold as 1% (0.01) to account for maximum number of frequent item sets.

Each city had its own dataset of inspection data. For my project, I considered two major cities of the USA: Chicago and New York City. The original Chicago dataset [6] (number of rows: 153,810) had the following variables: Inspection ID, Facility Name, License #, Facility Type, Risk, Address, City, State, Zip, Inspection Date, Inspection Type, Results, Violations, Latitude, Longitude and Location. "Violations" was not an atomic attribute because it contained more than one violation encountered during an inspection. It was therefore essential to convert the "Violations" attribute into an atomic attribute for the purpose of this project. There was a maximum of 23 violations possible for a given inspection from the dataset. I also calculated the number of violations during each inspection and stored them in a new attribute called "Number of Violations". Further, for the scope of the project, I required only the following attributes: Facility Name, Facility Type, Risk, Address, Zip, Inspection Date, Inspection Type, Results, Number of Violations and Violation Codes. The input dataset was parsed using Python [13] to produce a final dataset that contained all these attributes, each separated by a comma and ending with the new-line character. Similarly, the New York dataset [7] (number of rows: 399,918) had the following variables: Inspection ID, Facility Name, Borough Name, Building #, Street, Zip, Phone, Cuisine, Inspection Date, Violation Code, Violation Description, Criticality, Score, Grade, Grade Date, Record Date and Inspection Type. I combined Building # and Street into a single attribute called "Address". Violation Code was an atomic attribute; hence, I did not have to process it any further for the New York City dataset. For the scope of the project, I required only the following attributes: Facility Name, Borough Name, Address, Zip, Cuisine, Inspection Date, Violation Code, Criticality and Inspection Type.

Moreover, it was crucial to be able to classify [14] the facilities as either "Recommended" or "Not Recommended" based on a set of features. This set of features was obtained from the output of the frequent patterns in the list of transactions of the database. Each pattern was basically an item set in this regard. Classification [14] was important because it could help the public determine whether it was safe to purchase and consume food items from a given facility of the respective city depending upon whether the facility had been assigned a "Recommended" or "Not Recommended" status.

4 Methodology

To solve the first part of the problem, I used the frequent pattern growth mining algorithm, which utilized an item set mining technique to calculate the relative support for each frequent pattern in the list of transactions of the database [5]. For the purpose of this project, I set the minimum support threshold to 1% (0.01) in order to account for as many frequent patterns as possible. However, before executing the algorithm using the code, it was important to preprocess the data.

The Chicago dataset [6] had the following variables: Inspection ID, Facility Name, License #, Facility Type, Risk, Address, City, State, Zip, Inspection Date, Inspection Type, Results, Violations, Latitude, Longitude and Location. In this regard, the "Violations" attribute was not atomic because it contained more than one violation encountered during an inspection. It was therefore necessary to convert the "Violations" attribute into an atomic attribute for the purpose of this project. I also calculated the number of violations during each inspection and stored them in a new attribute called "Number of Violations". Further, for the scope of this project, I required only the following attributes: Facility Name, Facility Type, Risk, Address, Zip, Inspection Date, Inspection Type, Results, Number of Violations and Violation Codes.

By implementing the frequent pattern growth mining algorithm, I found the patterns that had high relative support and were most relevant for my analysis [5, 8]. For instance, it was important for me to ascertain the relative support for facilities that had been assigned a "High" risk. It was essential to determine what type of facilities were those that had the "High" risk and whether those facilities had passed their respective inspections. There were multiple types of inspections in the list of transactions of the database such as various complaints, license checks, task forces and canvasses. I focused on those types of inspections that had the highest relative support (typically the top) only. It was also necessary to find the frequent patterns of violation codes that had the highest relative support from the list of transactions and determine their occurrences for those facilities that had a medium to high risk. It was useful to consider the frequent types of inspections that were associated with these violations and their results for the purpose of analysis along with information such as addresses and zip codes.

Similarly, the New York dataset [7] had the following variables: Inspection ID, Facility Name, Borough Name, Building #, Street, Zip, Phone, Cuisine, Inspection Date, Violation Code, Violation Description, Criticality, Score, Grade, Grade Date, Record Date and Inspection Type. I combined Building # and Street into a single attribute called "Address". "Violation Code" was an atomic attribute; hence, I did not have to process it any further for New York City. For the scope of the project, I required only the following attributes: Facility Name, Borough Name, Address, Zip, Cuisine, Inspection Date, Violation Code, Criticality and Inspection Type.

Further, I implemented the frequent pattern growth mining algorithm and found the patterns that had high relative support and were most relevant for my analysis [5, 8]. For example, I ascertained the relative support for facilities that had been marked as "Critical". I determined what type of inspections were performed for these facilities and the boroughs to which they belonged. I found the frequent patterns of violation codes that had the highest relative support from the list of transactions in the database and determined their occurrences for these facilities. I considered the most frequent types of cuisines that were associated with the "Critical" flag and their violations and the different types of inspections along with information such as addresses and zip codes. This was performed because there was a likelihood of an association amongst such attributes of data for the given list of transactions [9, 10].

Moreover, by extracting some features from the analysis, I built a set of features matrix for each dataset and split the dataset into two parts: training and testing (in the ratio 30:70) [15]. I trained and tested a variety of text classifiers (Naïve Bayes, Random Forest, Logistic Regression, Support Vector Machines, Stochastic Gradient Descent, and Perceptron) with their default configuration using scikit-learn's TF-IDF vectorizer on the set of features matrix for the given set of labels [16, 17]. The set of features for the Chicago dataset were {Facility Type, Risk Level, Type of Inspection, Violation Code} and set of labels were {0: Fail, 1: Pass} [6]. Similarly, the set of features for the New York City dataset were {Cuisine Type, Borough, Type of Inspection, Violation Code} and set of labels were {0: Not Critical, 1: Critical} [7]. They were chosen from the outputs of the association rules, which helped me to estimate how well these sets of features were correlated in the given datasets [9, 10]. Finally, I selected the classification technique that produced the highest F1 score, which became the evaluation metric for my classifiers [14, 18]. This was performed to ensure that I had chosen the correct machine learning model for my work.

Moreover, the inspections were performed on each facility from Chicago and NYC for a number of times over many years (4 and 6 years, respectively) as evident from the given datasets. I calculated the number of times each facility from both these cities passed and failed the inspections, respectively. Based on that information, I assigned a tag to each facility. A facility was tagged as "Recommended" if the classifier's result was "Pass" (for Chicago) or "Not Critical" (for NYC) and the facility's number of inspections passed were greater than or equal to its number of inspections failed. Similarly, a facility was tagged as "Not Recommended" if the classifier's result was "Fail" (for Chicago) or "Critical" (for NYC) and the facility's number of inspections passed were lesser than its number of inspections failed for the same dataset.

5 Evaluation and Results

I used an implementation of the frequent pattern growth mining algorithm called the "fp-growth-0.1.3" package for finding the frequent patterns in the list of transactions of the given database [5, 19]. I defined three baselines for comparing my results of frequent patterns. The first baseline was a relative support threshold of 50%. The second baseline was a relative support threshold of 10%. The third baseline was a relative support threshold of 1%.

For classification, I used the worst performing classification technique as the baseline for each city [14]. The worst performing one was that which produced the lowest F1 score [18]. For Chicago, the Multinomial Naïve Bayes Classifier was the baseline, producing a F1 score of 0.879296472481. For New York City, the Radial Basis Support Vector Classifier was the baseline, producing a F1 score of 0.99985088271.

Below were my results for each baseline of each city during evaluation –

5.1 Observations for Chicago

For the first baseline of 50% minimum relative support threshold (76,905), the major observations were the following –

The package [19] took 1.9546 seconds to find the frequent patterns from the list of transactions in the database. The number of frequent patterns were 5. I removed those item sets that contained a "Pass" item and was left with only 4 item sets:

```
['Risk 1 (High)'] | Support: 107351
['Restaurant'] | Support: 101230
 ['Canvass'] | Support: 81712
['Risk 1 (High)', 'Restaurant'] | Support:
80196
```

I discovered only one association rule [9, 10, 11, 12]: 'Risk 1 (High)' –> 'Restaurant', having a support and confidence of 0.5214 and 0.7470, respectively.

For the second baseline of 10% minimum relative support threshold (15,381), the major observations were as follows –

The package [19] took 5.7637 seconds to find the frequent patterns from the list of transactions in the database. The total number of frequent patterns were 265. I removed those item sets that contained a "Pass" item and was left with only 185 item sets from which the top 50 were as follows –

```
['34'] | Support: 74716
['35'] | Support: 66320
['33'] | Support: 65995
['Risk 1 (High)', 'Canvass'] | Support:
59895
['38'] | Support: 56621
['32'] | Support: 55909
['Risk 1 (High)', '34'] | Support: 54695
['Restaurant', 'Canvass'] | Support: 54245
['Restaurant', '34'] | Support: 50701
['Risk 1 (High)', '33'] | Support: 50122
['Risk 1 (High)', '35'] | Support: 49810
['34', '33'] | Support: 49517
['Restaurant', '33'] | Support: 48173
['34', '35'] | Support: 47895
['Risk 1 (High)', 'Restaurant', 'Canvass']
| Support: 44511
['Restaurant', '35'] | Support: 43849
['Risk 1 (High)', '32'] | Support: 43411
['35', '33'] | Support: 42816
['Risk 1 (High)', '38'] | Support: 42330
['Risk 1 (High)', 'Restaurant', '34'] |
Support: 41119
['Restaurant', '32'] | Support: 40446
['Risk 1 (High)', 'Restaurant', '33'] |
Support: 39828
['34', '32'] | Support: 37942
['Canvass', '34'] | Support: 37764
['Restaurant', '38'] | Support: 37496
['Risk 1 (High)', '34', '33'] | Support:
37055
['34', '38'] | Support: 36814
['Risk 1 (High)', 'Restaurant', '35'] |
Support: 36296
['34', '35', '33'] | Support: 35902
['41'] | Support: 35851
['Risk 1 (High)', '34', '35'] | Support:
35560
['33', '32'] | Support: 35493
['Restaurant', '34', '33'] | Support:
35466
['35', '32'] | Support: 35450
['Canvass', '33'] | Support: 34996
['35', '38'] | Support: 34496
['Risk 1 (High)', 'Restaurant', '32'] |
Support: 34345
['Risk 1 (High)', '35', '33'] | Support:
32679
['33', '38'] | Support: 32502
['Canvass', '35'] | Support: 32259
['Risk 2 (Medium)'] | Support: 31845
['Restaurant', '34', '35'] | Support:
31684
```

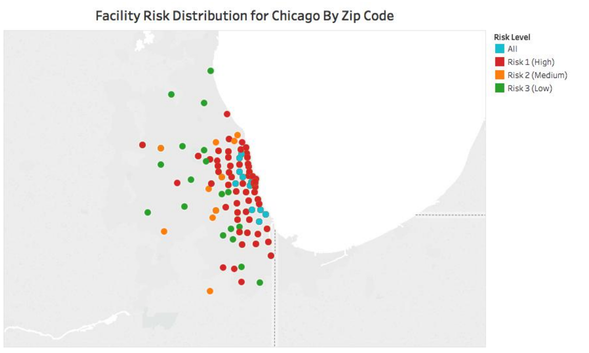
```
['Risk 1 (High)', 'Restaurant', '38'] |
Support: 30766
['Restaurant', '35', '33'] | Support:
30273
['Fail'] | Support: 29845
['Risk 1 (High)', 'Canvass', '34'] | Sup-
port: 29557
['Risk 1 (High)', 'Restaurant', '34',
'33'] | Support: 29225
['Risk 1 (High)', '34', '32'] | Support:
29163
['38', '32'] | Support: 28998
['34', '33', '32'] | Support: 28620
```

Some of the association rules [9, 10, 11, 12] that were discovered were as follows –

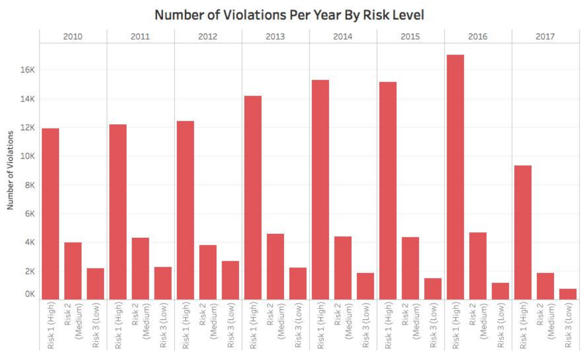
A. 'Risk 1 (High)' -> 'Canvass', having a support and confidence of 0.3894 and 0.5579, respectively.
B. 'Risk 1 (High)' -> '34', having a support and confidence of 0.3491 and 0.5002, respectively.
C. 'Restaurant' -> 'Canvass', having a support and confidence of 0.3527 and 0.5359, respectively.
D. 'Restaurant' -> '34', having a support and confidence of 0.3236 and 0.4916, respectively.

For the baseline of 1% minimum relative support threshold (1,539), the package [19] took 55.9738 seconds to find the frequent patterns from the list of transactions in the database. The number of frequent patterns were 10,672. The results were not so significant for evaluation.

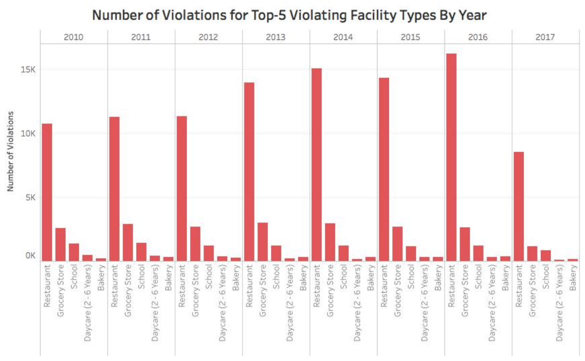
From the Tableau plots [20], most of the high-risk facilities (depicted in red) were located close to the coastline. Some of these facilities belonged to all the three risk levels (depicted in light blue) over the years, indicating that they had shown some improvement. The facilities belonging to the medium and low risk levels were depicted in orange and green, respectively. They were located in the distant suburbs of Chicago. As their distance from the city increased, their risk eventually decreased.



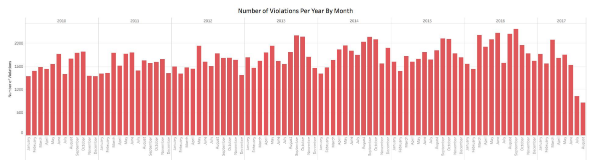
The number of violations for Risk Level 1 (High) increased gradually, whereas those for Risk Levels 2 (Medium) and 3 (Low) were almost steady for over the years.



The number of violations by restaurants increased gradually, whereas those by grocery stores, schools, daycares and bakeries remained almost steady for over the years.



The number of violations increased at the beginning of a new season (especially for spring and fall). They were minimum during the winter season and maximum during the fall season.

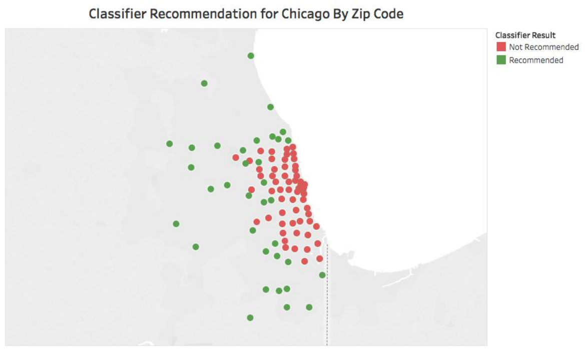


The following were the F1 scores of my classifiers (ranked from highest to lowest):

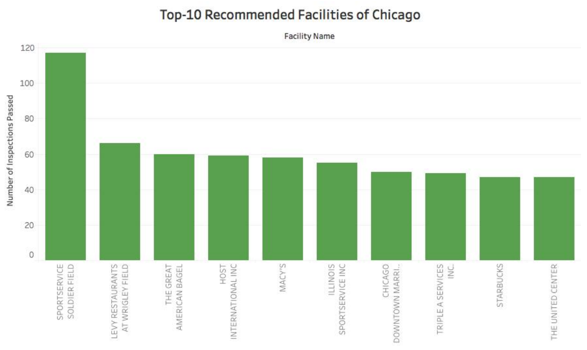
Classifier [14]	**F1 Score** [18]
Stochastic Gradient Descent	0.961991430571
Random Forest	0.960704837637
Linear SVC	0.959807806694
Logistic Regression	0.959769539464
Perceptron	0.946108704335
Radial Basis SVC	0.928968884009
Bernoulli Naïve Bayes	0.912075797753

Out of the 23,964 facilities of Chicago, the stochastic gradient descent classifier tagged 22,752 and 1,212 facilities as "Recommended" and "Not Recommended", respectively. The top-10 recommended and top-10 not recommended facilities were selected based on the number of inspections passed and failed, respectively.

Most of the not recommended facilities (depicted in red) were located along the coastline or close suburbs. The recommended facilities were located mostly in the distant suburbs with the exception of a few facilities along the northern and southern coasts.

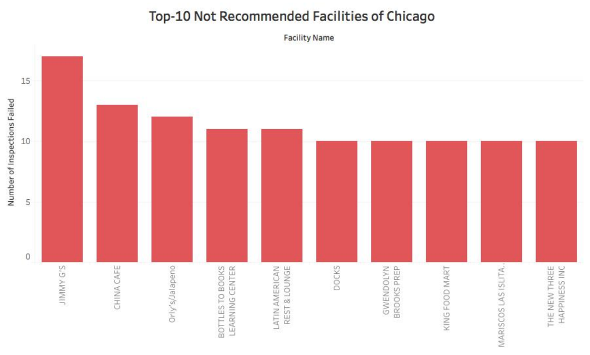


The top-10 recommended facilities with their number of inspections passed (in parentheses) were Sportservice Soldier Field (117), Levy Restaurants at Wrigley Field (66), The Great American Bagel (60), Host International Inc. (59), Macy's (58), Illinois Sportservice Inc. (55), Chicago Downtown Marriott (50), Triple A Services Inc. (49), Starbucks (47) and The United Center (47), respectively.



Chicago's Top-10 Recommended Facilities			
Facility Name	Facility Type	Street Address	Zip Code
SPORTSERVICE SOLDIER FIELD	Restaurant	1410 S MUSEUM CAMPUS DR	60605
LEVY RESTAURANTS AT WRIGLEY FIELD	Restaurant	1060 W ADDISON ST	60613
THE GREAT AMERICAN BAGEL	Restaurant	11601 W TOUHY AVE	60666
HOST INTERNATIONAL INC	Restaurant	11601 W TOUHY AVE	60666
MACY'S	Restaurant	111 N STATE ST	60602
ILLINOIS SPORTSERVICE INC	KIOSK	333 W 35TH ST	60616
CHICAGO DOWNTOWN MARRIOTT	Restaurant	540 N MICHIGAN AVE	60611
TRIPLE A SERVICES INC.	Mobile Food Dispenser	2637 S THROOP ST	60608
THE UNITED CENTER	Restaurant	1901 W MADISON ST	60612
STARBUCKS	Restaurant	11601 W TOUHY AVE	60666

Similarly, the top-10 not recommended facilities with their number of inspections failed were Jimmy G's (17), China Café (13), Orly's (12), Bottles to Books Learning Center (11), Latin American Rest & Lounge (11), Docks (10), Gwendolyn Brooks Prep (10), King Food Mart (10), Mariscos Las Islitas (10) and The New Three Happiness Inc. (10), respectively.



5.2 Observations for New York City

For the first baseline of 50% minimum relative support threshold (199,959), the major observations were as follows –

The package [19] took 4.1726 seconds to find the frequent patterns from the list of transactions in the database. The number of frequent patterns were 2. They were as follows –

```
['Cycle Inspection / Initial Inspection']
| Support: 230431
['Critical'] | Support: 220082
```

There were no association rules present for this baseline [9, 10, 11, 12].

For the second baseline of 10% minimum relative support threshold (39,992), the major observations were as follows –

The package [19] took 5.0771 seconds to find the frequent patterns from the list of transactions in the database. The total number of frequent patterns were 30. I removed those item sets that contained a "Not Critical" item and was left with only 22 item sets from which the top 20 were as follows –

```
['MANHATTAN'] | Support: 159574
['Cycle Inspection / Initial Inspection',
'Critical'] | Support: 136761
['BROOKLYN'] | Support: 99598
['Cycle Inspection / Re-inspection'] |
Support: 99436
['Cycle Inspection / Initial Inspection',
'MANHATTAN'] | Support: 94138
['QUEENS'] | Support: 92415
['American'] | Support: 90968
['Critical', 'MANHATTAN'] | Support: 88496
['Critical', 'Cycle Inspection / Re-
inspection'] | Support: 57242
['Cycle Inspection / Initial Inspection',
'BROOKLYN'] | Support: 56205
['Cycle Inspection / Initial Inspection',
'Critical', 'MANHATTAN'] | Support: 56065
['10F'] | Support: 55785
['Cycle Inspection / Initial Inspection',
'American'] | Support: 55317
['Critical', 'BROOKLYN'] | Support: 54030
['Cycle Inspection / Initial Inspection',
'QUEENS'] | Support: 52179
['Critical', 'QUEENS'] | Support: 50945
['Critical', 'American'] | Support: 49026
['MANHATTAN', 'American'] | Support: 47170
['Chinese'] | Support: 42321
['08A'] | Support: 39992
```

Some of the association rules [9, 10, 11, 12] that were significant for this baseline were as follows –

A. 'Cycle Inspection / Initial Inspection' -> 'MANHATTAN', having a support and confidence of 0.6120 and 0.4085, respectively.
B. 'Cycle Inspection / Initial Inspection' -> 'American', having a support and confidence of 0.3596 and 0.2401, respectively.

For the baseline of 1% minimum relative support threshold (4,000), the major observations were as follows –

The package [19] took 9.0152 seconds to find the frequent patterns from the list of transactions in the database. The number of frequent patterns were 509. I removed those item sets that contained a "Not Critical" item and was left with the following item sets:

```
['MANHATTAN', 'Cycle Inspection / Re-
inspection'] | Support: 39530
['BRONX'] | Support: 34895
['Cycle Inspection / Initial Inspection',
'Critical', 'BROOKLYN'] | Support: 33242
['Cycle Inspection / Initial Inspection',
'10F'] | Support: 33221
['Cycle Inspection / Initial Inspection',
'Critical', 'American'] | Support: 32267
```

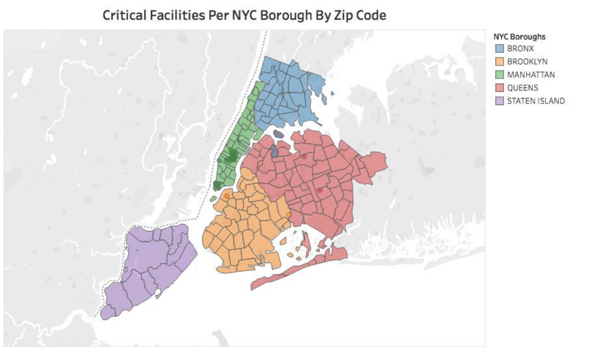
```
['Cycle Inspection / Initial Inspection',
'Critical', 'QUEENS'] | Support: 31098
['Cycle Inspection / Initial Inspection',
'MANHATTAN', 'American'] | Support: 29021
['02G'] | Support: 27852
['Critical', '02G'] | Support: 27852
['04L'] | Support: 27777
['Critical', '04L'] | Support: 27777
['06D'] | Support: 26115
['Critical', '06D'] | Support: 26115
['06C'] | Support: 25955
['Critical', '06C'] | Support: 25955
['Critical', 'MANHATTAN', 'American'] |
Support: 25872
['Critical', 'Chinese'] | Support: 24867
['Pre-permit (Operational) / Initial In-
spection'] | Support: 24702
['Cycle Inspection / Initial Inspection',
'08A'] | Support: 24693
['BROOKLYN', 'Cycle Inspection / Re-
inspection'] | Support: 24388
['Cycle Inspection / Re-inspection',
'QUEENS'] | Support: 23597
['Critical', 'MANHATTAN', 'Cycle Inspec-
tion / Re-inspection'] | Support: 23087
['Cycle Inspection / Initial Inspection',
'Chinese'] | Support: 22843
['10B'] | Support: 22141
['MANHATTAN', '10F'] | Support: 22053
['Cycle Inspection / Re-inspection',
'American'] | Support: 21945
['Cycle Inspection / Initial Inspection',
'BRONX'] | Support: 19988
['BROOKLYN', 'American'] | Support: 19675
['02B'] | Support: 19650
['Critical', '02B'] | Support: 19650
['04N'] | Support: 19482
['Critical', '04N'] | Support: 19482
['Latin (Cuban Dominican Puerto Rican
South & Central American)'] | Support:
19135
['Critical', 'BRONX'] | Support: 19027
['Pizza'] | Support: 18754
['Cycle Inspection / Initial Inspection',
'Critical', '02G'] | Support: 17834
['Cycle Inspection / Initial Inspection',
'02G'] | Support: 17834
['Italian'] | Support: 17732
['Cycle Inspection / Initial Inspection',
'Critical', '06D'] | Support: 17607
['Cycle Inspection / Initial Inspection',
'06D'] | Support: 17607
['Cycle Inspection / Initial Inspection',
'Critical', 'MANHATTAN', 'American'] |
Support: 17126
['Cycle Inspection / Initial Inspection',
'Critical', '04L'] | Support: 16910
['Cycle Inspection / Initial Inspection',
'04L'] | Support: 16910
['Caf\xc3\x83\xc2\xa9/Coffee/Tea'] | Sup-
port: 15568
['Cycle Inspection / Re-inspection',
'10F'] | Support: 15410
['Mexican'] | Support: 15217
['MANHATTAN', '08A'] | Support: 15145
['QUEENS', 'American'] | Support: 15033
['Critical', 'Pre-permit (Operational) /
Initial Inspection'] | Support: 14962
['Cycle Inspection / Initial Inspection',
'Critical', '06C'] | Support: 14898
['Cycle Inspection / Initial Inspection',
'06C'] | Support: 14898
['Japanese'] | Support: 14491
['Cycle Inspection / Initial Inspection',
'Critical', 'Chinese'] | Support: 14098
['BROOKLYN', '10F'] | Support: 13946
['Critical', 'BROOKLYN', 'Cycle Inspection
/ Re-inspection'] | Support: 13763
['Cycle Inspection / Initial Inspection',
'MANHATTAN', '10F'] | Support: 13447
['STATEN ISLAND'] | Support: 13427
['Critical', 'Cycle Inspection / Re-
inspection', 'QUEENS'] | Support: 13324
['Cycle Inspection / Initial Inspection',
'10B'] | Support: 13200
['American', '10F'] | Support: 13181
['QUEENS', '10F'] | Support: 13088
['BROOKLYN', 'Chinese'] | Support: 13069
['Caribbean'] | Support: 13021
['QUEENS', 'Chinese'] | Support: 12791
['Cycle Inspection / Initial Inspection',
'Critical', '02B'] | Support: 12555
['Cycle Inspection / Initial Inspection',
'02B'] | Support: 12555
['Critical', 'Cycle Inspection / Re-
inspection', 'American'] | Support: 12343
['Cycle Inspection / Initial Inspection',
'Critical', '04N'] | Support: 12334
['Cycle Inspection / Initial Inspection',
'04N'] | Support: 12334
['MANHATTAN', '02G'] | Support: 12160
['Critical', 'MANHATTAN', '02G'] | Sup-
port: 12160
['Bakery'] | Support: 11959
['Spanish'] | Support: 11695
['Cycle Inspection / Initial Inspection',
'BROOKLYN', 'American'] | Support: 11606
['Cycle Inspection / Initial Inspection',
'Critical', 'BRONX'] | Support: 11523
['MANHATTAN', 'Cycle Inspection / Re-
inspection', 'American'] | Support: 11516
['MANHATTAN', '06D'] | Support: 11491
['Critical', 'MANHATTAN', '06D'] | Sup-
port: 11491
['Cycle Inspection / Re-inspection',
'08A'] | Support: 11188
['10003'] | Support: 11105
['MANHATTAN', '10003'] | Support: 11105
['Cycle Inspection / Re-inspection', 'Chi-
nese'] | Support: 11085
['MANHATTAN', 'Chinese'] | Support: 11007
['Critical', 'Latin (Cuban Dominican Puer-
to Rican South & Central American)'] |
Support: 10866
['Cycle Inspection / Initial Inspection',
'Italian'] | Support: 10866
['Cycle Inspection / Initial Inspection',
'Pizza'] | Support: 10860
['MANHATTAN', '06C'] | Support: 10646
['Critical', 'MANHATTAN', '06C'] | Sup-
port: 10646
['Cycle Inspection / Initial Inspection',
'Latin (Cuban Dominican Puerto Rican South
& Central American)'] | Support: 10611
['BROOKLYN', '08A'] | Support: 10570
['Pre-permit (Operational) / Re-
inspection'] | Support: 10568
['MANHATTAN', 'Italian'] | Support: 10393
['Critical', 'BROOKLYN', 'American'] |
Support: 10302
```

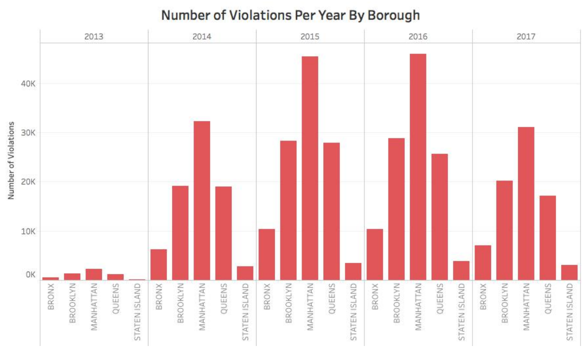
```
['Critical', 'Pizza'] | Support: 10276
['MANHATTAN', '04L'] | Support: 10127
['Critical', 'MANHATTAN', '04L'] | Sup-
port: 10127
```

Some of the association rules [9, 10, 11, 12] that were significant for this baseline were as follows –

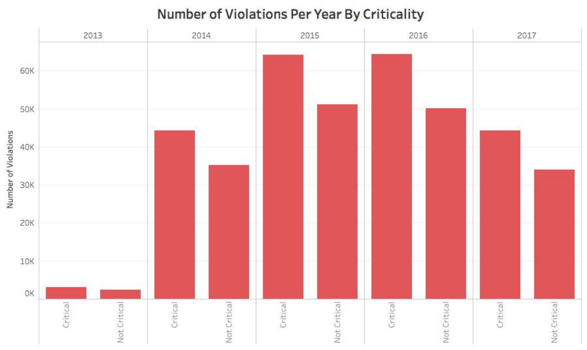
A. 'Cycle Inspection / Initial Inspection' -> '10F', having a support and confidence of 0.2160 and 0.1442, respectively.
B. 'MANHATTAN' -> 'American', having a support and confidence of 0.3067 and 0.2956, respectively.
C. 'MANHATTAN' -> '10F', having a support and confidence of 0.1434 and 0.1382, respectively.

From the Tableau plots [20], Manhattan (depicted in green) had the highest number of critical facilities followed by Brooklyn (depicted in orange) and Queens (depicted in red). They increased gradually each year for these boroughs. Bronx (depicted in blue) and Staten Island (depicted in lavender) had no critical facilities at all for the same duration.

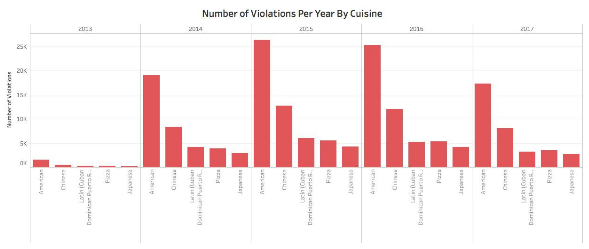




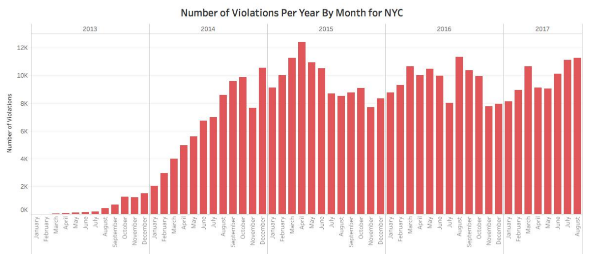
The number of violations were higher for the critical facilities than the non-critical ones. They increased gradually each year for both the critical and non-critical facilities.



The number of violations were highest for the American cuisines, followed by Chinese, Latin, Pizza, and Japanese. They increased gradually each year for every cuisine.



The number of violations increased at the beginning of a new season (especially for spring and fall). They were minimum during the winter season and maximum during either the spring or the fall season.



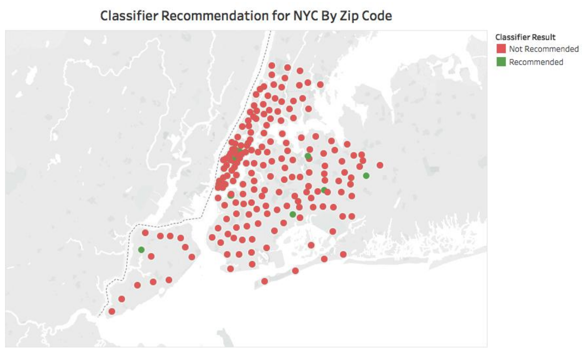
The following were the F1 scores of my classifiers (ranked from highest to lowest):

Classifier [14]	**F1 Score** [18]
Stochastic Gradient Descent	0.999999999586
Multinomial Naïve Bayes	0.999999998451
Linear SVC	0.999999993127
Logistic Regression	0.999999970394

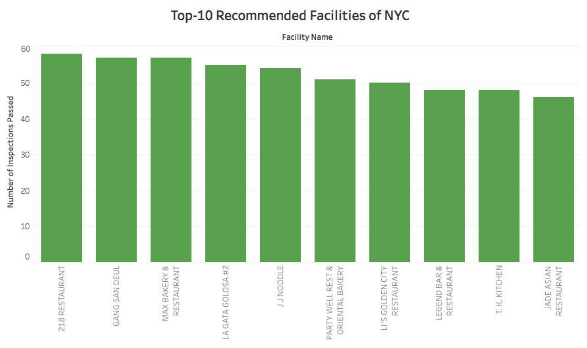
Bernoulli Naïve Bayes	0.999993504342
Perceptron	0.999987016107
Random Forest	0.999974003185

Out of the 25,219 facilities of New York City, the stochastic gradient descent classifier tagged 15,689 and 9,530 facilities as "Recommended" and "Not Recommended", respectively. The top-10 recommended and top-10 not recommended facilities were selected based on the number of inspections passed and failed, respectively.

Many of the New York City restaurants were not recommended (depicted in red) by the classifier and were distributed equally amongst all the boroughs. A few of them were recommended (depicted in green) but were superimposed by the strength of the not recommended ones (depicted in red).

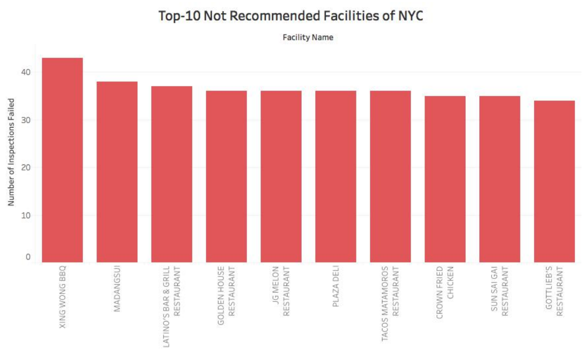


The top-10 recommended facilities with their number of inspections passed (in parentheses) were 218 Restaurant (58), Gang San Deul (57), Max Bakery & Restaurant (57), La Gata Golosa #2 (55), JJ Noodle (54), Party Well Rest. & Oriental Bakery (51), Li's Golden City Restaurant (50), Legend Bar & Restaurant (48), T.K. Kitchen (48) and Jade Asian Restaurant (46), respectively.



NYC's Top-10 Recommended Facilities			
Facility Name	Cuisine	Street Address	Zip Code
218 RESTAURANT	Chinese	218220 GRAND STREET	10013
GANG SAN DEUL	Korean	248-25 NORTHERN BLVD	11362
MAX BAKERY & RESTAURANT	Chinese	3763 90 STREET	11372
LA GATA GOLOSA #2	Bakery	8901 37 AVENUE	11372
J J NOODLE	Chinese	19 HENRY STREET	10002
PARTY WELL REST. & ORIENTAL BAKERY	Korean	14932 41ST AVE	11355
LI'S GOLDEN CITY RESTAURANT	Chinese	2324 ARTHUR AVENUE	10458
LEGEND BAR & RESTAURANT	Chinese	88 7 AVENUE	10011
T. K. KITCHEN	Asian	26 SAINT MARKS PLACE	10003
JADE ASIAN RESTAURANT	Chinese	13628 39 AVENUE	11354

Similarly, the top-10 not recommended facilities with their number of inspections failed were Xing Wong BBQ (43), Madangsui (38), Latino's Bar & Grill Restaurant (37), JG Melon Restaurant (36), Plaza Deli (36), Tacos Matamoros Restaurant (36), Golden House Restaurant (36), Crown Fried Chicken (35), Sun Sai Gai Restaurant (35) and Gottlieb's Restaurant (34), respectively.



6 Discussion

From my overall analysis, I was able to obtain a number of qualitative and quantitative findings that helped me answer my research questions: Were the facilities compliant with the city health regulations? Could public health officials take regulatory actions against them? Could those facilities be recommended to the public?

In the case of Chicago, I found that a majority of the facilities having "High" risks were located within the city, especially next to the coast. This meant that those facilities were not taking appropriate actions to ensure that they were striving to minimize their regulatory non-compliances. The public health officials should have focused on this part of the city than on the distant suburbs, which had mostly "Low" and "Medium" risk facilities. Some of the "High" risk facilities near the city were improving over time since they were belonging to a "Low" or "Medium" risk category at some point of time. These facilities should have been rewarded by the public health officials because they served as motivators of improvement for the facilities that belonged to the "High" risk category. The frequent item sets [5] revealed that the topmost type of inspection was "Canvass" (count: 81,712), which was an official examination performed in detail to ascertain the authenticity of the regulatory compliances of facilities. Though this didn't have any particular implication, it revealed the fact that the public health officials were highly interested in keeping a check on the facilities of the city and its neighborhoods. Further, the top-10 violations revealed the following information [6] –

1. Code 34: The floors were either not constructed as per the protocol or not cleaned or repaired properly using the correct methods, or they did not have a coving (i.e. molded uniform profile) installed on them.
2. Code 35: The walls, ceiling and attached equipment were not constructed as per the protocol, or they were not repaired or cleaned properly with the correct methods.
3. Code 33: The food utensils and non-food contact equipment were not cleaned properly and free of abrasive detergents.
4. Code 38: The kitchen ventilation and vented equipment were not as per the protocol and the plumbing was not installed properly and well-maintained.
5. Code 32: The food and non-food contact surfaces were not properly designed, constructed and well-maintained.
6. Code 41: The premises were not maintained free of litter and unnecessary articles, or the cleaning equipment was not properly stored.
7. Code 40: The refrigeration and metal stem thermometers were not installed or were conspicuous.
8. Code 21: There were no certified food managers on site when potentially hazardous foods were prepared and served.
9. Code 31: Single-serve articles were either reused or not stored properly and multi-use utensils were not well-cleaned.
10. Code 29: Minor violations from a previous inspection were not corrected.

These violation codes were mostly applicable to restaurants whose number of violations increased over time, especially at the beginning of a new season (specifically, for spring and fall). This indicated that the changes in climatic conditions may have affected the number of violations, which were minimum during the winter season and maximum during the fall season. There could have also been other accountable internal and external factors responsible for these seasonal changes. These factors are not known unfortunately.

Moreover, the association rules [9, 10, 11, 12] revealed that the Facility Type, Risk Level, Type of Inspection, and Violation Code attributes had a strong correlation amongst themselves (i.e. their confidence was between 50-55%), making them good candidates for selecting the features of the classifiers. These features scored the highest F1 score with the Stochastic Gradient Descent classification technique, which beat the Multinomial Naïve Bayes baseline by nearly 9% [14, 18]. The high F1 score helped me rely on the accuracy of the classifier and tag the facilities as either "Recommended" or "Not Recommended" based on the output of the classifier (i.e. Pass, Fail) and the absolute support of the number of inspections passed and failed, respectively. The percentage of the recommended facilities (94.94%) was significantly higher than the percentage of the not recommended ones (5.06%). This is a good indication for the people of Chicago because more than 90% of the facilities were considered to be "safe". The top-10 recommended facilities and top-10 not recommended facilities (mentioned in my results) were the best and worst places to purchase & consume food items, respectively. The people of Chicago should refrain from visiting the top-10 not recommended facilities because they pose as a serious hazard to public health.

In the case of New York City, I found that all of the facilities belonging to the "Critical" category were located in either Manhattan, Brooklyn or Queens, indicating that these boroughs were likely to have a higher rate of non-compliance. The Bronx and Staten Island boroughs had no "Critical" facilities at all. This is good news for the public because they can purchase and consume food items from these boroughs without much concern. The public health officials should have rewarded the facilities in these boroughs for their consistent regulatory compliances. They should have been more concerned about the Manhattan, Brooklyn and Queens boroughs. Further, the frequent item sets [5] revealed that most of the violations occurred during either the initial inspection or the cycle inspections (count: 230,431). This high count indicates that the public health officials were keen at keeping a track on the facilities in all of the boroughs throughout the entire lifecycle of inspections. The top-10 violations revealed the following information [7] –

1. Code 10F: The non-food contact surfaces and equipment were not properly constructed or unacceptable material was used to construct them. They might have not been properly maintained or sealed, raised, spaced or movable to allow accessibility for cleaning on all sides, above and underneath the unit.
2. Code 08A: The facility was not vermin proof, or it provided harborage or conditions conducive to attracting vermin to the premises or allowing vermin to exist.
3. Code 02G: The cold food items were held above 41ºF (with smoked fish and reduced oxygen packaged foods above 38ºF) except during necessary preparation.
4. Code 04L: There was evidence of mice or live mice present in the facility's food or non-food areas.
5. Code 06D: The food contact surfaces were not properly washed, rinsed and sanitized after each use and following any activity when the contamination occurred.
6. Code 06C: The food was not protected from potential source of contamination during storage, preparation, transportation, display or service.
7. Code 10B: The plumbing was not properly installed or maintained, or the anti-siphonage or backflow prevention device was not provided where required, or the equipment or floor was not properly drained, or the sewage disposal system was in disrepair or not functioning properly.
8. Code 02B: The hot food items were not held at or above 140ºF.
9. Code 04N: The filth flies or food/refuse/sewage-associated (FRSA) flies were present in the facility's food or non-food areas. The filth flies included house flies, little house flies, blow flies, bottle flies and flesh flies. The food/refuse/sewage-associated flies included fruit flies, drain flies and Phorid flies.

10. Code 04M: There were live roaches present in the facility's food or non-food areas.

These violations were higher for the critical facilities than the non-critical ones. They increased gradually each year for both the critical and non-critical facilities. The American, Chinese, Latin, Pizza and Japanese cuisines were more affected than the other cuisines. The violations increased gradually each year for every cuisine. They increased at the beginning of a new season (especially for spring and fall). They were minimum during the winter season and maximum during either the spring or the fall season. This indicates that the changes in climatic conditions might have affected the number of violations similar to the phenomena observed in Chicago. There could have been other factors also responsible for these seasonal changes. The exact causes are not known.

Moreover, the association rules [9, 10, 11, 12] revealed that the Cuisine Type, Borough, Type of Inspection and Violation Code attributes had a moderate correlation amongst themselves (i.e. their confidence was between 13-41%), making them good candidates for selecting the features of my classifiers. These features scored the highest F1 score with the Stochastic Gradient Descent classification technique, which beat the Radial Basis Support Vector Classifier baseline by less than 1% [14, 18]. This high F1 score helped me rely on the accuracy of my classifier and tag the facilities as either "Recommended" or "Not Recommended" based on the output of the classifier (Critical, Not Critical) and the absolute support of the number of inspections passed and failed, respectively. The percentage of the recommended facilities (62.21%) was higher than the percentage of the not recommended facilities (37.78%). This is not a bad indication for the public of New York City because more than 60% of its facilities are considered to be "safe". The top-10 recommended facilities and top-10 not recommended facilities (mentioned in my results) were the best and worst places to purchase and consume food items, respectively. The people of New York City should refrain from visiting the top-10 not recommended facilities because they pose as a serious hazard to their health. Additionally, the high percentage (35-40%) of not recommended facilities should be a concern for the city's Department of Public Health. Strict actions should be taken against these facilities because they have had some serious violations over the years and can potentially affect public safety and sanitation within the city.

7 Conclusion

From the results and discussion, it is evident that the facilities from New York City are not only less compliant than those from Chicago but are also having more serious problems such as improper sewage disposal, vermin, mice, roaches, filth flies and FRSA flies. New York City has a significant 37% non-compliance as compared to the 5% non-compliance of Chicago. Therefore, it is important for the New York City public health officials to take stringent actions against these dangerous regulatory non-compliances concerning public safety, especially for maintaining the safety and sanitation of food items bought, sold, or prepared at those facilities. It will also be important for people to be more careful whilst purchasing or consuming food items at these facilities of New York City. They might be at a risk of falling ill from multiple health problems caused by lack of hygiene or food poisoning after consumption. As a further step, public health officials might even want to publish the details (i.e. name and location) of such facilities (to blacklist them) or revoke their licenses.

8 Acknowledgements

I'd like to sincerely thank Professor David Jurgens for all his support and guidance with which I obtained the right direction to conceptualize and pursue my idea. It was a pleasure attending his course on Data Mining: Methods and Applications at the University of Michigan School of Information. The weekly readings of this course helped me understand the core concepts of the data mining techniques. The course assignments were fantastic and helped me to understand and use different data mining packages, which were finally selected for this project. I'd like to also thank all of my classmates for their constructive feedback about my project work during the class discussions. It helped me to refine the scope of my project over the course of the semester and take the right decisions at the right time.

9 References

[1] Sadilek, A., Brennan, S., Kautz, H., & Silenzio, V. (2013). NEmesis: Which Restaurants Should You Avoid Today? Proceedings of the First AAAI Conference on Human Computation and Crowdsourcing, 139-146.

[2] Kang, J., Kuznetsova, P., Luca, M., & Yejin, C. (2013). Where Not to Eat? Improving Public Policy by Predicting Hygiene Inspections Using Online Reviews. EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 6 pages.

[3] Okojie P. W., & Isah E. C. (2014). Sanitary Conditions of Food Vending Sites and Food Handling Practices of Street Food Vendors in Benin City, Nigeria: Implication for Food Hygiene and Safety. Journal of Environmental and Public Health, vol. 2014, Article ID 701316, 6 pages.

[4] Pang, J., Chua, S. W. J. L., & Hsu, L. (2015). Current knowledge, attitude and behaviour of hand and food hygiene in a developed residential community of Singapore: a cross-sectional survey. BMC Public Health, 15(1), 12 pages.

[5] Han (2000). Mining Frequent Patterns Without Candidate Generation. Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. SIGMOD '00: 1–12.

[6] Chicago, C. O. (2017, August 30). Chicago Restaurant Inspections. Retrieved November 20, 2017, from https://www.kaggle.com/chicago/chi-restaurant-inspections

[7] York, C. O. (2017, August 29). New York City Inspections. Retrieved November 20, 2017, from https://www.kaggle.com/new-york-city/nyc-inspections

[8] Hahsler, M. (2015). A Probabilistic Comparison of Commonly Used Interest Measures for Association Rules. Retrieved November 20, 2017, from http://michael.hahsler.net/research/association_rules/measures.html

[9] Piatetsky-Shapiro, G., & Frawley W. J. (1991). Discovery, analysis, and presentation of strong rules. Knowledge Discovery in Databases, AAAI/MIT Press, Cambridge, MA, ISBN: 9780262660709.

[10] Agrawal, R., Imieliński, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. Proceedings of the 1993 ACM SIGMOD international conference on Management of data - SIGMOD '93. p. 207.

[11] Hahsler, M. (2005). Introduction to association rules — A computational environment for mining association rules and frequent item sets. Journal of Statistical Software.

[12] Hipp, J., Güntzer, U., & Nakhaeizadeh, G. (2000). Algorithms for association rule mining — A general survey and comparison. ACM SIGKDD Explorations Newsletter.

[13] Python (programming language). (n.d.). Retrieved November 20, 2017, from https://en.wikipedia.org/wiki/Python_(programming_language)

[14] Mohri M., Rostamizadeh A., & Talwalkar A. (2012). Foundations of Machine Learning, MIT Press, Cambridge, MA, ISBN: 9780262018258.

[15] Sklearn.model_selection.train_test_split. (n.d.). Retrieved November 20, 2017, from http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[16] Working With Text Data. (n.d.). Retrieved November 20, 2017, from http://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html

[17] Sklearn.feature_extraction.text.TfidfVectorizer. (n.d.). Retrieved November 20, 2017, from

http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[18] Van Rijsbergen, C. J. (1979). Information Retrieval (2nd ed.). Butterworth.

[19] Fp-growth 0.1.3 : Python Package Index. (n.d.). Retrieved November 20, 2017, from https://pypi.python.org/pypi/fp-growth/0.1.3

[20] Tableau Software. (n.d.). Retrieved December 17, 2017, from https://en.wikipedia.org/wiki/Tableau_Software

A Supplemental Material

Python code for preprocessing the data –

```
import csv

fread =
open("/Users/omkarsunkersett/Downloads/inspections/chicago.csv", 'r')
rows = csv.reader(fread)
fwrite =
open("/Users/omkarsunkersett/Downloads/inspections/chicago-final.csv", 'w')
fwrite.write('Facility Name,Facility Type,Risk Level,Address,Zip Code,Inspection Date,Type of Inspection,Result,Number of Violations,Violation Code 1,Violation Code 2,Violation Code 3,Violation Code 4,Violation Code 5,Violation Code 6,Violation Code 7,Violation Code 8,Violation Code 9,Violation Code 10,Violation Code 11,Violation Code 12,Violation Code 13,Violation Code 14,Violation Code 15,Violation Code 16,Violation Code 17,Violation Code 18,Violation Code 19,Violation Code 20,Violation Code 21,Violation Code 22,Violation Code 23\n')
rows.next()

for row in rows:
	row[13] = row[13].strip()
	if row[13] != '':
		if ' | ' not in row[13]:
			vcodes = row[13].split(' - Comments: ')[0].split('.')[0]
		else:
			vcodes = ','.join([s.split()[0].rstrip('.') for s in row[13].split(' | ')])
	else:
		vcodes = ''
	fwrite.write(row[1].replace(',','')+','+row[4].replace(',','')+','+row[5].replace(',','')+','+row[6].replace(',','')+','+row[9].replace(',','')+','+row[10].replace(',','')+','+row[11].replace(',','')+','+row[12].replace(',','')+','+vcodes+'\n')
fwrite.close()
fread.close()

fread =
open("/Users/omkarsunkersett/Downloads/inspections/nyc.csv", 'r')
rows = csv.reader(fread)
fwrite =
open("/Users/omkarsunkersett/Downloads/inspections/nyc-final.csv", 'w')
fwrite.write('Facility Name,Borough,Street Address,Zip Code,Cuisine,Inspection Date,Violation Code,Criticality,Type of Inspection\n')
rows.next()

for row in rows:
	fwrite.write(row[1].replace(',','')+','+row[2].replace(',','')+','+row[3].replace(',','')+" "+row[4].replace(',','')+','+row[5].replace(',','')+','+row[7].replace(',','')+','+row[8].replace(',','')+','+row[10].replace(',','')+','+row[12].replace(',','')+','+row[17].replace(',','')+'\n')
fwrite.close()
fread.close()
```

Python code for the Chicago classifier –

```
import csv, time
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import BernoulliNB, GaussianNB, MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression, SGDClassifier, Perceptron
from sklearn import svm
from sklearn.metrics import f1_score

start_time = time.time()
full_df =
pd.read_csv('/Users/omkarsunkersett/Downloads
/inspections/chicago-final.csv',
keep_default_na=False)
full_df = full_df.query("Result == 'Pass' | Result
== 'Fail'")
train_df, test_df = train_test_split(full_df,
test_size = 0.7)
del full_df

facilities = dict()
train_data = []
train_labels = []
for index, row in train_df.iterrows():
        i = 1
        violations = []
        while i < 24:
                if row['Violation Code '+str(i)]
!= '':
                        viola-
tions.append(row['Violation Code '+str(i)])
                i += 1
        violations = ' '.join(violations)
        if '1' in row['Risk Level']:
                risk_level = 'High'
        elif '2' in row['Risk Level']:
                risk_level = 'Medium'
        elif '3' in row['Risk Level']:
                risk_level = 'Low'
        else:
                risk_level = 'All'
        if row['Result'] == 'Fail':
                result = 0
        elif row['Result'] == 'Pass':
                result = 1
        if (row['Facility Name'], row['Facility
Type'], row['Address'], row['Zip Code']) not in
facilities.keys():
                facilities[(row['Facility Name'],
row['Facility Type'], row['Address'], row['Zip
Code'])] = {'n_pass': 0, 'n_fail': 0}
        if row['Result'] == 'Pass':
                facilities[(row['Facility Name'],
row['Facility Type'], row['Address'], row['Zip
Code'])]['n_pass'] += 1
        elif row['Result'] == 'Fail':
                facilities[(row['Facility Name'],
row['Facility Type'], row['Address'], row['Zip
Code'])]['n_fail'] += 1
        train_data.append(row['Facility Type']+'
'+risk_level+' '+row['Type of Inspection']+'
'+violations)
        train_labels.append(result) # 0: Fail, 1:
Pass; Ground Truth

test_data = []
test_labels = []
for index, row in test_df.iterrows():
        i = 1
        violations = []
        while i < 24:
                if row['Violation Code '+str(i)]
!= '':
                        viola-
tions.append(row['Violation Code '+str(i)])
                i += 1
        violations = ' '.join(violations)
        if '1' in row['Risk Level']:
                risk_level = 'High'
        elif '2' in row['Risk Level']:
                risk_level = 'Medium'
        elif '3' in row['Risk Level']:
                risk_level = 'Low'
        else:
                risk_level = 'All'
        if row['Result'] == 'Fail':
                result = 0
        elif row['Result'] == 'Pass':
                result = 1
        test_data.append(row['Facility Type']+'
'+risk_level+' '+row['Type of Inspection']+'
'+violations)
        test_labels.append(result) # 0: Fail, 1:
Pass; Groud Truth

vectorizer = TfidfVectorizer()
train_vectors = vectoriz-
er.fit_transform(train_data)
test_vectors = vectorizer.transform(test_data)

#classifier = BernoulliNB()
#classifier = MultinomialNB()
#classifier = RandomForestClassifier()
#classifier = LogisticRegression()
#classifier = svm.SVC(kernel='rbf', probabil-
ity=True)
#classifier = svm.LinearSVC()
classifier = SGDClassifier()
#classifier = Perceptron()

classifier.fit(train_vectors, train_labels)
prediction = classifier.predict(test_vectors)

i = 0
for index, row in test_df.iterrows():
        if (row['Facility Name'], row['Facility
Type'], row['Address'], row['Zip Code']) not in
facilities.keys():
                facilities[(row['Facility Name'],
row['Facility Type'], row['Address'], row['Zip
Code'])] = {'n_pass': 0, 'n_fail': 0}
        if prediction[i] == 1:
```

```
			facilities[(row['Facility Name'], row['Facility Type'], row['Address'], row['Zip Code'])]['n_pass'] += 1
		elif prediction[i] == 0:
			facilities[(row['Facility Name'], row['Facility Type'], row['Address'], row['Zip Code'])]['n_fail'] += 1
		i += 1

fwrite = open('/Users/omkarsunkersett/Downloads/inspections/chicago-results.csv', 'w')
fwrite.write('Facility Name,Facility Type,Address,Zip Code,Number of Inspections Passed, Number of Inspections Failed,Classifier Result\n')
for k,v in facilities.items():
	if v['n_pass'] >= v['n_fail']:

	fwrite.write(k[0]+','+k[1]+','+k[2]+','+k[3]+','+str(v['n_pass'])+','+str(v['n_fail'])+',Recommended\n')
	else:

	fwrite.write(k[0]+','+k[1]+','+k[2]+','+k[3]+','+str(v['n_pass'])+','+str(v['n_fail'])+',Not Recommended\n')
fwrite.close()

elapsed_time = time.time() - start_time

print "F1 Score: ",f1_score(test_labels, prediction, average='binary')
print "Elapsed Time: ",str(elapsed_time),"seconds"
```

Python code for the New York City classifier –

```
import csv, time
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import BernoulliNB, GaussianNB, MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression, SGDClassifier, Perceptron
from sklearn import svm
from sklearn.metrics import f1_score

start_time = time.time()

full_df = pd.read_csv('/Users/omkarsunkersett/Downloads/inspections/nyc-final.csv', keep_default_na=False, encoding='latin-1', low_memory=False)
full_df = full_df.query("Criticality == 'Critical' | Criticality == 'Not Critical'")
train_df, test_df = train_test_split(full_df, test_size = 0.7)
del full_df

facilities = dict()
train_data = []
train_labels = []
for index, row in train_df.iterrows():
	if row['Criticality'] == 'Not Critical':
		result = 0
	elif row['Criticality'] == 'Critical':
		result = 1
	if (row['Facility Name'], row['Cuisine'], row['Street Address'], row['Zip Code']) not in facilities.keys():
		facilities[(row['Facility Name'], row['Cuisine'], row['Street Address'], row['Zip Code'])] = {'n_pass': 0, 'n_fail': 0}
	if row['Criticality'] == 'Not Critical':
		facilities[(row['Facility Name'], row['Cuisine'], row['Street Address'], row['Zip Code'])]['n_pass'] += 1
	elif row['Criticality'] == 'Critical':
		facilities[(row['Facility Name'], row['Cuisine'], row['Street Address'], row['Zip Code'])]['n_fail'] += 1
	train_data.append(row['Borough'].replace(' ','-')+' '+row['Cuisine'].replace(' ','-')+' '+row['Violation Code']+' '+row['Type of Inspection'].replace(' ','-'))
	train_labels.append(result) # 0: Not Critical, 1: Critical; Ground Truth

test_data = []
test_labels = []
for index, row in test_df.iterrows():
	if row['Criticality'] == 'Not Critical':
		result = 0
	elif row['Criticality'] == 'Critical':
		result = 1
	test_data.append(row['Borough'].replace(' ','-')+' '+row['Cuisine'].replace(' ','-')+' '+row['Violation Code']+' '+row['Type of Inspection'].replace(' ','-'))
	test_labels.append(result) # 0: Not Critical, 1: Critical; Ground Truth

vectorizer = TfidfVectorizer()
```

```
train_vectors = vectoriz-
er.fit_transform(train_data)
test_vectors = vectorizer.transform(test_data)

#classifier = BernoulliNB()
#classifier = MultinomialNB()
#classifier = RandomForestClassifier()
#classifier = LogisticRegression()
#classifier = svm.SVC(kernel='rbf', probabil-
ity=True)
#classifier = svm.LinearSVC()
classifier = SGDClassifier()
#classifier = Perceptron()

classifier.fit(train_vectors, train_labels)
prediction = classifier.predict(test_vectors)

i = 0
for index, row in test_df.iterrows():
        if (row['Facility Name'], row['Cuisine'],
row['Street Address'], row['Zip Code']) not in fa-
cilities.keys():
                facilities[(row['Facility Name'],
row['Cuisine'], row['Street Address'], row['Zip
Code'])] = {'n_pass': 0, 'n_fail': 0}
        if prediction[i] == 1:
                facilities[(row['Facility Name'],
row['Cuisine'], row['Street Address'], row['Zip
Code'])]['n_pass'] += 1
        elif prediction[i] == 0:
                facilities[(row['Facility Name'],
row['Cuisine'], row['Street Address'], row['Zip
Code'])]['n_fail'] += 1
        i += 1

results_df = []
for k,v in facilities.items():
        if v['n_pass'] >= v['n_fail']:
                results_df.append([k[0], k[1],
k[2], k[3], str(v['n_pass']), str(v['n_fail']), 'Rec-
ommended'])
        else:
                results_df.append([k[0], k[1],
k[2], k[3], str(v['n_pass']), str(v['n_fail']), 'Not
Recommended'])

results_df = pd.DataFrame(results_df, col-
umns=['Facility Name', 'Cuisine', 'Address', 'Zip
Code', 'Number of Inspections Passed', 'Number
of Inspections Failed', 'Classifier Result'])
re-
sults_df.to_csv('/Users/omkarsunkersett/Downlo
ads/inspections/nyc-results.csv', encoding='latin-
1', index=False)

elapsed_time = time.time() - start_time

print "F1 Score: ",f1_score(test_labels, predic-
tion, average='binary')
print "Elapsed Time:
",str(elapsed_time),"seconds"
```