Predicting Risk Category From LIVE Restaurant Inspection Data

Ashish Verma
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Canada
averm019@uottawa.ca

***Abstract*—This paper studies data shared by Government of San Francisco to predict the risk category of restaurants. The dataset used is LIVE and is updated daily by the government. This paper explores steps involved in pre-processing of data like null values, noise and outlier detection, to improve training and test set. Post cleaning of data, important features are selected and data is then divided into training and testing split of 80% and 20%. Using training data, machine learning algorithm from various family are considered and trained. The performance of the algorithms is measured using accuracy, recall, precision and F1 score. Most of the models performed averagely estimated using k-fold cross validation on the test set. Some of the models didn't perform well and were evaluated to be overfit. At last comparison of algorithms is done using performance metrics and box and whisker plot to select the best model within family and overall for the dataset.**

Keywords—Machine Learning, Overfitting, Accuracy, True positive, True negative, Precision, Recall, F1-score.

I. INTRODUCTION

Any technology that any user uses today is benefitted from machine leaning. Machine learning is a subfield of artificial intelligence. The ultimate milestone when using machine learning is to understand the structure of data and fit that data into the model, which can then be used to predict values based on same type of the data. Ultimately, machine learning analyses the data for relationship and patterns. Unlike, traditional computer programs where instruction are explicitly programmed to do certain job, machine learning can by itself understand the relationship between various features of the data. This experiment explores different algorithms used by different models in machine learning, to predict the risk category for restaurants.

Restaurant inspections are very important for the prevention of food borne diseases. To predict if any restaurant is continually behaving in common pattern or is a periodic defaulter, machine learning can be used. This will in turn help in predicting future risk category for the restaurants and those with high risk category can be inspected on regular basis.

In this paper we'll understand the data recorded for each restaurant in San Francisco, California, recorded during inspection. This data will then be preprocessed and cleaned to remove any noise if present. Post cleaning the data it will be made to fit in different machine learning algorithms and models will be measured for their accuracy on predicting risk category. At the end various comparison between different models are done to select the best model for this dataset.

II. PROBLEM DOMAIN

More than 54 billion meals are served at 844,000 commercial food establishments in the United States each year [1]. Of which 46% of money, Americans spend on food goes for the restaurant meals. When it comes to food, restaurant makes up more than 40% of the market share. On any ordinary day 44% of adults in the United States eat at a restaurant. On an average 550 foodborne disease outbreaks reported to the Centers for Disease Control and Prevention each year. From 1993 through 1997, saw rise of outbreak, which is greater than 40%. This rise was attributed to commercial food establishments. Preventing restaurant-associated foodborne disease outbreaks is an important task of the safety of public and public health departments [2].

On regular basis restaurants in United states are inspected by local, county, or health department. The principle goal of inspection is to prevent foodborne disease by the means of inspection scores. Inspection scores are intended to enhance food safety. During inspection data is recorded y inspection officers in database. This data is publicly available and is also shared by local media on regular basis.

For this experiment we have limited our dataset to San Francisco, California location in United States. This data is easily available from the website of Office of the chief data officer, city and county of San Francisco [3]. This data is updated on regular basis and can be used to predict values like inspection score, inspection type, risk category etc.

This paper will try to explore different features in the data and would use them effectively to classify restaurants on the basis of risk category i.e. one of low, medium and high.

By classifying such values, the restaurants which are identified as high-risk category can be inspected on a regular basis and certain measures can be effectively put in place before any outbreak.

III. DATASET

Government of San Francisco administer and manages a web portal DataSF [3]. This portal can be accessed by anyone. The mission of DataSF is to empower use of data to continuously improve city services and operations. All the datasets available on this portal are regularly updated and is sometimes referred to as LIVE data. The available published data in non-fabricated and supports broad trust, transparency and accountability [4]. It will be beneficial to experiment on LIVE data and use obtained trained model for greater good.

A. Source and description

Dataset used in this paper is named as Restaurant Scores - LIVES Standard. The source of the dataset is DataSF[3]. When browsing web portal, different categories are available to select data from. This dataset falls under the category of Health and Social Services [5]. This data is updated daily and reflect real values. This dataset was created on October 28, 2015, and from then on, it is updated daily. San Francisco's LIVES restaurant inspection data leverages the LIVES Flattened Schema, which is based on LIVES version 2.0, cited on Yelp's website [5].

Dataset comprises of 17 columns and 53.7K rows. However, for this experiment data recorded from year 2016 – February 22, 2019 is used to train model and test various algorithms in machine learning.

Below is the summary of dataset, Table 1, with column name and their data type. Business in column name refers to restaurant.

TABLE I. COLUMNS IN DATASET WITH DATA TYPE

Column	Data Type
business_id	Integer/Number
business_name	String/Object
business_address	String/Object
business_city	String/Object
business_state	String/Object
business_postal_code	Integer/Number
business_latitude	Float/Decimal point
business_longitude	Float/Decimal point
business_location	String/Object
business_phone_number	String/Object
inspection_id	String/Object
inspection_date	Datetime
inspection_score	Float/Decimal point
inspection_type	String/Object
violation_id	String/Object
violation_description	String/Object
risk_category	String/Object

B. *Preprocessing*

To preprocess data, each and every column was carefully analyzed. The goal of preprocessing is to convert the data type of all columns in a particular symmetry. Our approach here, tries to convert all the columns to numerical data type. During the phase of preprocessing a lot of noisy data was removed. Also, during feature engineering a lot of redundant data from specific columns were removed and unique elements were kept, to help model train better. Below is detailed process of data preprocessing for each column with feature selection.

- business_name: This column lists restaurant name. However, restaurants can be uniquely identified by business_id, so it was dropped.
- business_address: This feature is not important because we have postal codes, latitude and longitude data to observe business location. Hence, this was dropped.
- business_city: It was observed that business city in whole dataset was San Francisco. As this column cannot contribute much towards training of data, it was dropped.
- business_state: It was observed that business state in whole dataset was California. As this column cannot contribute towards training of data, it was dropped.
- business_location: This column is combination of longitude and latitude. As it introduces redundancy in data, it was dropped.
- business_phone_number: This doesn't seem an important feature when predicting risk category for restaurant. This column was dropped.
- inspection_type: This data was converted into categorical data. But after removing null values it was realized that only one category was consistent in all the rows. So, this column was dropped.

Upon further analysis, it was observed that there were some mis-typed data. business_postal_code was having values like '64110' whereas all the postal codes were of the form 9XXXX. However, these types of data were corrected. Also, there were some string literals like 'CA', 'Ca', '941'. Rows containing these values were dropped as it was unlikely to estimate postal code from other values. Also, some of the postal codes were of the form 9XXXX-XXXX, in this case we trimmed the column and kept first five digits to maintain symmetry.

While checking null and unique values it was observed that postal codes and latitude were having zero values. These rows were dropped from the dataset, as it can impact training of the data.

Further, inspection_date was broken down into three columns namely year, month and day to help model train better.

C. *Feature selection*

Feature selection for training data is very important. If proper features are not selected than it can either underfit or overfit the model. Thus, making our model highly unreliable. It is very important to select features which can convey some story or pattern in the data. Our model will try to learn pattern within the data and will make predictions based on it. Some of the below features were re-engineered and some were kept as is.

- business_id: This column is very important as it uniquely identifies all the restaurants.
- business_postal_code: This column can uniquely identify region of the restaurant, which can be an important feature to consider.
- business_latitude: This column lists latitude of restaurant, which can be used on a geographical map and estimate popular location.
- business_longitude: Same as above, it can be used on a geographical map and estimate popular location.
- inspection_id: This column is a combination of business id and inspection_date. So it was dropped. E.g. 62010-20160415. Here 62010 is business id and 20160415 is date on which inspection was conducted.
- inspection_date: This column lists date on which inspection was conducted. It also has a time unit attached to it which was trimmed during pre-processing as it was same in all rows.

Identify applicable funding agency here. If none, delete this text box.

- inspection_score: This is very important feature and wasn't altered.
- violation_id: This column uniquely identifies violation description. So, it was part of the training data.
- violation_description: As this data had finite unique values, it was converted to categorical numerical data. i.e. *Hygiene:*1, *Legal:*2, *Noncompliance:*3 *and Lack of Infrastructure:*4
- risk_category: This column is very important part of the dataset. We will be predicting risk category from the dataset. So, categories were converted to numerical data. i.e. *Low Risk*:1, *Moderate Risk*:2, *High Risk*:3

Converting data to numerical categorical values also favored in class balancing, and most of classes which were having least count got eliminated.

Post feature engineering our dataset was having 22,273 rows and 8 columns in total. Dataset representation is shown in Table 2.

TABLE II. COLUMNS IN DATASET WITH DATA TYPE POST PROCESSING

Int64Index: 22273 entries, 0 to 53324			
Data columns (total 8 columns)			
business_id	22273	non-null	int64
business_latitude	22273	non-null	float64
business_longitude	22273	non-null	float64
inspection_score	22273	non-null	float64
violation_id	22273	non-null	float64
violation_description	22273	non-null	float64
risk_category	22273	non-null	int64
day	22273	non-null	int64
dtypes: float64(5), int64(3)			

Based on the above data, few assumption and analysis were done to see if certain features are important with respect to score of the restaurant.

Postal code vs. count plot, see Fig.1, describes that few of the postal codes have very less density and less relative data, when compared to other postal codes.

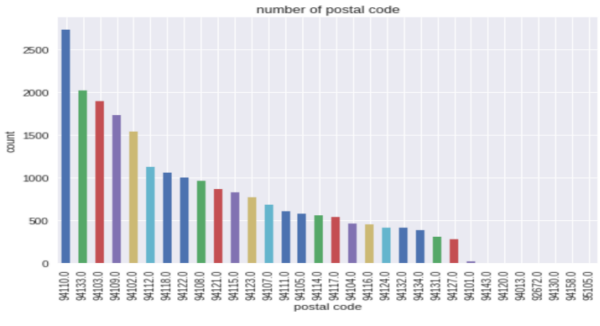

Fig.1. Plot showing postal code vs count relationship

Latitude vs Longitude plot see Fig. 2, describes that certain regions are hotspots for restaurants to run their business. It would be interesting to predict the risk category for restaurants clustered in specific regions.

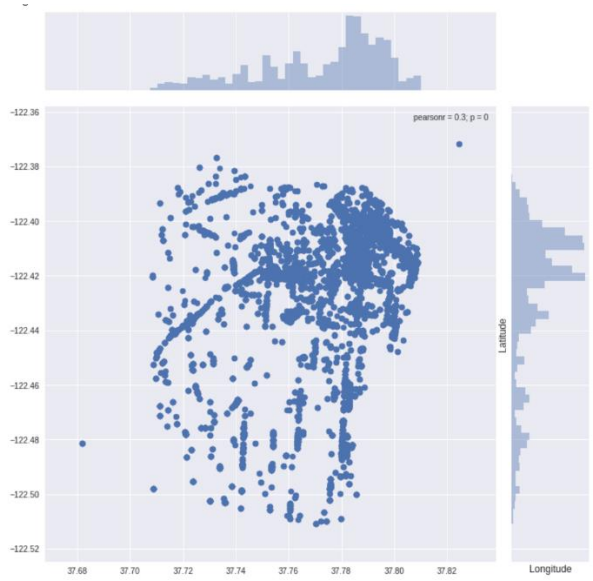

Fig.2. Plot showing latitude vs longitude relationship

Post analyzing the location of the restaurants, important features were selected using Extra Trees Classifier. This algorithm assigned score to each feature with high score as best and low score as not so important feature. Fig 3 illustrate the bar chart of the features with scores.

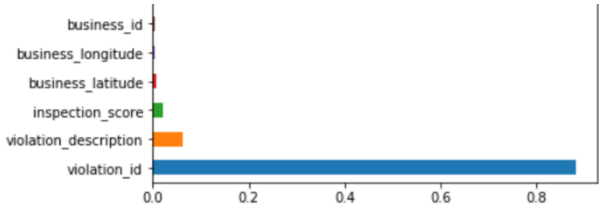

Fig.3. Plot showing feature scores

At last, it was decided to retain all the features except year, business_postal_code and month, post pre-processing, feature engineering and feature importance, as they will contribute towards training of the model. The day column wasn't dropped as it was having significantly higher value as compared to dropped ones.

IV. EXPERIMENTAL SETUP AND EVALUATION

To train model two separate datasets were used. One dataset was used for training and other for testing. After training the model, it was evaluated on few performance parameters like confusion matrix, precision, recall, accuracy and box plot. Below sections explains the setup and evaluation in detail.

A. Experimental setup

Our model will be trained to classify data into one of three risk categories. So, risk category will become predict label. After selecting the predictor, we will split our data into train and test datasets. Usually, training data is bigger than the test set. So, in our case training data will account for 80% of the total data and test data will account for 20%. This split of data is not done in sequential way. Dataset is first randomly shuffled and then it is divided into desired proportion. This is

done so that model gets exposed to data in a random manner. This also helps in training the model better.

TABLE III. TRAIN TEST SPLIT

Dataset	Percentage
Train	80%
Test	20%

Post-split, training data will be fit into multiple models. These models range from tree-based, rule-based, distance-based methods, as well as linear models and ensembles. After training, test data was used for determining prediction and score. Based on score multiple performance evaluation metrics were computed, which helped in determining the best model for the dataset. All models were executed on Google's Colaboratory environment, which requires no setup and runs entirely in the cloud [23]. The only exception was with rule-based model, they were executed on Weka [26].

B. Model evaluation

To select the best model, each model will be quantified to some metric value and graph plots. These metric values and graphs will help in the judgement of the best model for the dataset. These parameters are explained in detail below:

a) K-fold cross validation

Cross validation is procedure to evaluate machine learning model on a limited data sample using resampling [6]. It is often referred to as k-fold cross validation because k decides number of groups a given data sample is to be split into. Cross validation is used in machine learning model on unseen data to estimate the skill of the model which is less biased and less optimistic estimate, as compared to other methods.

b) Root mean squared error

Root mean squared error or RMSE is used to measure differences between values as predicted by model and actually observed. If RMSE is low it means performance is better [7]. It is usually used with linear models and rule-based models

c) R^2 or R-squared score

It determines how close the data is to fitted regression line. In general, higher the R-square value better the model fits data [8]. It is mostly used with linear models and rule-based models.

d) Accuracy

It is the ratio of number of correct predictions to total number of samples. Higher the accuracy better the model fits data. If accuracy is too high i.e. above 95% it means model is overfitting the data [11].

e) Precision score

It is the ratio of number of correctly predicted positives to total number of positives in the sample [10].

f) Recall

It is the ratio of number of correctly predicted positives to total number of all relevant samples. Which means all samples which should have been predicted as positives [10].

g) F1-score

F1 score is a harmonic mean between precision and recall. It signifies how robust and precise the model is [10].

h) Confusion matrix

It summarizes the performance of model by sampling true positive and negatives as well as false positives and negatives in a matrix [9] [10]. This further helps in deciding amongst which class model tends to get confused with.

i) Algorithm comparison using box and whisker plot

Box and whisker plot is used to evaluate performance of multiple models by plotting box and whisker plot of mean accuracy and mean variance, generated from k-fold cross validation [12]. It helps in evaluating performance of model visually.

V. MODEL AND ALGORITHMS

This section lists all the models and algorithms used in this experiment. Further in section 6, all the models are compared to choose the best one.

A. Linear models

Linear models in machine learning signify that model is described as a linear combination of feature. Linear models fit data using line equation, wherein a decision boundary is estimated to classify different classes lying on either side of boundary. The learning process calculates weight for each feature to build a model that can predict or classify target value. Some of the linear models used in the experiment are stated below with their performance indicators [14] [15].

a) Linear Regression

Linear regression fits the linear model with coefficients or features to minimize the residual sum of squares between observed responses in dataset and responses predicted by linear approximation. This model is not useful for classification task, but was used as a baseline model to evaluate against.

Below are the results from linear regression:

TABLE IV. LINEAR REGRESSION PERFORMANCE

Performance Indicator	Value
Root mean squared error	0.058
R^2 score	0.882

The performance of regression as computed by R^2 score is 88%, which at initial state is better.

b) Logistic Regression – newton cg

It is same as Linear regression. However, this regression task is used for classification. It uses logistic function at the core to train model. The key difference is that the output value is modelled as a binary value rather than numeric value. [21]. This model implements regularized logistic regression using newton-cg solver.

TABLE V. LOGISTIC CLASSIFIER-NEWTON CG ACCURACY

Performance Indicator	Value
Accuracy	0.645

The performance of regression as computed by accuracy is 64%, which is below our baseline model.

c) Logistic Regression – saga

This is similar to above logistic regression. Only key difference is that it implements regularized logistic regression using saga solver.

TABLE VI. LOGISTIC CLASSIFIER-SAGA CG ACCURACY

Performance Indicator	Value
Accuracy	0.509

The performance of regression as computed by accuracy is 50%, which means it underperformed as compared to linear regression.

d) Ridge Classifier

As it implements l2 regularization so it is called as ridge regression [14]. It adds squared magnitude of coefficient as a penalty to loss function.

TABLE VII. RIDGE CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.797

The performance of Ridge classifier as computed by accuracy is 79%, which is far better than Logistic regression.

e) SGD Classifier

Stochastic gradient descent is very efficient approach as it uses convex loss functions to train model with linear classifier [16].

TABLE VIII. SGD CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.509

The performance of SGD classifier as computed by accuracy is 50%, which is almost same as logistic regression and SGD classifier.

B. Geometric models

These models use knowledge from geometry such as separating (hyper-) planes, linear transformation and distance metrics to perform classification and regression [17].

a) Support Vector Machines

It finds data points (support vectors) that distinguish between classes [17]. It is highly effective in high dimensional space and where number of dimensions is higher than number of samples.

TABLE IX. SUPPORT VECTOR CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.592

b) K-Nearest Neighbours using manhattan distance

The main principle behind nearest neighbor is to find training samples closest in distance to new point and predict classes from these [19]. Distance is calculated using Manhattan method.

TABLE X. KNN-M CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.843

c) K-Nearest Neighbours using euclidean distance

This is same as above but uses Euclidean method to calculate distance.

TABLE XI. KNN-E CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.845

C. Probablistic model

These are modelled by the means of probability distribution. These models learn by reducing uncertainty in the data [17].

a) GaussianNB

To select most probable hypothesis about the data, we can use prior knowledge. Bayes theorem provides a way by which we can calculate probability of hypothesis on the basis of prior knowledge [20].

TABLE XII. GNB CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.987

D. Tree based model or Logical model

These models are defined in terms of easily interpretable logical expressions. These model use boolean decision making logic and leaves are labelled using majority or proportional system [17].

a) Decision Tree Classifier

In decision tree classifier, to divide the classes, information gain holds a key role. One that gives maximum information gain is selected to divide the classes at each step in the tree.

TABLE XIII. TREE CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.849

E. Ensembles

The main goal of ensemble models is to merge prediction of several base estimator built with particular learning algorithm, to improve generalizability and robustness over single estimator [18].

a) Voting Classifier

It predicts label after performing majority vote on average predicted probabilities, which is obtained by combining conceptually different machine learning classifiers [22]. For this classifier, combination of Ridge classifier, K-NN-M and decision tree classifier is used.

TABLE XIV. VOTING CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.904
RMSE	0.011

b) Bagging Classifier using KNN

It uses aggregation of predictions formed by fitting data on base classifier of random subset, to form final prediction.

TABLE XV. BAGGING CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.854

c) Random Forest Classifier

It fits number of decision tree classifier on various subsample to form final prediction using averaging and controls overfitting of the model.

TABLE XVI. RANDOM FOREST CLASSIFIER ACCURACY

Performance Indicator	Value
Accuracy	0.999

F. Rule based model

The main characteristics of rule-based model is that it identifies and uses a set of relational rules which collectively represent the knowledge captured by the system.

a) Decision table

Decision table are part of rule-based algorithm wherein, concise representation of the system is logically computed by rule formation. These rules can be simply put into If-Else conditions to predict the end result or behavior of the system using knowledge map of rules [24]. There are various rule-based algorithm most popular being Apriori algorithm [25], but decision table works best on numerical data which is suitable for our training and testing set. Section 6 further explores on the results and number of rules formed for the dataset.

TABLE XVII. DECISION TABLE CLASSIFIER PERFORMANCE

Performance Indicator	Value
Correlation coefficient	0.999
Mean absolute error	0.999
Root mean squared error	0.0316
Root relative squared error	4.4275 %

VI. EXPERIMENTAL RESULTS

To compare algorithms a box and whisker plot was plotted for all the models. The data points for box and whisker plot were calculated using 10-fold cross validation with random seed value and their mean accuracy and mean variance across each cross-validation fold, for each algorithm, were plotted to evaluate the range and percentiles.

Below is legend for reference which is used in box and whisker plot to identify different models.

TABLE XVIII. LEGEND FOR MODELS

Model	Legend	Name
Linear models	LGRN	Logistic regression. Solver newton-cg
	LGRS	Logistic regression. Solver saga
	RC	Ridge classifier
	SGD	SGD Classifier
Geometric models	SVC	Support vector classifier
	KNN-M	K-Nearest neighbor using Manhattan distance
	KNN-E	K-Nearest neighbor using Euclidean distance
Probabilistic models	GNB	Gaussian Naïve Bayes
Logical model	tree	Decision Tree Classifier
Ensembles	VOT	Voting Classifier
	BC	Bagging Classifier
	RFC	Random Forest Classifier (*Baseline model)*

A. Linear models

During training with different linear models, it was realized that few models were not meant for this type of data. Models like Logistic regression with solver as newton-cg and saga didn't produce great results. Although the accuracy for both is above 50%, but it failed to decide the decision boundary which can classify on the basis of risk category. It is possible that data simply couldn't be fit by a logistic model and it may produce random results when provided with random data. The confusion matrix for model using saga solver wasn't accurate enough. It wasn't able to predict medium and high-risk category and so its performance indicators were not having values for those categories. Whereas, model with newton-cg tried to predict all values. Table 19 and 20 represents logistic model performance.

TABLE XIX. LOGISTIC CLASSIFIER-NEWTON CG PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.723	0.525	0.595
Recall	0.829	0.527	0.244
F1 Score	0.773	0.526	0.346
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	1884	360	26
Moderate Risk	693	850	69
High Risk	27	406	140

TABLE XX. LOGISTIC CLASSIFIER-SAGA PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.509	0	0
Recall	1	0	0
F1 Score	0.675	0	0
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2270	-	-
Moderate Risk	1612	-	-
High Risk	573	-	-

Now coming onto ridge classifier and stochastic gradient descent, although, out of these two, ridge classifier performed much better and going by the values from performance indicators i.e. Table 21 and 22, it looks much reliable. With SGD classifier we again came across same problem as seen with logistic classifier using saga solver. SGD couldn't predict all classes and values of low risk category are only reflected. Ridge classifier gave an accuracy of 79% whereas, SGD was just 50%.

TABLE XXI. RIDGE CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.850	0.709	0.887
Recall	1	0.746	0.137
F1 Score	0.919	0.727	0.238
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2270	0	0
Moderate Risk	399	1203	10
High Risk	1	493	79

TABLE XXII. SGD CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.509	0	0
Recall	1	0	0
F1 Score	0.675	0	0
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2270	-	-
Moderate Risk	1612	-	-
High Risk	573	-	-

Fig.4 suggests that logistic regression and ridge classifier are much better model for further analysis. But ridge classifier has produced far much better results as compared to logistic regression. All performance based metric values for ridge classifier are reliable whereas, it is not so for logistic regression due to less number and missing values during prediction.

Out of all linear models ridge classifier performed the best. As the magnitude of few coefficients was very high in some of the features, logistic regression didn't work as expected. Upon using Ridge regression, it uses regularization which decreased the magnitude of coefficient. Hence it performed better and can be fine-tuned for further analysis.

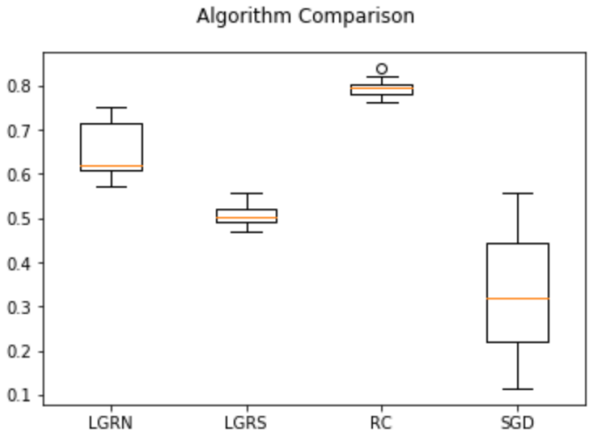

Fig.4. Box plot of linear models

B. *Geometric models*

Geometric models performed far much better than linear models on the whole. None of these models were overfit or underfit. The first model trained was support vector classifier. This model is based on support vector machine and gave a mean accuracy of 52%. Also, the accuracy for this model has very small range of 50% to 59%. Table # describes that its performance was best in classifying high risk category as compared to low and medium risk category. Also, this model was highly confused between actual category of medium risk and predicted category of low risk.

TABLE XXIII. SUPPORT VECTOR CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.557	0.961	1
Recall	0.999	0.217	0.034
F1 Score	0.715	0.354	0.067
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2268	2	0
Moderate Risk	1262	350	0
High Risk	541	12	20

The other two models trained were K-Nearest Neighbors, using Manhattan and Euclidean distance. The first model using Manhattan distance gave pretty good results. The mean accuracy was 70%. Also, this model has the highest recall for low category and very high precision for high risk category. On the other hand, the second model using Euclidean distance has mean accuracy of 65%, which is pretty low as compared to first model. Surprisingly from table 24 and 25, it is worth noticing that both these models got confused while predicting two classes, i.e. actual high with predicted medium risk category.

TABLE XXIV. KNN-M CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.881	0.778	0.912
Recall	0.959	0.818	0.455
F1 Score	0.918	0.798	0.607
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2179	90	1
Moderate Risk	268	1320	24
High Risk	26	286	261

TABLE XXV. KNN-E CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.892	0.771	0.895
Recall	0.959	0.831	0.434
F1 Score	0.924	0.8	0.585
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2177	92	1
Moderate Risk	244	1340	28
High Risk	18	206	249

SVMs are generally meant for binary classification. So, it uses One-vs-rest approach. But in our case, we need to classify amongst three classes. Once trained SVM predicts on the type of data, by majority result among all results of SVM. As we were having comparatively low number of high risk category data, it didn't predict it too well and accuracy average nearly 53%.

Fig.5 suggests that both KNN models performed well, but model using Manhattan distance gave good results as compared to others and should be explored for further analysis.

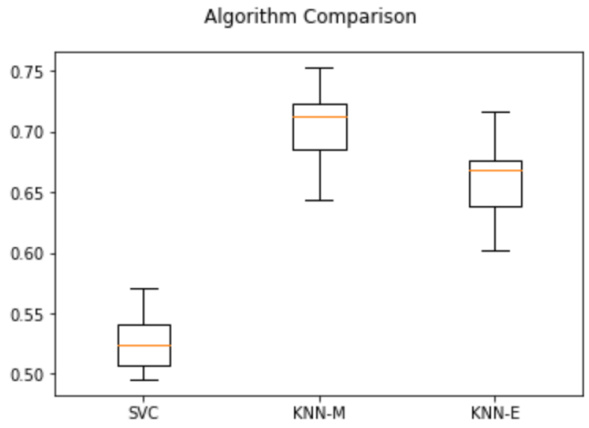

Fig.5. Box plot of geometric models

C. Probablistic models

In probabilistic models GaussianNB was trained. This model performed well and gave mean accuracy of 98%. However, this can also correspond to overfitting of the model. To evaluate overfitting k-fold cross validation using test set was used. In both training and cross validation, the accuracy remains 98%. Which means it was overfitting. To further verify this, confusion matrix was checked if it wrongly predicted some classes and it did. Table 26, shows that model got confused between actual low and predicted medium risk category, and had highest number of true positives, which ultimately means it was leading to overfitting.

TABLE XXVI. GNB CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.99	0.966	1
Recall	0.998	1	0.905
F1 Score	0.999	0.982	0.950
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2267	3	0
Moderate Risk	0	1612	0
High Risk	1	53	519

Fig. 6, The box and whisker plot also suggest that its accuracy stays 98% even with cross validation.

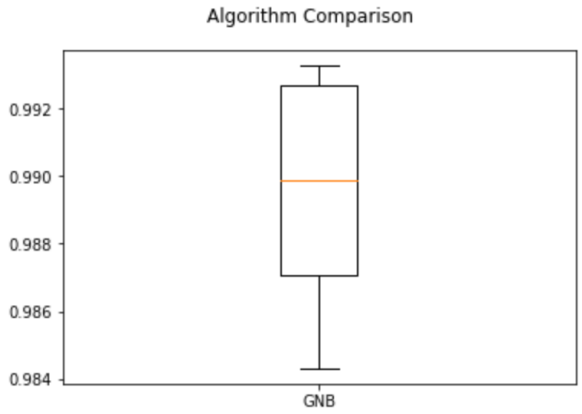

Fig.6. Box plot of probabilistic model

Probabilistic models work best when data is following normal distribution. As the features were not following the normal distribution, this model was overfitting the data.

D. Tree based model or Logical model

To train logical model Decision Tree Classifier with maximum depth of 8 was used. Fig. 7 suggests that, mean accuracy for tree is 94%, where its value ranged from 92% to 98%. Table 27 also confirms that although it misinterpreted some classes but the count was low. Most of the true positive and true negatives were predicted correctly which gave accuracy of more than 95%.

TABLE XXVII. DECISION TREE CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.903	0.779	0.834
Recall	0.981	0.885	0.228
F1 Score	0.940	0.829	0.358
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2227	40	3
Moderate Risk	161	1428	23
High Risk	78	364	131

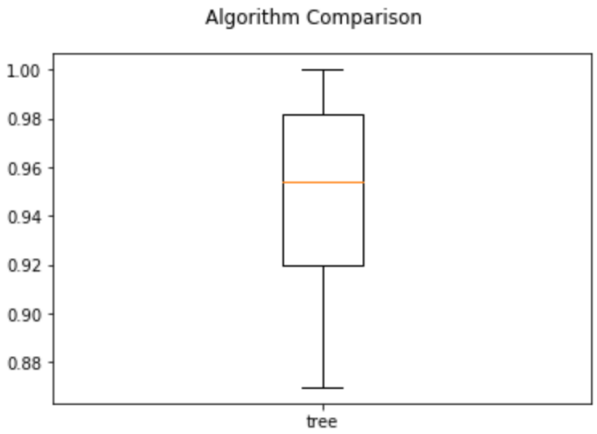

Fig.7. Box plot of tree-based model

Overfitting in Decision tree is pretty common as learning algorithm continues to develop hypotheses that reduce training set error at the cost of an increased test error. To avoid this pre-pruning and post-pruning of trees can be used.

E. Ensembles

To train ensembles three models were used. First one is Voting classifier. This model is a combination of all the previous best models namely, Ridge classifier, K Neighbors classifier and Decision Tree classifier. Voting classifier performed well with mean accuracy of 86%. However, it failed to interpret high risk category and instead predicted it to be medium for few values. Overall the true positive and true negative metrics are decent for this model which can be observed in Table 28.

TABLE XXVIII. VOTING CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.948	0.831	1
Recall	0.998	0.924	0.476
F1 Score	0.972	0.875	0.645
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2267	3	0
Moderate Risk	122	1490	0
High Risk	1	299	273

The second model trained was Bagging Classifier. For this model K Neighbors Classifier using Manhattan distance was used. This model underperformed as compared voting classifier with mean accuracy of 71%. Also, it has pretty low recall value for high risk category i.e. 52%, which can be observed in Table 29.

TABLE XXIX. BAGGING CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.900	0.784	0.887
Recall	0.945	0.844	0.525
F1 Score	0.922	0.813	0.660
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2146	123	1
Moderate Risk	214	1361	37
High Risk	22	250	301

The third model trained was Random Forest Classifier. This was used as baseline model to evaluate the performance of voting and bagging classifier. This model is highly sensitive to distribution of classes. This model is based on logical model. However, like decision tree, this model also overperformed as expected due to imbalanced classes. This model gave mean accuracy of 96%. For the very initial run it gave an accuracy of 100% as well. This is one such model which should not be considered for this dataset. Even the confusion matrix, see Table 30, depicts that it clearly predicted all classes correctly. Random forest can be used after oversampling, undersampling or by applying weights to less frequent class.

TABLE XXX. RANDOM FOREST CLASSIFIER PERFORMANCE

Performance Indicator	Low Risk	Moderate Risk	High Risk
Precision Score	0.999	1	1
Recall	1	1	0.998
F1 Score	0.999	1	0.999
Confusion Matrix			
Actual	**Predicted**		
	Low Risk	**Moderate Risk**	**High Risk**
Low Risk	2270	0	0
Moderate Risk	0	1612	0
High Risk	1	0	572

Out of these three models, Random forest should not be considered. But after careful analysis of box and whisker plot for ensembles, see Fig.8, it can be said that Voting classifier is best for further analysis. It also performed consistently with cross validation and has highest accuracy amongst two models.

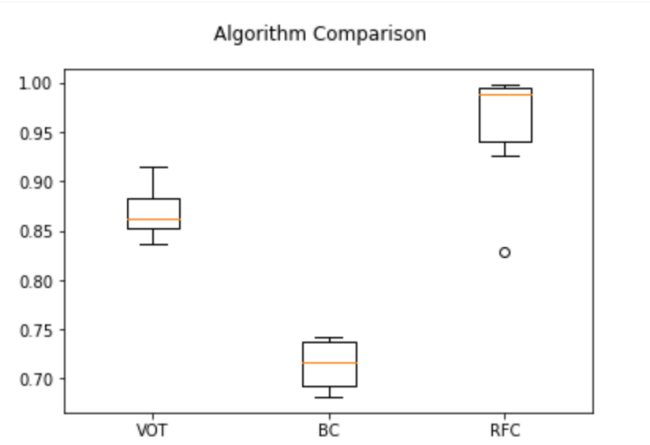

Fig.8. Box plot of ensembles

F. Rule based model

In Rule based model, Decision table was modelled to fit the data with k-fold cross validation. The value for cross validation (10 fold) was kept same as previous models, so as to maintain standard with validation. While searching for rules in dataset Best First search was used. In total 27 rules were discovered which are listed in Table 31. The rules were modelled by automatically identifying violation_id, violation_description and risk_category as important features for this algorithm.

TABLE XXXI. DECISION TABLE CLASSIFIER RULES

violation_id	violation_description	risk_category
'(103109.5-103117]'	'(3.7-inf)'	3.0
'(103147-103154.5]'	'(3.7-inf)'	1.0
'(103124.5-103132]'	'(3.7-inf)'	2.0
'(103139.5-103147]'	'(3.7-inf)'	1.0
'(103117-103124.5]'	'(3.7-inf)'	2.0
'(103169.5-inf)'	'(2.8-3.1]'	1.0
'(103147-103154.5]'	'(2.8-3.1]'	1.0
'(103154.5-103162]'	'(2.8-3.1]'	1.0
'(103139.5-103147]'	'(2.8-3.1]'	1.0
'(103117-103124.5]'	'(2.8-3.1]'	2.0
'(103109.5-103117]'	'(2.8-3.1]'	2.0
'(103124.5-103132]'	'(2.8-3.1]'	2.0
'(-inf-103109.5]'	'(2.8-3.1]'	3.0
'(103132-103139.5]'	'(2.8-3.1]'	1.0
'(103169.5-inf)'	'(1.9-2.2]'	2.0
'(103162-103169.5]'	'(1.9-2.2]'	1.0
'(103109.5-103117]'	'(1.9-2.2]'	3.0
'(103154.5-103162]'	'(1.9-2.2]'	1.0
'(103139.5-103147]'	'(1.9-2.2]'	1.0
'(103109.5-103117]'	'(-inf-1.3]'	2.991825613079019
'(103154.5-103162]'	'(-inf-1.3]'	1.0
'(103132-103139.5]'	'(-inf-1.3]'	1.9837310195227766
'(103117-103124.5]'	'(-inf-1.3]'	2.0
'(103147-103154.5]'	'(-inf-1.3]'	1.0
'(103139.5-103147]'	'(-inf-1.3]'	1.0
'(103124.5-103132]'	'(-inf-1.3]'	2.0
'(-inf-103109.5]'	'(-inf-1.3]'	3.0

In total 28 rules were generated, which comprises of lower range and higher range values of violation_id and violation_description. These range can help in deciding the risk category of the restaurants. Here 1 corresponds to low risk, 2 correspond to medium risk and 3 corresponds to high risk category. Amongst all the rules only two rules were able to classify on fractional basis. If put in practice these two rules can lead to false positives and false negatives.

Overall the performance of model was good with only 0.002 as mean absolute error and 0.03 as root mean square error which is very less and confirms that model will perform better with prediction.

VII. DISCUSSION AND LESSONS LEARNED

To select the best model out of all family of models i.e. Ridge classifier, KNN using Manhattan distance, Gaussian naïve Bayes, Decision Tree classifier and Voting classifier were carefully analyzed. Ridge and KNN both gave a mean accuracy of 70% to 80%. Probabilistic models and Tree-based models gave a mean accuracy of 95%, which looks quite unrealistic. It clearly means that in some respect both these models are overfitting that data. Few possible reasons for this overfitting are:

1. Very few data of high risk category of restaurants. Our models trained well on low and medium category of data due to large number of rows as compared to high category. Also, in reality, it is unlikely that most of the restaurants would fall under high risk category.
2. Most of the data, classified under high risk got eliminated at the time of pre-processing. This is because data is recorded in a casual way, wherein phone number and addresses of most restaurants were missing. For some data, the inspection score was missing which is an important feature to classify the restaurants. In a way it contributed to class imbalance.
3. Data was not normally distributed. Due to this probabilistic model also failed to predict risk categories and as a result they were overfitting the data.

So, both Probabilistic and Tree based models are not suitable for classifying restaurants. One possible reason for the same is class imbalance and not following normal distribution. We have a greater number of low and medium category of restaurants as compared to high risk category. To use these kinds of model, it is better to either oversample or undersample the data. Another technique which can be used is by applying greater weights to less frequent class.

When ridge classifier is compared with KNN, Ridge classifier performed much better, the mean accuracy for ridge classifier is 79% whereas, for KNN it is 70%. Out of these two it is best to consider ridge classifier for this task.

At last, we have Voting classifier which is combination of best models from each category and decision table. When voting classifier is compared with ridge classifier, its performance is much better i.e. 86%. It also performed well with cross validation. So, it is fair to say that Voting classifier is best suited for this dataset, and should be considered for further study. But when the same data is modeled using decision table we have very low root mean square error. Moreover, there is not much difference between RMSE of voting classifier and decision table. For voting classifier RMSE accounted for 0.011 and for decision table it is 0.031. Lower the root means square value better the performance of model. So, to further extend analysis and training, voting classifier and decision tables should be used.

Additionally, to improve the models, the source dataset can have some mandatory values as not null at time of data entry. This would enhance the training of models and will

provide models with wide variety of different coefficients to learn.

VIII. Conclusion

This experiment tried to classify restaurants into three risk categories based on the data shared by Government of San Francisco. The dataset comprises of geographical location, inspection scores and violations by restaurants. However, the data which is LIVE i.e. updated daily and is not clean contains lots of features which are not important, messy merely acts like noise for model during training. The data was pre-processed and was made suitable for training on various family of models.

During selection of features, inspection score got the highest score, whereas day, month and year got the least. At last, few year and month columns were not part of training data.

Models from all the families were considered and trained. It was observed that few models were overfitting the data and some were not able to calculate the decision boundary for classification at hand. However, ridge classifier and geometric models worked perfectly for this dataset. Although, there were some problem with interpretation of some classes, but the mean accuracy for both of them was above 70%. At last ensembles and rule-based models were trained. Voting classifier which was modelled using best of previous models worked best for this dataset, and decision table had the least root mean square value. These two models should be trained further with more improved dataset for further study.

IX. Future work

This dataset can be used identify restaurants falling under high risk category beforehand, and proper checks and measures can be applied. This would further minimize the chance of food borne diseases and will help in maintaining health and safety standards for the city. To further extend work and knowledge from this dataset one can:

1. Predict most desired locations for restaurant to exist.
2. Predict inspection score provided data is not messy.
3. Predict risk category based on location of the restaurants.
4. Predict what violation most restaurants are likely to commit.

All the above problems can help leveraging business, health and safety. Further one can analyze most common pattern of the restaurants while committing violations.

X. References

[1] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3323064/
[2] Olsen SJ, MacKinnon LC, Goulding JS, Bean NH, Slutsker L Surveillance for foodborne-disease outbreaks, United States, 1993–1997. MMWR CDC Surveill Summ. 2000;49:1–62
[3] https://datasf.org/
[4] https://datasf.org/about/
[5] https://data.sfgov.org/Health-and-Social-Services/Restaurant-Scores-LIVES-Standard/pyih-qa8i
[6] https://machinelearningmastery.com/k-fold-cross-validation/
[7] https://datascience.stackexchange.com/questions/9167/what-does-rmse-points-about-performance-of-a-model-in-machine-learning
[8] https://blog.minitab.com/blog/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit
[9] https://machinelearningmastery.com/confusion-matrix-machine-learning/
[10] https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234
[11] https://developers.google.com/machine-learning/crash-course/classification/accuracy
[12] https://machinelearningmastery.com/compare-machine-learning-algorithms-python-scikit-learn/
[13] https://blog.minitab.com/blog/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit
[14] https://scikit-learn.org/stable/modules/linear_model.html#linear-model
[15] https://docs.aws.amazon.com/machine-learning/latest/dg/linear-models.html
[16] https://scikit-learn.org/stable/auto_examples/linear_model/plot_bayesian_ridge.html#sphx-glr-auto-examples-linear-model-plot-bayesian-ridge-py
[17] Herna L Viktor, CSI5155 Machine Learning, Topic1_IngredientsML, slide 17-28
[18] https://scikit-learn.org/stable/modules/ensemble.html
[19] https://scikit-learn.org/stable/modules/neighbors.html#neighbors
[20] https://machinelearningmastery.com/naive-bayes-for-machine-learning/
[21] https://machinelearningmastery.com/logistic-regression-for-machine-learning/
[22] https://scikit-learn.org/stable/modules/ensemble.html#voting-classifier
[23] https://colab.research.google.com/notebooks/welcome.ipynb
[24] https://en.wikipedia.org/wiki/Decision_table
[25] R. Agrawal, T. Imieliński and A. Swami, "Mining association rules between sets of items in large databases", ACM SIGMOD Record, vol. 22, no. 2, pp. 207-216, 1993. Available: 10.1145/170036.170072
[26] https://www.cs.waikato.ac.nz/~ml/weka/