Capstone Project 1

Boston Fire False Alarm Prediction

Summary

This report analyzes Boston Fire Alarm Incidents and establishes a predictive ML algorithm to predict these incidents, which helps Boston Fire Department improve operational efficiency. Five parts are included in this report: Project Overview, Data Wrangling, EDA Analysis, ML Algorithm Development and Conclusions/Deliverables. Source code is shared here.

Project Overview

- **Background**

Boston fire department receives various alarms every day. Some alarms are urgent and might lead to disastrous incidents, which require more firefighters. However, more reports are minor, even false or unwanted, and fewer firefighters are needed. So types of fire incidents has a great impact on Boston fire department's resource distribution and operational efficiency.

- **Problem**

Among eight Incident Type, False alarm often incurred a lot of time and labor costs. In order to improve the operational efficiency, Identification of the false alarm effectively becomes the first question. So this project focuses on how to identify the false alarm.

- **Client**

Boston Fire Departments wants to improve their operating system efficiency to deal with fire incident reporting. As a data scientist in a consulting company, I will analyze data, gain insights from data, build an algorithm to predict false or true alarm type for Boston fire department to improve their operations.

- **Approach**

I will extract data mainly from monthly Boston fire incident reports from 2012-2018. Meanwhile, data from daily historical weather, and community population density would be useful. (So far, The entire incidents would be about 350,000). Based on zip code and address, these data would be merged into one. The main features in merged data would contain incidents reporting time, location(zip code, district, address etc.), alarm type, property owners information, property assessment results, daily weather information, local population density and nearby events. Through EDA, those data would be visualized and more stories would be discovered. More insights regarding of incident types would be obtained. In predictive model, other alarm types except False Alarm would be aggregated as one type: True Alarm. Then the predictive model can effectively identify the False Alarm.

- **Deliverables**

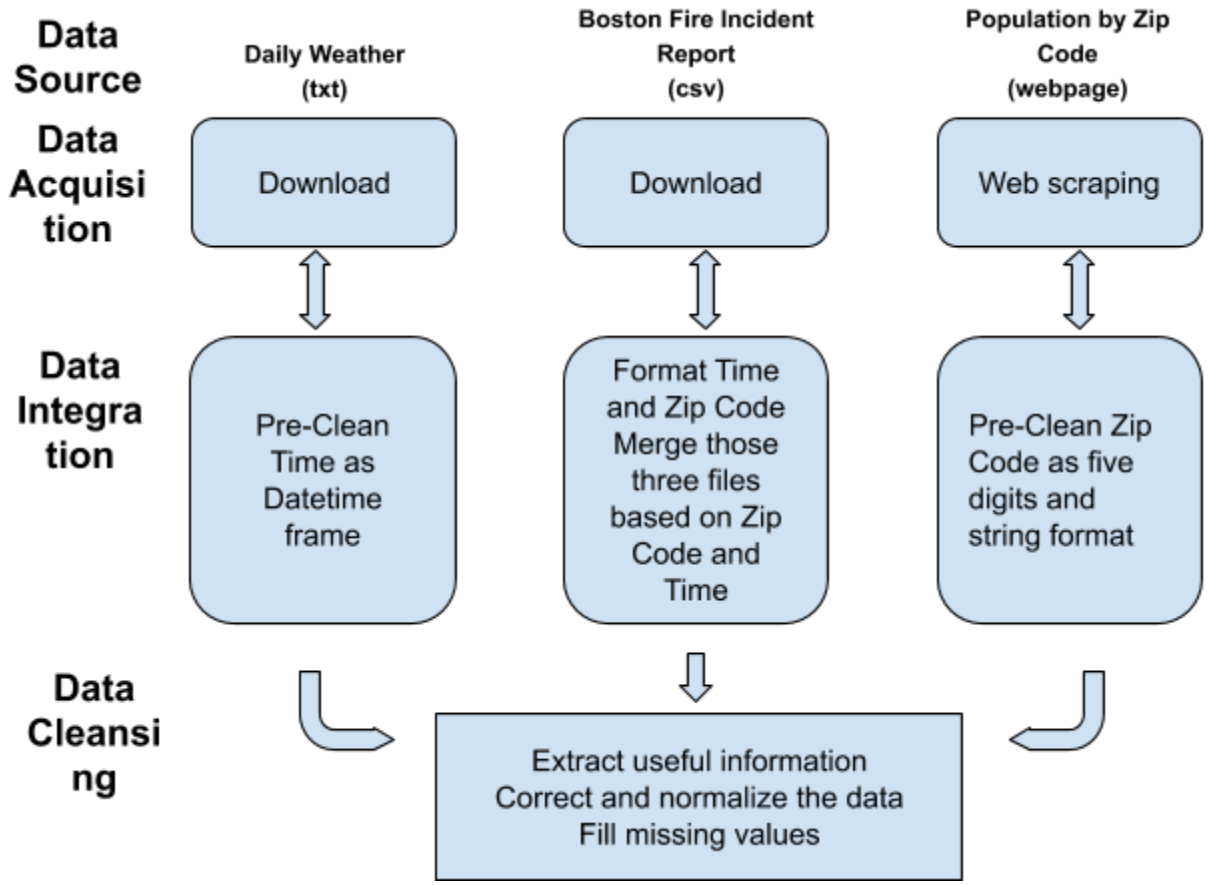
Source code will be displayed in Github A report and powerpoint slides would be shared. And I will post the results in Medium (Hopefully, I can try to deploy the model in AWS or Google Cloud or build an API for the dataset etc.)

Data Wrangling

In this part, I will explain how to collect, wrangle and clean data for Capstone Project 1, Boston Fire Alarm Type Prediction.The goal for this step is to get clean data and prepare for data analytics and prediction model. There are five steps: Data Acquisition, Data Integration, Data

Extraction, Data Normalization and Data Correction. The data sources come from three files: Fire Alarm Incident Report, Hourly Weather and Community Population.

Data_Preparation.ipynb completes the first two steps while Data_Cleansing.ipynb completes the last three steps. Source code can be found here. I will explain these steps in detail. The process will be followed as the figure below.



- **Data Acquiring**

Data are downloaded individually from Boston Fire Incident Report(https://data.boston.gov/dataset/fire-incident-reporting), Historical Boston weather(http://www.frontierweather.com/historicaldataonly/KBOS.txt) and Population Community (http://zipatlas.com/us/ma/zip-code-comparison/population-density.5.html)

- **Data Integration**

1. Format data: Zipcode must contain five digits only and first two digits must be “01” or “02”. Date must be datetime frame format: Month/Day/Year Hour.
2. Data Integration: Based on time, Fire Incident would be merged with hourly weather. Then the data from population would be merged based on zip code.

- **Data Extraction**

We need useful information from some columns, such as Incident Type, District and Weather. For example, to simplify the problem, the first number is extracted to represent the Incident Type. Meanwhile, non-digit characters in Incident Type column need to be converted to empty blank.

- **Data Normalization**

Some columns having various representations need to be converted to the same format, such as Main Address and Address 2. We need to combine Street Number, Prefix, Type, Suffix. Some columns have to be dropped

- **Data Correction**

Some rows with missing values must be removed, such as Incident Type. Some missing values might be filled as 0 or mean values such as Precipitation, Population and Population Density. The NaN of text columns would be filled as the string “None” or “Unknown”.

- **Data Validation**

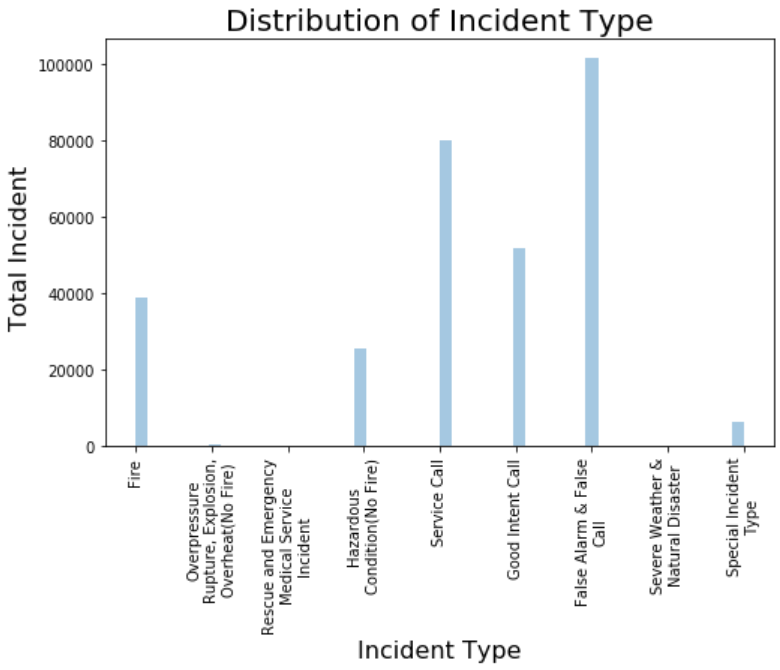
After the steps above are finished, we can check the dataset if they are ready for analysis. We check it by single column row and statistical approach. Every item fall into their categories as required.

EDA Analysis:

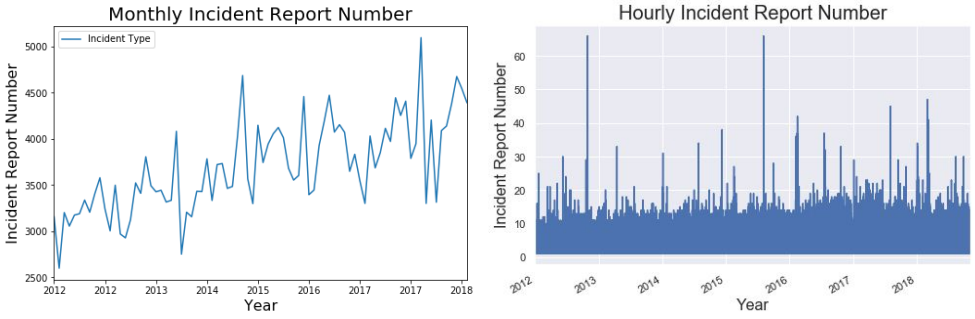
- **Graphical Analysis**

  Through graphical analysis, a few conclusions can be summarized as follows. More detailed analysis can be found in the powerpoint slides attached.

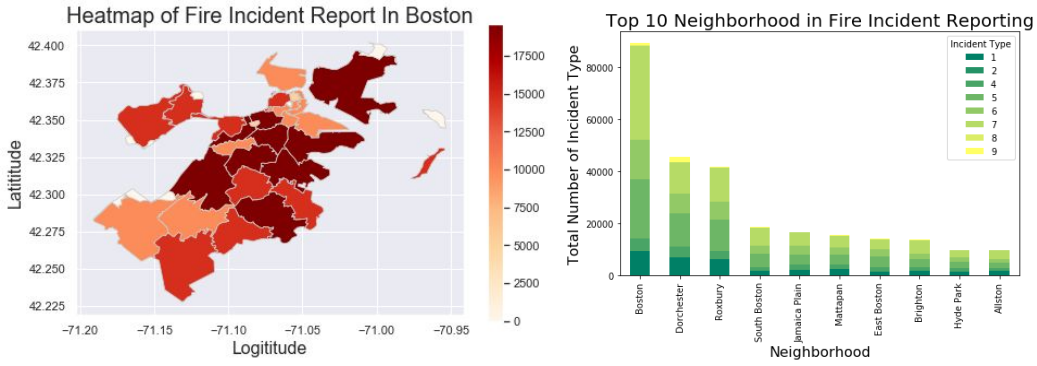
  - The top three Incident reporting type is False Alarm and Fake Call, Service Call and Good Intent Call,



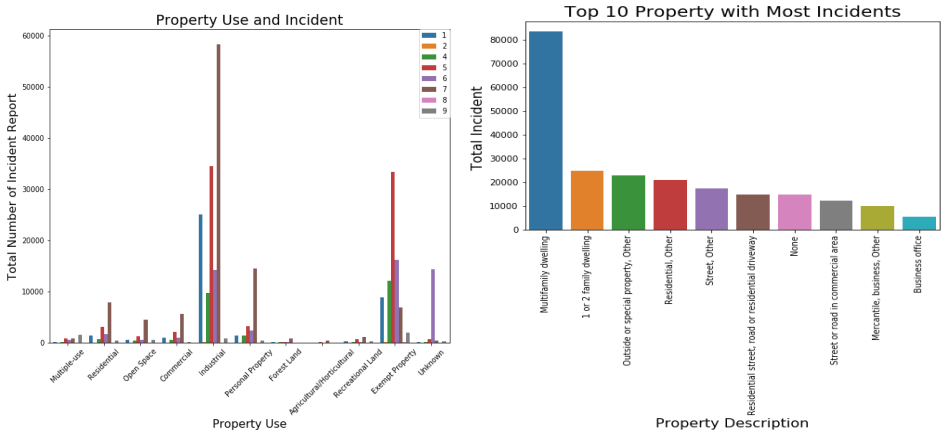
- The real time incident report does not change while the monthly count of incident reporting is increasing every year.



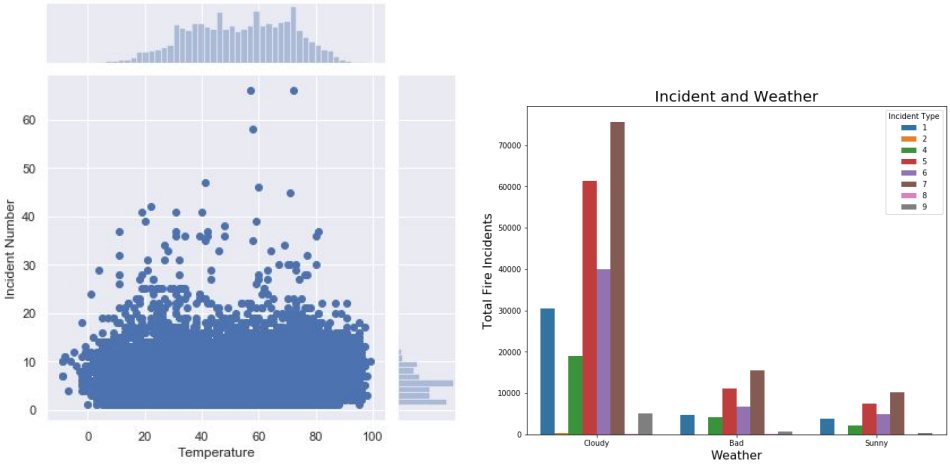
- The top area of Incident reporting is Boston, Dorchester and Roxbury



- ❖ The top three property with most incident reporting is Industrial, Recreational Land, Personal Property



- ❖ Weather is an important factor on Incident Reporting.



- **Statistical Inference:**

From the previous graphical analysis, we have further questions to answer:

1. Is real-time Incident Type time dependent?
2. Is the monthly counts of Incident reporting time dependent?
3. Does Temperature really correlate to Incident Type?
4. Does Temperature have linear correlation with Dewpoint?

We will answer questions through Hypothesis Testing. So we start from Hypothesis first.

- ❖ <u>Hypothesis:</u>

In order to answer these questions, we need to perform validate our assumptions by hypothesis. We start from hypothesis respectively.

Question 1:

Null Hypothesis H0: Real-time Incident type is time dependent

Alternative Hypothesis Hα : Real-time Incident Type is not time dependent

Question 2:

Null Hypothesis H0: Monthly counts of Incident reporting is time dependent

Alternative Hypothesis Hα : Monthly counts of Incident reporting is not time dependent

Question 3:

Null Hypothesis H0: Temperature does correlate to Incident Type

Alternative Hypothesis Hα : Temperature does NOT correlate to Incident Type

Question 4:

Null Hypothesis H0: Temperature has NO linear correlation with Dewpoint

Alternative Hypothesis Hα : Temperature has linear correlation with Dewpoint

- Hypothesis Testing:

Since Question 1 and 2 are related to time, Adfuller Testing is used. Question 3 is correlation testing, so we use Chi Square testing. For question 4, Pearson correlation is performed.

- Testing Results:

Question 1: α = 0.0 is less than 0.05 (significance level). So we reject Null Hypothesis H0 and accept Alternative Hypothesis, which means Real-time Incident Type is not time dependent

Question 2: α = 0.856808 is greater than 0.05. Null Hypothesis fails to be rejected. Real-time Incident Type is not time dependent

Question 3: α = [0.92368965 0.45178618 0.52189569 0.46904684 0.88481471 0.98657539 0.99999469 0.55348292] for each incident type is greater than 0.05. Null Hypothesis is accepted. Temperature does correlate with Incident Type.

Question 4: α = 0.0 is less than 0.05 and correlation coefficient is 0.890731345258. That means Temperature has a linear correlation with Dewpoint

Modelling:

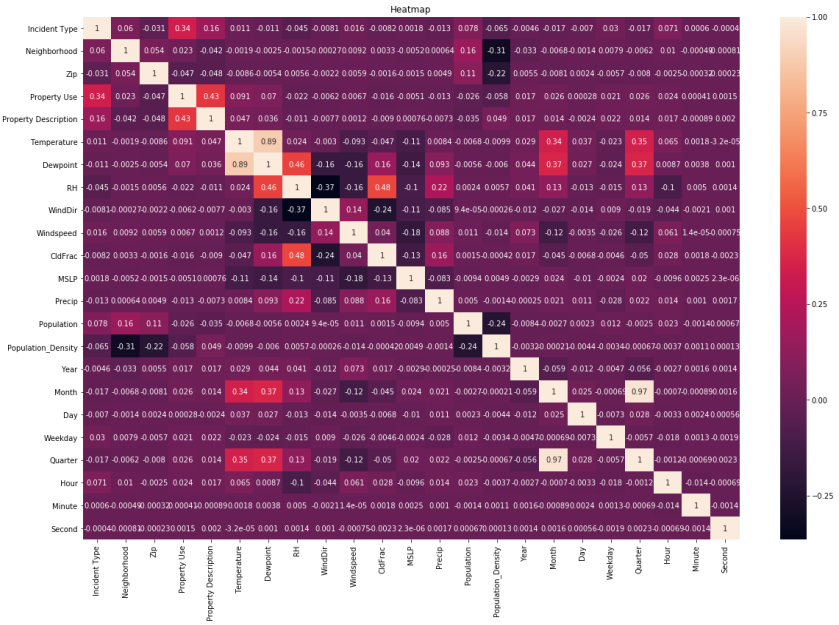
After data is prepared, a machine learning model will be built to classify the Incident Type. Six steps are followed:

- **Problem Tweaking**

  For the original problem of classifying every Incident Type, The dataset is very imbalanced (This was supposed to be discovered in data exploration steps). So we have to group different datasets. From a business perspective, the original problem is to predict the Incident Type to improve the operational efficiency . False Alarm is the biggest concern. So the original problem is converted to a binary classification problem: Classify the False Alarm and True Alarm. So this conversion can be effectively to improve the operational efficiency.

- **Feature Selection**

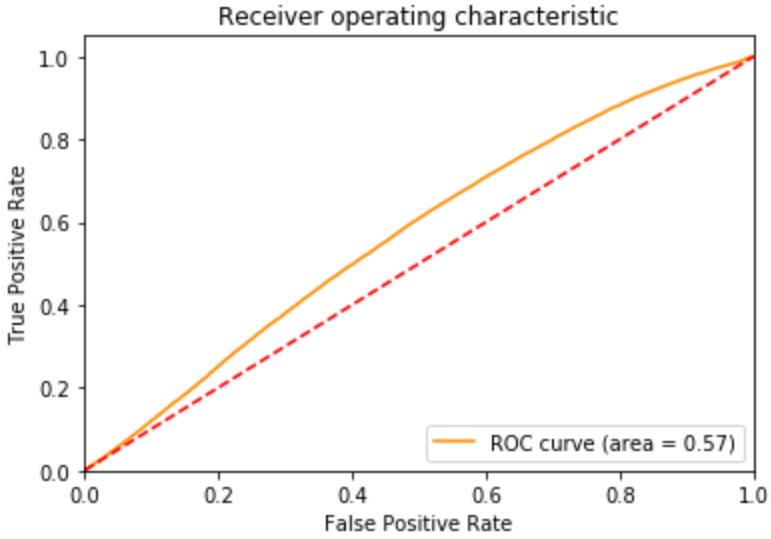
  At the beginning, I extract features from time variables as many as I can. Then heatmap is used to select feature selections. Property Use and Description are the most important features. Additionally, Neighborhood, Population and Hour are less important.
  We also need to remove some features. For example, we need to remove Minutes and Seconds, which are much less relevant to the target Incident Type. Some variables have high very collinearity, such as Temperature and DewPoint. We also need to remove DewPoint. This process is iterated with the following steps.



Eventually, we selected categorical features: 'Neighborhood', 'Property Use', 'Property Description', 'Weekday' and numerical features: 'Temperature', 'Windspeed', 'Relative Humid', 'Precip','Population', 'Population_Density', 'Hour'.

- **Baseline**

  The value of False Alarm is assigned as 0 and the values of True Alarm Type are assigned as 1. So we use Label Encoder to convert categorical variables to numbers and scale numerical variables. Then we use Logistic Regression as the baseline, since it is simple. Every default value is kept for the model and cross validations (KFold=5) is applied. Then we can get accuracy of test score: 0.6540. For the ROC curve, this model barely has the ability of classifying False Alarm.



- **Model Comparison**

  Considering the computing limitations, we are selecting six models with three feature engineering approach. We use cross validations (KFfold=5) for each model. The result is shown below: (The accuracy is all for test score)

Table 1. Model Performance

	Ordinal Encoder (11 features)		Dummies (100 features)		Target Encoder (11 features)	
	Accuracy (Avg.)	Time (Seconds)	Accuracy	Time	Accuracy (Avg.)	Time

Logistic Regression	0.671021 (Baseline)	4.21	0.711583	77.14	0.703140	0.94
Linear SVM	0.669355	222.23	0.710618	595.71	0.702514	106.24
Linear Discriminant Analysis(LDA)	0.668759	2.7	0.710908	22.72	0.703239	0.43
Gaussian Naive Bayes	0.666259	1.15	0.690258	10.84	0.703407	0.3
AdaBoosting	0.687943	70.8	0.710898	299.77	0.709084	28.35
AdaBoosting(Logistic)	0.668512	90.48	0.694552	265.39	0.702850	49.59

From the table, we can see Adaboosting Classifier performs best in terms of accuracy. But it is also the slowest model. Apparently, Gaussian Naive Bayes is the fastest model. Logistic Regression and LDA have pretty decent accuracy and speed.
It also shows feature engineering is very important. Dummies and target encoder have much better accuracy than Ordinal. With the consideration of computing and accuracy. We will go for Logistic Regression with dummies.

- **Model Selection**

  Considering the accuracy and running time, the model with LDA and Logistic Regression are preferred from the table above with less running time and high accuracy. LDA can be also used as feature reduction instead of Principal Component Analysis. Logistic Regression can be used as base_estimator for AdaBoosting. So the pipeline of our improved model can be LDA for feature reduction and Logistic Regression as the base estimator (Weak classifier) , then AdaBoosting Classifier as ensembler. The accuracy of testing data is 0.71227139.

  But LDA is very difficult for interpretation. If interpretability is also considered, we might select PCA as feature engineering approach and Logistic Regression as modelling approach.

- **Hyperparameter Optimization**

Now we need to optimize hyperparameters for the pipeline built above. The hyperparameters to be optimized are learning rate and n_estimator. We choose GridSearch as the optimizer. The optional is n_estimators = [10, 50, 100]
Lr (learning rate) = [0.1, 1, 10], cv = [3, 5, 10]. The optimal parameters are learning_rate = 0.1, n_estimator = 10 and cv=10. The final result shows below.

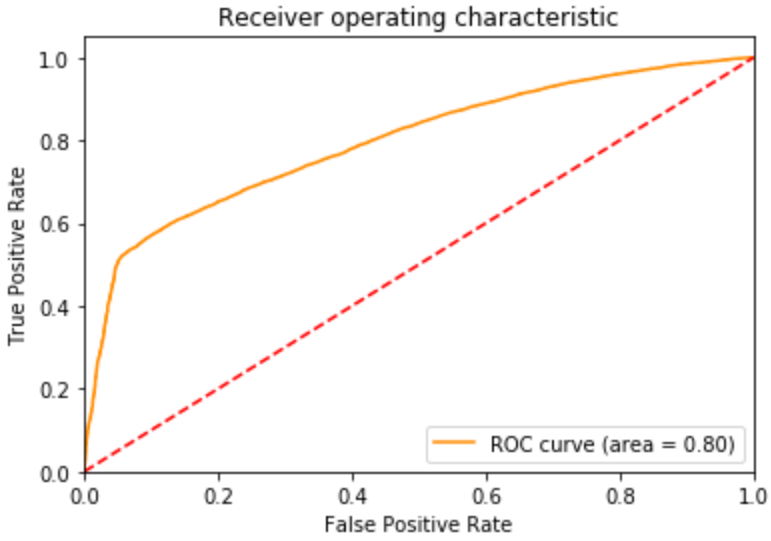
Test Score of Model: 0.7134812961011591

Confusion Matrix

[[3888 6838]
 [1863 17779]]

Classification Report

	precision	recall	f1-score	support
0	0.68	0.36	0.47	10726
1	0.72	0.91	0.80	19642



The result shows that the accuracy improves greatly. The recall of 0 (False Alarm) and f1-score increases and ROC curve lifts. That shows our model improves a lot.

- **Result Analysis**

  Now we can analyze the results and evaluate our model further.

  Wrong Predictions:

  The first is False Negative. The model mistakenly sees True Alarm as False Alarm, which is also our biggest concern. We can see the results below that it often happens in the Property 1, Residential Use. From the confusion matrix, we can see the type of errors are relatively low.

	Incident Type	Predicted Incident Type	District	Neighborhood	Zip	Property Use	Property Description	Temperature	Dewpoint	RH	...	Population	Population_Density
273350	1	0	11	Allston	02134	5	Mercantile, business, Other	40	27	59	...	21389.0	15139.35
273380	1	0	9	Roxbury	02115	3	Hospital - medical or psychiatric	34	32	92	...	25486.0	30823.24
273395	1	0	4	Boston	02115	1	Assembly, Other	35	25	67	...	25486.0	30823.24
273443	1	0	4	Boston	02116	1	Eating, drinking places, other	33	24	69	...	19682.0	26352.26
273450	1	0	4	Boston	02116	1	Eating, drinking places, other	33	25	72	...	19682.0	26352.26

The first type of result is False Positive. The prediction is 0 (False Alarm) but the true value is 1 (True Alarm). This case often happens in Industrial Property (Property Use 4). Those properties are often located in Boston Dorchester etc. We may need other variables to reduce the error, such as sensor data from those buildings.

	Incident Type	Predicted Incident Type	District	Neighborhood	Zip	Property Use	Property Description	Temperature	Dewpoint	RH	...	Population	Population_Density
273315	0	1	7	Dorchester	02121	4	Multifamily dwelling	44	27	51	...	25057.000000	19592.250000
273328	0	1	3	Boston	02203	1	Rapid transit station	41	28	60	...	25516.718868	16182.523225
273330	0	1	11	Brighton	02135	4	Multifamily dwelling	41	28	60	...	43887.000000	14809.880000
273336	0	1	4	Boston	02115	4	Multifamily dwelling	40	28	62	...	25486.000000	30823.240000
273339	0	1	8	Dorchester	02124	4	Residential, Other	40	26	57	...	50781.000000	16127.900000
273344	0	1	3	Boston	02114	4	Residential, Other	40	27	59	...	10868.000000	19975.480000

Correct prediction:

When the correct value is predicted, the Neighborhood and Property Use are more diverse. That means our model can make predictions with broader variables pretty well.

	Incident Type	Predicted Incident Type	District	Neighborhood	Zip	Property Use	Property Description	Temperature	Dewpoint	RH	...	Population	Population_Density
273314	1	1	8	Mattapan	02126	4	Residential, Other	44	27	51	...	27815.0	13523.15
273316	1	1	9	Roxbury	02121	4	1 or 2 family dwelling	44	27	51	...	25057.0	19592.25
273318	1	1	7	Roxbury	02119	4	Multifamily dwelling	42	26	53	...	23823.0	14856.36
273319	1	1	11	Allston	02134	1	Playground	42	26	53	...	21389.0	15139.35
273320	1	1	9	Roxbury	02120	9	Street, Other	42	26	53	...	10842.0	18116.78

	Incident Type	Predicted Incident Type	District	Neighborhood	Zip	Property Use	Property Description	Temperature	Dewpoint	RH	...	Population	Population_Density
273317	0	0	6	South Boston	02210	7	Manufacturing, processing	44	27	51	...	592.0	757.88
273329	0	0	3	Boston	02111	4	Hotel/motel, commercial	41	28	60	...	5138.0	15967.11
273341	0	0	3	Boston	02114	1	Bar or nightclub	40	26	57	...	10868.0	19975.48
273342	0	0	3	Boston	02114	1	Bar or nightclub	40	26	57	...	10868.0	19975.48
273348	0	0	11	Brighton	02135	5	Mercantile, business, Other	40	27	59	...	43887.0	14809.88
273352	0	0	8	Hyde Park	02126	1	Church, mosque, synagogue, temple, chapel	40	27	59	...	27815.0	13523.15

Future Work

- Collecting more data, especially data about the building and location. We can geocode address, then find more location data.
- Feature Interpretation. In this model, LDA does not provide functions on interpretations. So we have to interpret it manually, which can become next step.
- We can try more feature engineering techniques, such as polynomial etc.
- With higher computing, we can try more models, like LightGBM, XBoostGBM and Neural Network.
- Besides the False Alarm, we can multi-classify the True Alarm.