Crime in Chicago

Shuting Gu
First Year Master
Data Science
NYU
sg5686@nyu.edu

Zihao Guo
First Year Master
Data Science
NYU
zg866@nyu.edu

Anshan He
First Year Master
Data Science
NYU
ah4734@nyu.edu

Ziyu Lei
First Year Master
Data Science
NYU
zl2350@nyu.edu

Abstract

Exploratory Data Analysis (EDA) plays a crucial rule in the crime analysis, especially Heat Maps. This technique reveals the aggregated crime locations and types over a certain period of time. Our first goal is to estimate when and where crimes will happen and EDA helps us visualize the result. The second goal in our project is to predict the type of a crime given the time and location that crime might happen. We proposed several models to tackle this problem, where XGBoost is superior in quality but requires longer training time. We achieved 41.8% F1-score for XGBoost.

1 Introduction and Motivation

Like many other large metro areas, Chicago has an overall high crime rate. Although its overall number of crimes declines in the 2000s, there is a rebound in overall murders the mid-2010s (Wikipedia, 2018a). Our project will focus on the crimes in Chicago from 2012 to 2016. In this period, there is a decline in the number of crimes happening per month. And seasonality is present. There are more crimes committed in summer months but less in the winter period (see Graph A).

Compared with other cities in the US, Chicago has long been notorious for frequent felonies including murder and rape. Besides the types mentioned above, other types of crimes, such as theft, battery, and criminal damage also happen frequently in Chicago (see Graph B). Needless to say, its responsible for almost half of 2016s increase in homicides in the US. Hence, it is urgent and imperative that Chicago builds a system or model to analyze the crimes data, derive insights from different features of each crime, and predict when, where and how a crime might happen in the near future.

In this project, we are going to identify when and where crimes are most likely to happen in Chicago and specifically predict the type of potential crime given the time and location. This project can benefit different parties. Chicago Police Department would be able to arrange their resources accordingly and increase patrol in high-crime-rate locations over certain periods. The government can also benefit from this research if it's possible to figure out the root cause of a compact region/time of crimes so that it can launch various programs to facilitate the development of a safe neighborhood, such as public schools, shelters, affordable insurance, harsh penalty on drug dealers, etc.

2 Data Exploration

2.1 Raw Data

Our raw data has 1456714 instances and contains 23 variables, downloaded from Kaggle.com (Kaggle.com, 2018). There are 4 different variables that relate to the ID of each crime, 3 variables that show the time associated with crimes, 13 variables that describe the crime locations based on different classification systems, 2 variables that describe the type of each crime, 1 variable that shows whether the suspect was arrested or not. It is obvious that we cannot choose all variables because collinearity would exist and the computational cost would hit up the ceiling for the size of crime data we will have in practice.

2.2 Exploratory Data Analysis

Heat map is an eye-catching method that can be used to show the relative frequency and intensity of our data. In our project, we plot heat maps for crime frequency in different combinations of location and time. If the crimes are committed more frequently at the specific time and location, the value of the corresponding cell will be higher, which will result in a ”hot” color.

From the heat maps, we can intuitively observe that the crime frequencies vary significantly accompanied with the change of time and location. We first drew a heat map (see Graph C) to display the number of crimes happening in a specific location and at a particular time in a day. We can conclude that the whole city might be relatively safe during midnight and early morning (1am-7am). However, in the afternoon (12pm-5pm), there are more crimes committed in some public area like restaurants and gas station. As for outdoor places like street and sidewalk, they are more dangerous in the evening (6pm-10pm). When it comes to days in a whole week, the heat map (see Graph D) indicates that Monday to Thursday is much safer while Friday is most dangerous day of the week. Generally, outside activities during weekends may face more risks.

3 Models

3.1 Target

Although the overall number of crimes declines since 2012, the tendencies for different types of crimes are various (see Graph E). Therefore, it is meaningful to predict the *Primary Type* of each crime given the information about when and where the crime is committed.

The target variable *Primary Type* has 33 distinct values, meaning we have 33 crime types. After using heat maps to analyze our data, we found that location, time and domestic (inside or outside of buildings) information is closely related to the crime types (see Graph F and Graph G). Thus, those features including relevant information will be considered in our model.

3.2 Features

After scrupulous features inspection based on the issue we were facing, 4 features were picked out of 22. These variables are *Date*, *Location Description*, *Domestic* and *Community Area*. We further divided *Date* into 5 variables which are *Year*, *Month*, *Day*, *Time*, and *Day of Week*. More detailed date information will be given without bringing so much collinearity because time is a hierarchical variable. For example, knowing *Year* doesn’t give us any information about *Month*. This step in our data processing provided us with a potential to pinpoint the type of crime in a specific time of a day in a certain month.

3.3 Feature Engineering

We first split our data into training and test sets. For the target variable *Primary Type* in training set, we transcribed it from string to integer and we created a dictionary of the target to keep tracking what crime type that each number represents. If there is any type in test set but not in training set, we just encode that type into value 0. *Location Description* is a nominal feature, so we decided to create dummy variables for it. However, there are 142 distinct values in this feature and it would be computationally expensive for such a huge dimension, so we only left the top 25 most frequently occurred values in *Location Description* and classified less frequent values into a newly defined class ’OTHER’. If there is any location in the test set not shown in train set, we just encode it into 0.

3.4 Baseline Model – Logistic Regression

We’ll use logistic regression as our baseline model because it can be easily interpreted for the classification problem and doesn’t require a lot of assumptions like a linear model. Some of the important assumptions are: small amount of collinearity, independent instances (no repeated records), linearity of independent variables and log odds, etc (statisticssolutions, 2018). We’ll use L1 penalty (Lasso) in regularization to avoid the overfitting caused by the multi-collinearity and filter out those less important variables.

3.5 Decision Tree

Just like Logistic regression, decision tree is easy to understand and interpret. It can handle both numerical and categorical variables, which is our case. We’ll use ’Gini Impurity’ as our metric for measuring each split, which is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset (Wikipedia, 2018b). Decision Tree usually fits large data set well, which is suitable to our case here. However, decision tree could be unstable, which means small variance in data can cause a totally different tree structure. Also, decision tree deploys greedy search over the parameter space, which yields local optimal solution but doesn't generate the global optimal tree. Some other tree algorithm that uses beam search should be preferable.

3.6 Random Forest

Random Forest is an ensemble learning method. It uses bootstrap aggregating technique to randomly select a sample from the train set with replacement. Then, it randomly chooses a certain number of features at each terminal node which are used for building a tree iteratively. We can build as many trees as we want but a large number of trees won't increase the accuracy but the training cost. The advantage of random forest is that the instability in decision tree is compromised by building more trees and random feature selection. We'll use 'Entropy' as our metric to determine each node split.

3.7 KNN

KNN is an instance-based learning method, which means it doesn't really perform explicit generalization on training data but just store training set in memory. KNN calculates the metric (distance) between an observation in the validation set and all observations in training set, and determines the label of that record based on its nearest K neighbors. KNN performs well in data with a relatively small dimension. However, if the data has a large dimension, which is our case, then the distance between some pairwise instances would be pretty large. The distance would be dominated by the features with large magnitude, thus neglecting effects of other important variables. Hence, for KNN, we normalized our features into the similar scale so that KNN can perform relatively well.

3.8 XGBoost

XGBoost is a gradient boosting algorithm, which means an ensemble of decision trees. Unlike random forest that builds separate trees and then takes a majority vote, XGBoost adds tree iteratively to the previous tree based on the residuals of the previous tree. Every time it fits a tree to data, calculates the residuals and then fits a new tree to model the residuals. This process is repeated until the specified number of trees are aggregated. As an expansion, we hope to apply in our project to improve our model performance.

4 Train and Evaluate

4.1 Cross-Validation

We'll use cross-validation to tune our parameters and find the best model settings. **scikit-learn** provides a exhaustive search function *GridSearchCV*. Then, we wrote a pipeline function to find the best model setting and the mean F1 score of K-fold cross-validation.

4.2 Evaluation Metric – F1 Score

Our target is a multi-class variable and the class is imbalanced, so we don't want to use accuracy as our evaluation metric. The dominating classes will make the accuracy deceptively large. Instead, we use F1 score, which is a harmonic average of precision and recall. This metric takes both accuracy and misclassification into account, thus our algorithm won't predict all instances with dominate class (Quara.com, 2018).

4.3 Logistic Regression

Since logistic regression is our baseline model, so we only tune the regularization weight **C**. The best setting is actually the default value 1 for the weight.

4.4 Decision Tree

In decision tree, we used greedy search and 4-fold cross validation algorithm to tune our 4 parameters: *min_samples_split* = [100, 200, 300, 400, 500, 600, 800, 1000], *min_samples_leaf* = [25, 50, 75, 100, 300, 400, 500], *max_features* = [None, 'auto', 'log2'] and *class_weight* = [None, 'balanced']. The model with the best setting is as follows:
DecisionTreeClassifier(class_weight = None, max_features = None, min_samples_leaf = 50, min_samples_split = 500)

4.5 Random Forest

In random forests, we used greedy search and 4-fold cross validation algorithm to tune 1 parameter: *min_samples_leaf* = [25, 50, 75, 100, 300, 400, 500]. We will not limit *max_depth* and we will set *min_sample_split* = [1] since we want to over-fit the individual trees. As for *n_estimators*, we will set it to be 100 to decrease the variance of our model. The model with the best setting is as follows:
RandomForestClassifier(criterion = 'entropy',

max_depth = None, min_samples_leaf = 50, min_samples_split = 1, n_estimators = 100)

4.6 KNN

We explored deeper into KNN by carrying out greedy search for the best parameter in the model. We set the cross-validation folds as default number of 3 and implemented the GridSearchCV. We tuned two parameters: n_neighbors = [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30], weights = ['distance', 'uniform']. The best KNN has parameter setting:
KNeighborsClassifier(n_neighbors = 26, weights = 'uniform').

4.7 XGBoost

We tuned 7 parameters for XGBoost with a greedy search. Since XGBoost requires the longest time training compared to other methods, we need to pick our parameters' values carefully by pre-assessing each setting. By pre-assessing, we mean we picked two distinct values with relatively large discrepancy for each parameter and then evaluate them without cross-validation. Then, we picked those promising values that might achieve a better model performance and use grid search to find the optimal one. Hence, parameters we tuned are: max_depth = [3, 7, 10], learning_rate = [0.01,0.1], n_estimators = [100, 200, 300, 400], gamma = [0, 0.2, 0.4], min_child_weight = [1, 3, 5], subsample = [0.6, 0.8, 1], colsample_bytree = [0.8, 1]. The best XGBoost has parameter setting:
XGBClassifier(colsample_bytree = 1, gamma = 0, learning_rate = 0.1, max_depth = 10, min_child_weight = 1, n_estimators = 300, objective = 'multi:softmax', subsample = 1)

4.8 Model Comparison

As shown below, the XGBoost has the best result with the highest F1 Score, which is 0.418.

Table 1: Crime Type Prediction F1 Score Table

Model	F1 Score
Logistic Regression	0.226
K-Nearest Neighbors	0.361
Random Forests	0.391
Decision Tree	0.397
XGBoost	0.418

We can identify a large gap between Logistic Regression and all other models. The explanation comes from two aspects. First, logistic regression is relatively robust, so it has low variance but high bias. The L1 regularization also restrains the model fitting. Second, there exist complete boundaries among different classes. Different classes are aggregated closely in our high dimension data and there are non-linear boundaries to separate them. Thus, our baseline model performs poorly. Also, one interesting result we find is that decision tree outperforms random forest. As mentioned before, we have a large data set and dominating classes have a unique cluster of features values. This makes our data relatively stable. Hence, random forest won't improve our result by taking the majority vote of trees. Not surprisingly, XGBoost yields the highest F1 Score, so we'll implement it into our test set.

5 Results & Conclusion

Applying our best XGBoost configuration, we retrained our whole training set and used the trained model on the test set, where we got **F1 Score** 0.419. From the confusion matrix (see Graph H), we can see that our classifier performs relatively well on those majority classes but fails on those minority. Also, the top 5 classes are *THEFT*, *CRIMINAL DAMAGE*, *NARCOTICS*, *OTHER OFFENSE*, *BATTERY*, where we can expect these types of crimes might happen at similar time and locations, so it's hard for our classifier to make the right call without additional information on each crime.

As what we studied from heat maps, crime frequencies are related with time and location. We may suggest the Chicago Police Department to pay more attention to the most dangerous time period and places like outdoor area at night. Also, more police deployment is needed during the holidays. By applying prediction models to predict crime types given time and location, police can be arranged accordingly. However, the prediction result is sometimes not accurate enough due to the fact that usable features in our models are few, there can be some latent factors contribute to the prediction. For the future work, we will try to figure out those latent factors, and more features can be included during the data collection step.

6 Note

You can find our code on our Github Account

References

Kaggle.com. 2018. *Crimes in Chicago* https://www.kaggle.com/currie32/crimes-in-chicago/home.

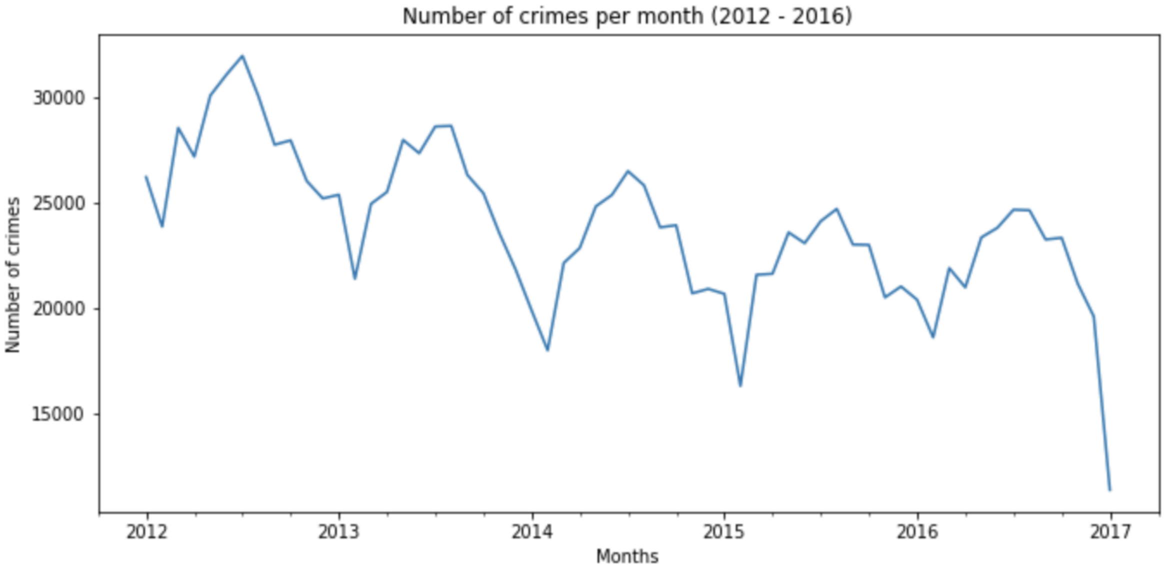
Quara.com. 2018. *What's the advantage of using the F1 score when evaluating classification performance?* https://www.quora.com/Whats-the-advantage-of-using-the-F1-score-when-evaluating-classification-performance.

statisticssolutions. 2018. *Assumptions of Logistic Regression* https://www.statisticssolutions.com/assumptions-of-logistic-regression/.

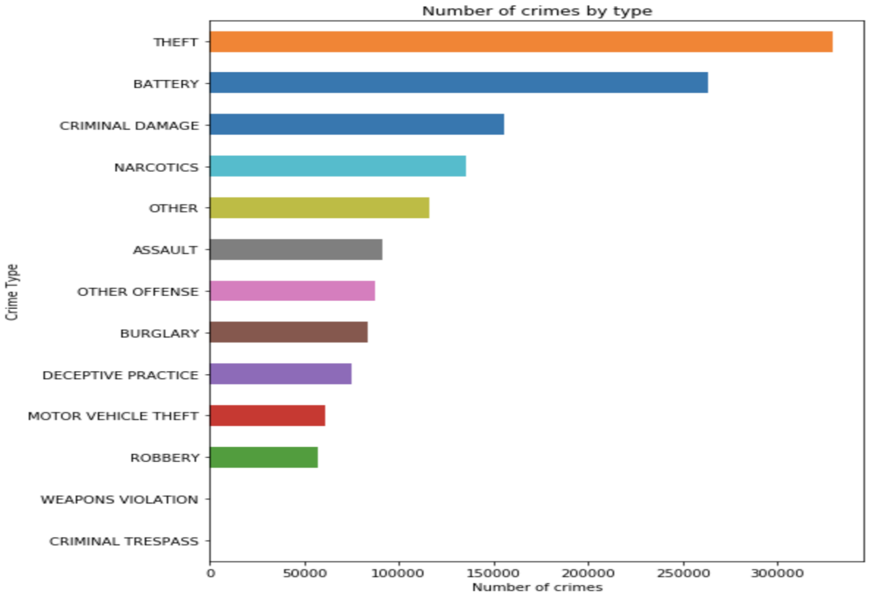
Wikipedia. 2018a. *Crime in Chicago.* https://en.wikipedia.org/wiki/Crime_in_Chicago#Chicago_Police_crime_reporting_accuracy.

Wikipedia. 2018b. *Decision Tree Learning.* https://en.wikipedia.org/wiki/Decision_tree_learning.

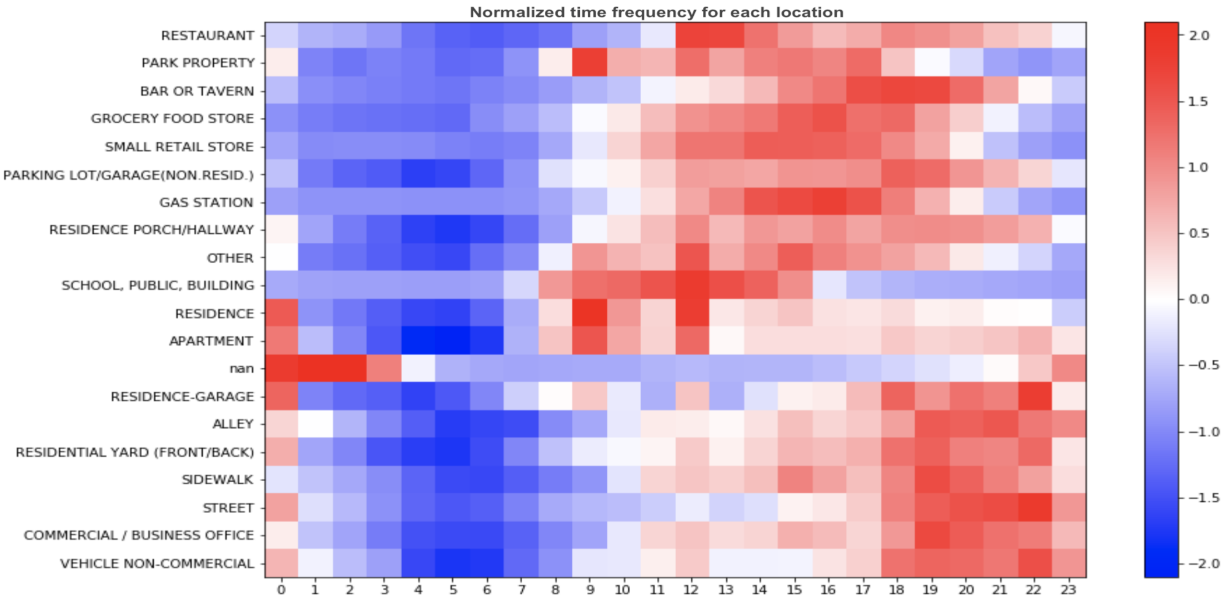
A


Number of crimes per month (2012 - 2016)
Number of crimes
30000
25000
20000
15000
2012
2013
2014
2015
2016
2017
Months

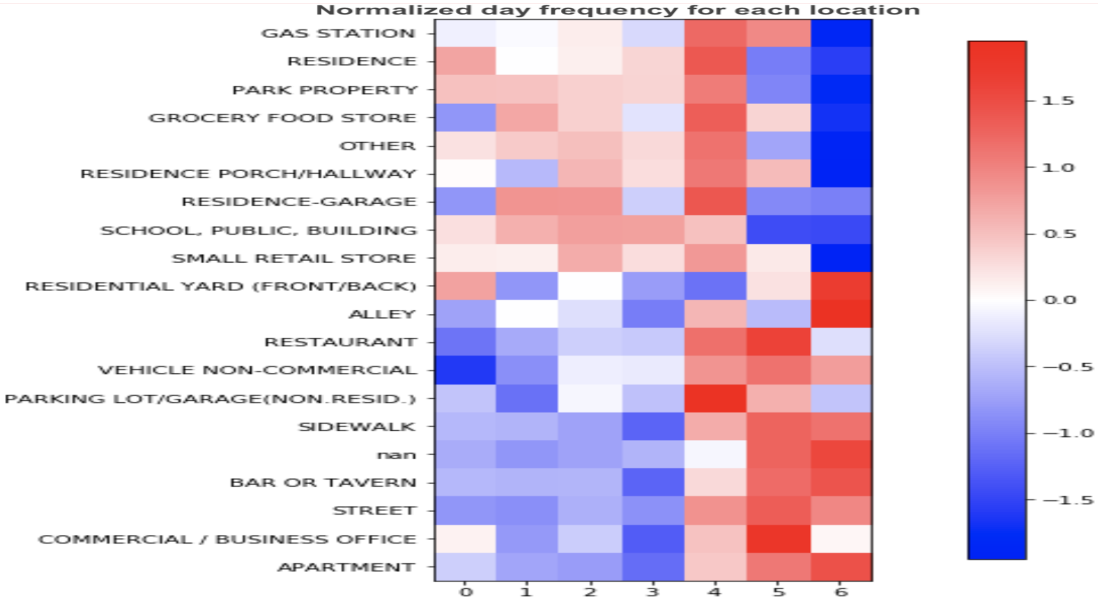
B


Number of crimes by type
Crime Type
THEFT
BATTERY
CRIMINAL DAMAGE
NARCOTICS
OTHER
ASSAULT
OTHER OFFENSE
BURGLARY
DECEPTIVE PRACTICE
MOTOR VEHICLE THEFT
ROBBERY
WEAPONS VIOLATION
CRIMINAL TRESPASS
0
50000
100000
150000
200000
250000
300000
Number of crimes

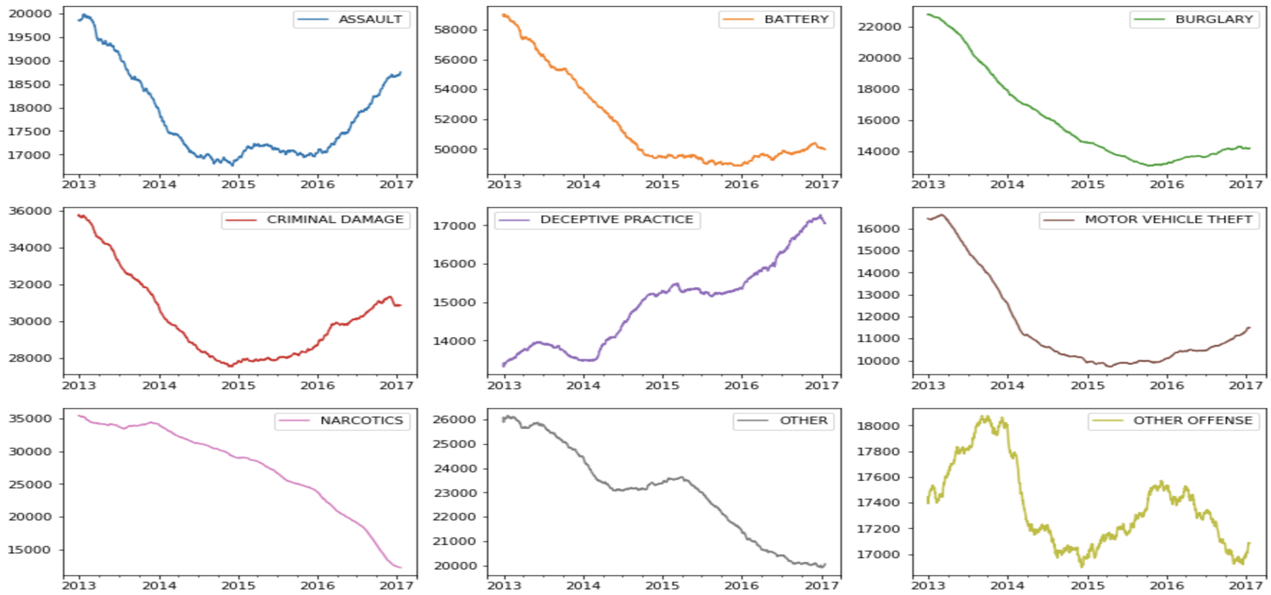
C


Normalized time frequency for each location
RESTAURANT
PARK PROPERTY
BAR OR TAVERN
GROCERY FOOD STORE
SMALL RETAIL STORE
PARKING LOT/GARAGE(NON.RESID.)
GAS STATION
RESIDENCE PORCH/HALLWAY
OTHER
SCHOOL, PUBLIC, BUILDING
RESIDENCE
APARTMENT
nan
RESIDENCE-GARAGE
ALLEY
RESIDENTIAL YARD (FRONT/BACK)
SIDEWALK
STREET
COMMERCIAL / BUSINESS OFFICE
VEHICLE NON-COMMERCIAL
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0

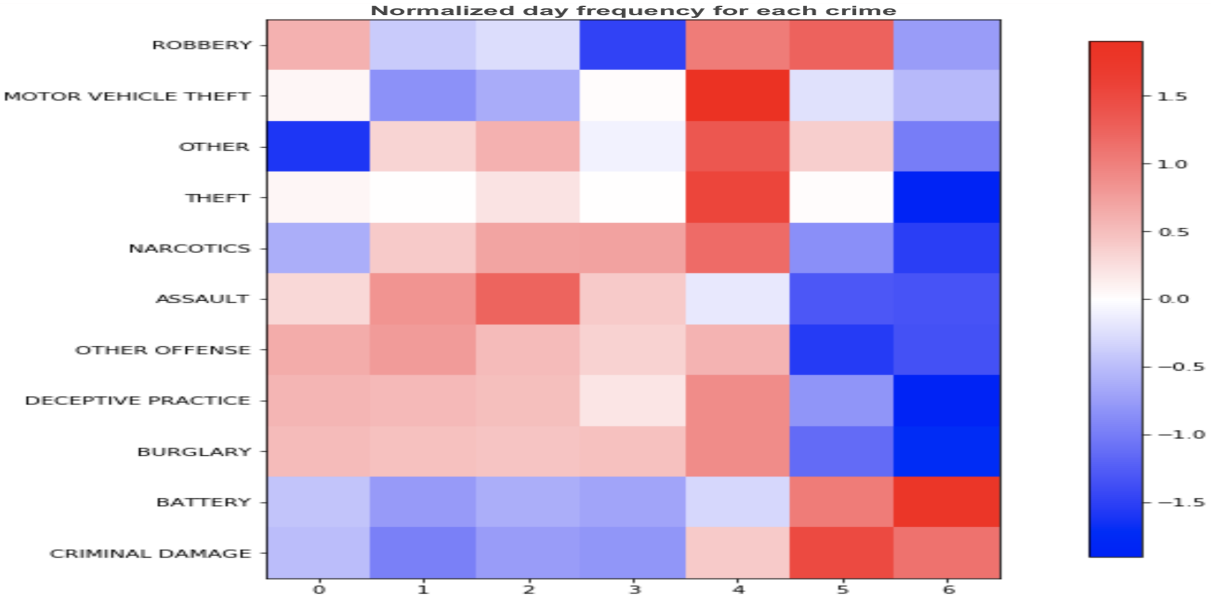
D


Normalized day frequency for each location
GAS STATION
RESIDENCE
PARK PROPERTY
GROCERY FOOD STORE
OTHER
RESIDENCE PORCH/HALLWAY
RESIDENCE-GARAGE
SCHOOL, PUBLIC, BUILDING
SMALL RETAIL STORE
RESIDENTIAL YARD (FRONT/BACK)
ALLEY
RESTAURANT
VEHICLE NON-COMMERCIAL
PARKING LOT/GARAGE(NON.RESID.)
SIDEWALK
nan
BAR OR TAVERN
STREET
COMMERCIAL / BUSINESS OFFICE
APARTMENT
0
1
2
3
4
5
6
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5

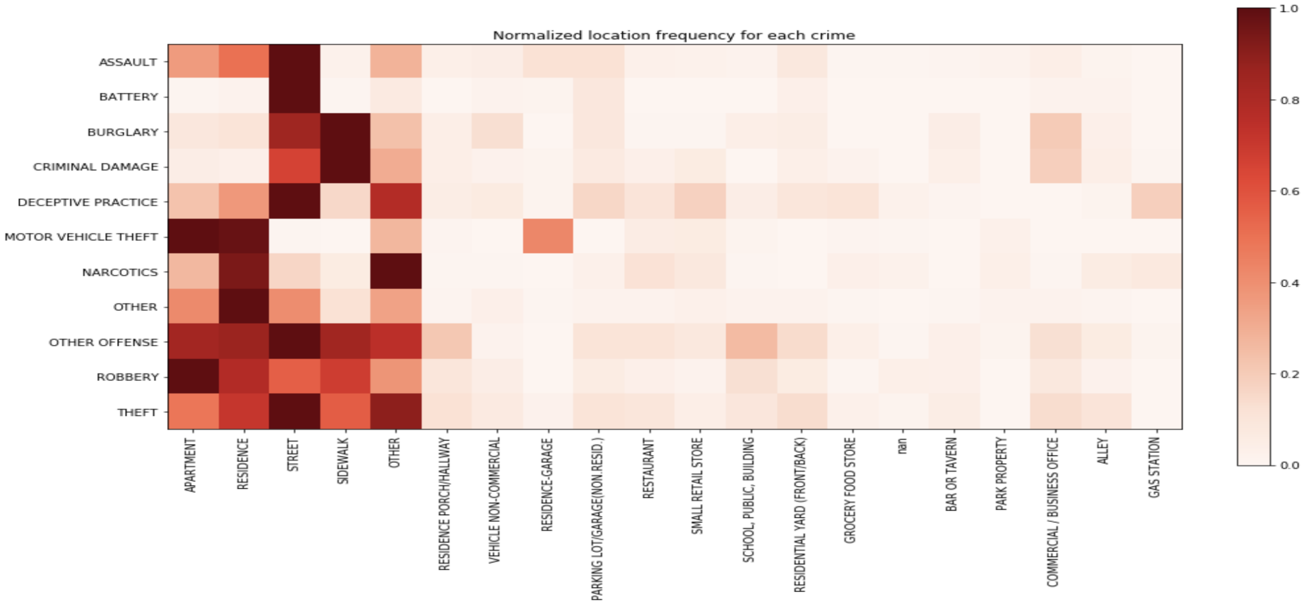
E


ASSAULT
20000
19500
19000
18500
18000
17500
17000
BATTERY
58000
56000
54000
52000
50000
BURGLARY
22000
20000
18000
16000
14000
CRIMINAL DAMAGE
36000
34000
32000
30000
28000
DECEPTIVE PRACTICE
17000
16000
15000
14000
MOTOR VEHICLE THEFT
16000
15000
14000
13000
12000
11000
10000
NARCOTICS
35000
30000
25000
20000
15000
OTHER
26000
25000
24000
23000
22000
21000
20000
OTHER OFFENSE
18000
17800
17600
17400
17200
17000
2013
2014
2015
2016
2017

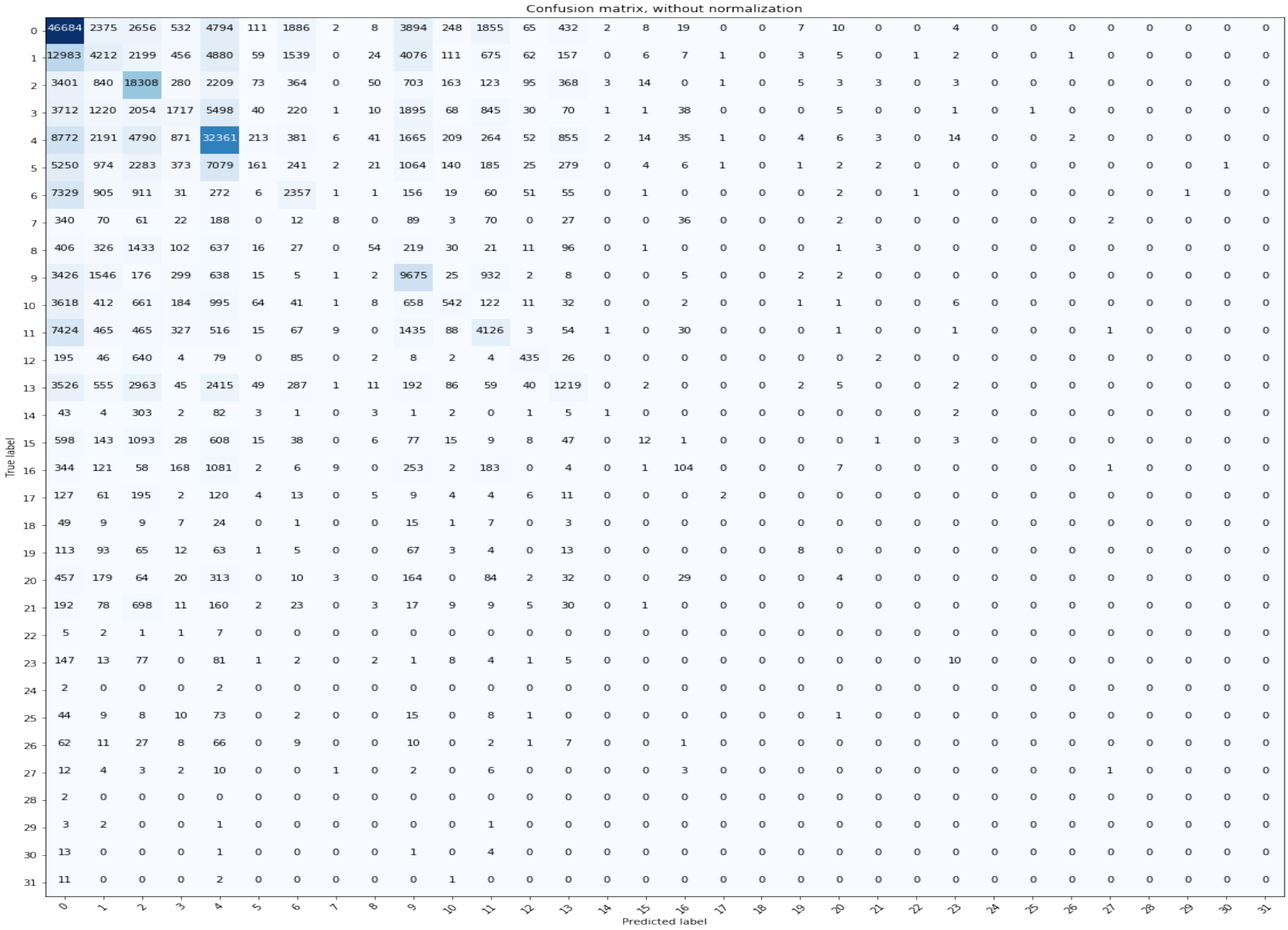
F


Normalized day frequency for each crime
ROBBERY
MOTOR VEHICLE THEFT
OTHER
THEFT
NARCOTICS
ASSAULT
OTHER OFFENSE
DECEPTIVE PRACTICE
BURGLARY
BATTERY
CRIMINAL DAMAGE
0
1
2
3
4
5
6
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5

G


Normalized location frequency for each crime
ASSAULT
BATTERY
BURGLARY
CRIMINAL DAMAGE
DECEPTIVE PRACTICE
MOTOR VEHICLE THEFT
NARCOTICS
OTHER
OTHER OFFENSE
ROBBERY
THEFT
APARTMENT
RESIDENCE
STREET
SIDEWALK
OTHER
RESIDENCE PORCH/HALLWAY
VEHICLE NON-COMMERCIAL
RESIDENCE-GARAGE
PARKING LOT/GARAGE(NON.RESID.)
RESTAURANT
SMALL RETAIL STORE
SCHOOL, PUBLIC, BUILDING
RESIDENTIAL YARD (FRONT/BACK)
GROCERY FOOD STORE
nan
BAR OR TAVERN
PARK PROPERTY
COMMERCIAL / BUSINESS OFFICE
ALLEY
GAS STATION
1.0
0.8
0.6
0.4
0.2
0.0

H


Confusion matrix, without normalization
True label
Predicted label
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0 46684 2375 2656 532 4794 111 1886 2 8 3894 248 1855 65 432 2 8 19 0 0 7 10 0 0 4 0 0 0 0 0 0 0 0
1 12983 4212 2199 456 4880 59 1539 0 24 4076 111 675 62 157 0 6 7 1 0 3 5 0 1 2 0 0 1 0 0 0 0 0
2 3401 840 18308 280 2209 73 364 0 50 703 163 123 95 368 3 14 0 1 0 5 3 3 0 3 0 0 0 0 0 0 0 0
3 3712 1220 2054 1717 5498 40 220 1 10 1895 68 845 30 70 1 1 38 0 0 0 5 0 0 1 0 1 0 0 0 0 0 0
4 8772 2191 4790 871 32361 213 381 6 41 1665 209 264 52 855 2 14 35 1 0 4 6 3 0 14 0 0 2 0 0 0 0 0
5 5250 974 2283 373 7079 161 241 2 21 1064 140 185 25 279 0 4 6 1 0 1 2 2 0 0 0 0 0 0 0 0 1 0
6 7329 905 911 31 272 6 2357 1 1 156 19 60 51 55 0 1 0 0 0 0 2 0 1 0 0 0 0 0 0 1 0 0
7 340 70 61 22 188 0 12 8 0 89 3 70 0 27 0 0 36 0 0 0 2 0 0 0 0 0 0 2 0 0 0 0
8 406 326 1433 102 637 16 27 0 54 219 30 21 11 96 0 1 0 0 0 0 1 3 0 0 0 0 0 0 0 0 0 0
9 3426 1546 176 299 638 15 5 1 2 9675 25 932 2 8 0 0 5 0 0 2 2 0 0 0 0 0 0 0 0 0 0 0
10 3618 412 661 184 995 64 41 1 8 658 542 122 11 32 0 0 2 0 0 1 1 0 0 6 0 0 0 0 0 0 0 0
11 7424 465 465 327 516 15 67 9 0 1435 88 4126 3 54 1 0 30 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0
12 195 46 640 4 79 0 85 0 2 8 2 4 435 26 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0
13 3526 555 2963 45 2415 49 287 1 11 192 86 59 40 1219 0 2 0 0 0 2 5 0 0 2 0 0 0 0 0 0 0 0
14 43 4 303 2 82 3 1 0 3 1 2 0 1 5 1 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0
15 598 143 1093 28 608 15 38 0 6 77 15 9 8 47 0 12 1 0 0 0 0 1 0 3 0 0 0 0 0 0 0 0
16 344 121 58 168 1081 2 6 9 0 253 2 183 0 4 0 1 104 0 0 0 7 0 0 0 0 0 0 1 0 0 0 0
17 127 61 195 2 120 4 13 0 5 9 4 4 6 11 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0
18 49 9 9 7 24 0 1 0 0 15 1 7 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
19 113 93 65 12 63 1 5 0 0 67 3 4 0 13 0 0 0 0 0 8 0 0 0 0 0 0 0 0 0 0 0 0
20 457 179 64 20 313 0 10 3 0 164 0 84 2 32 0 0 29 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0
21 192 78 698 11 160 2 23 0 3 17 9 9 5 30 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
22 5 2 1 1 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
23 147 13 77 0 81 1 2 0 2 1 8 4 1 5 0 0 0 0 0 0 0 0 0 10 0 0 0 0 0 0 0 0
24 2 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
25 44 9 8 10 73 0 2 0 0 15 0 8 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
26 62 11 27 8 66 0 9 0 0 10 0 2 1 7 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
27 12 4 3 2 10 0 0 1 0 2 0 6 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
28 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
29 3 2 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
30 13 0 0 0 1 0 0 0 0 1 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
31 11 0 0 0 2 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0