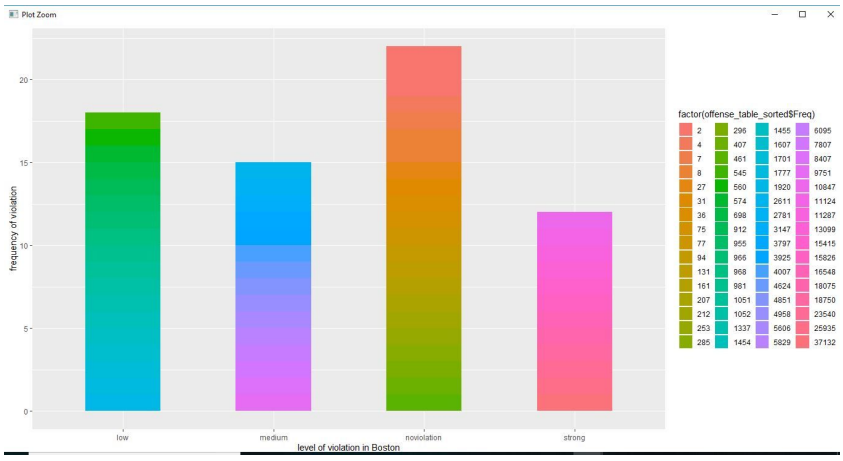
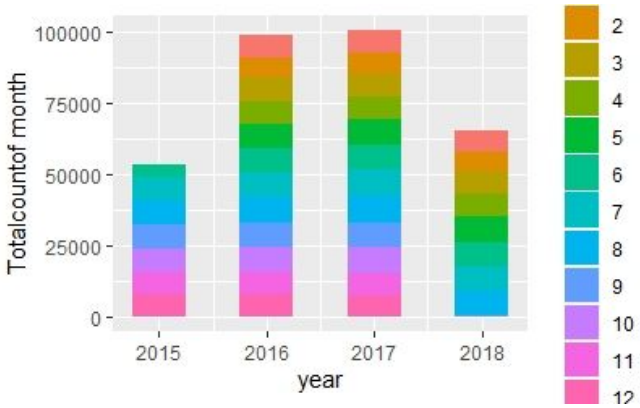
Progress Report

1. **Initial modeling and analysis efforts:**

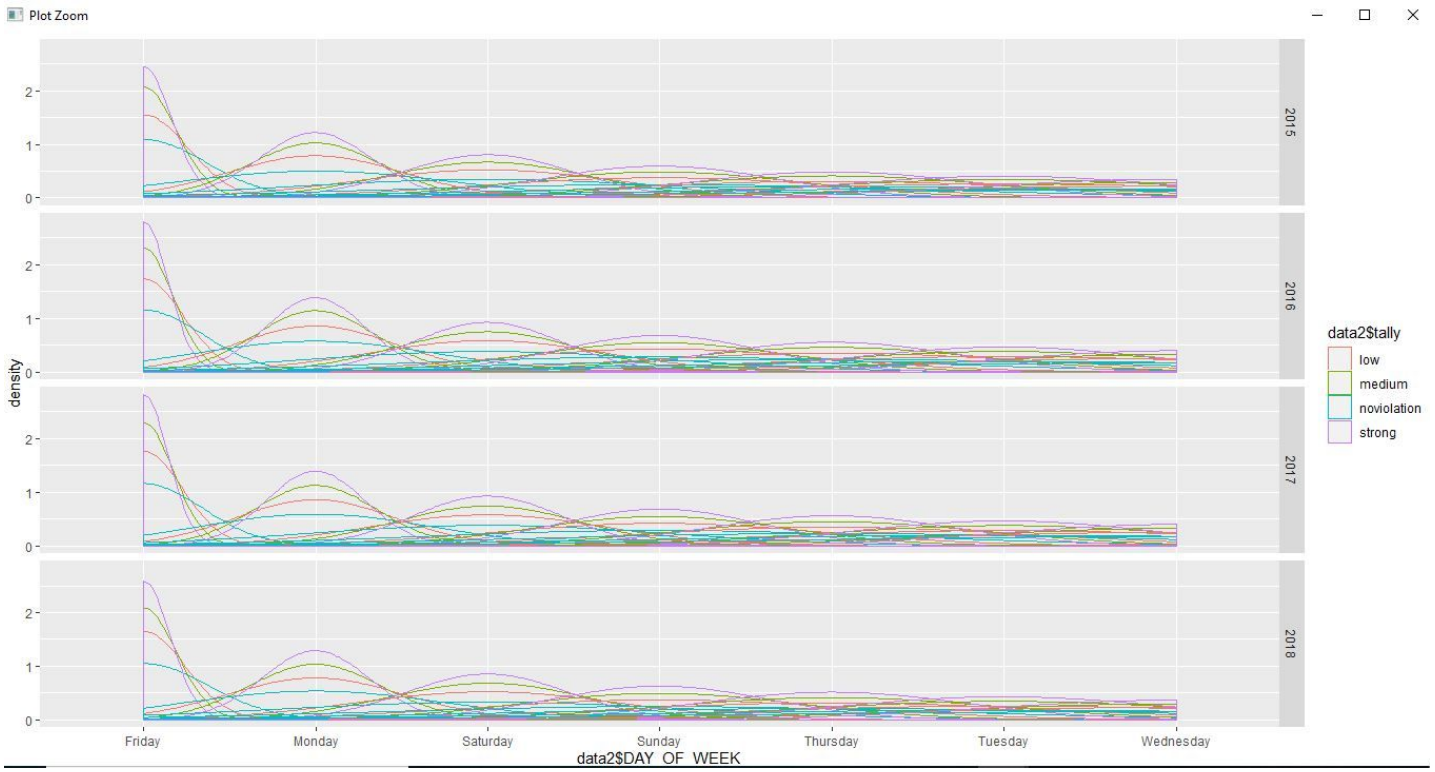
We wanted to see the impact on the level of violation. The ucr_part column says about the four types of category of the crime but didn't tell about the intensity of each crime. So we mutate a column here which tells actually the levels of violation based on the frequency of the crime.



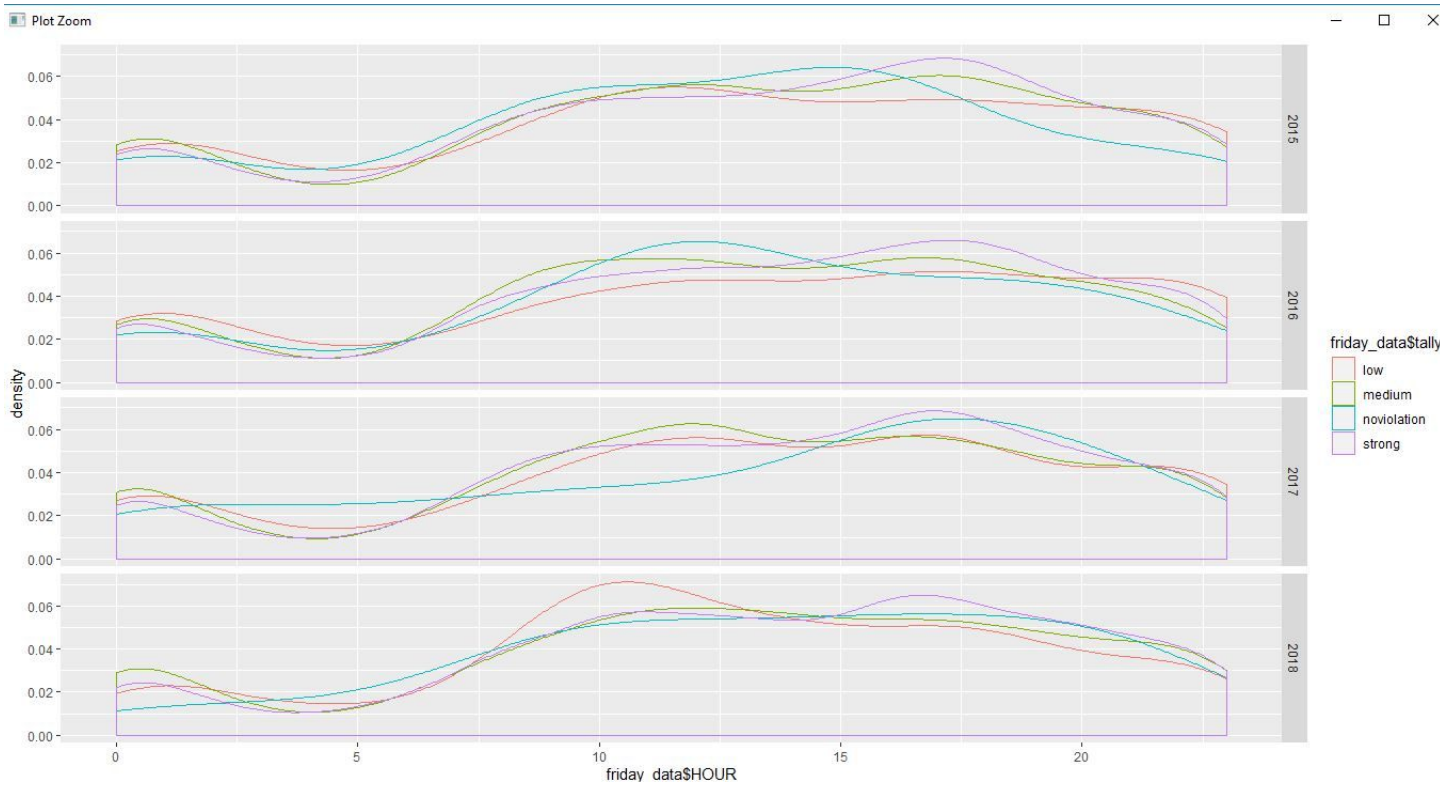
This is a barplot plotted against the frequency of violation VS level of violation. We introduced a column named tally consisting of the level of violence occurred. Those crimes have occurred in high frequency they are mentioned as strong, the lowest frequent occurred crimes are mentioned as non-violation. The color bar in this ggplot shows how the impact of the crime has been changed for a particular category of crime.



This bar plot shows the average distribution of crime over 4 years with respect to their months. It is quite surprising that in 2018 there is no crime happened in the month of December.



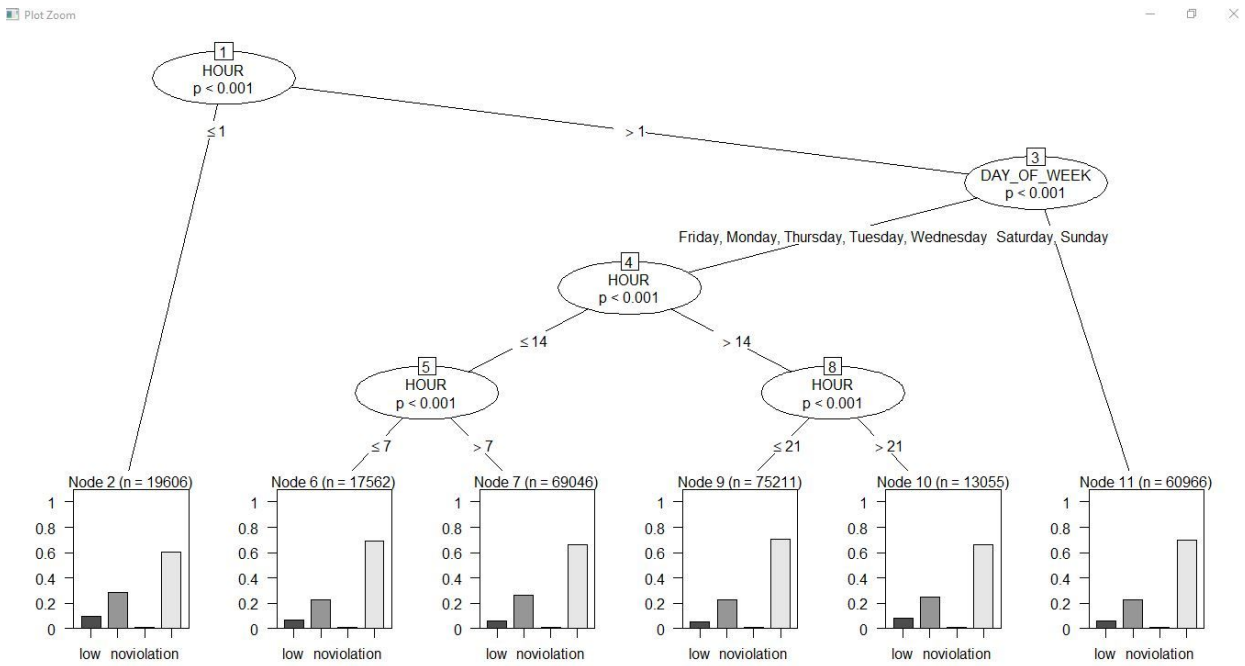
This above graph explained the intensity of crime over the day of the week. Its quite obvious trend observed from the graph that all strong violations started at a high peak from the Friday but surprisingly didn't get continued on Saturday and Sunday at a stretch when the next intensity is observed on Monday which is on weekdays. The x-axis in the graph sorted based on the sequence of intensity comes after one another. Another interesting pattern of the graph that at the starting point the isolation from between the different types of crime is very high and as the time increased there separation bandwidth decreases and tend to converge near zero. This scenario describes that Friday has 4 separate time zones which are known for 4 types of crime occurrence very strongly but that time zone converges to one on Wednesday.



So, as the Friday as the most occurrence of crime data in the last four years, we have separated the Friday data from the data-frame. Now we wanted to see in which time of Friday is the crucial hour for the occurrence of time. That's why we plotted a graph between level_of_violation(tally) vs an hour. It has been observed that the highest rate of strong violation crime occurrence is observed at 17:00 pm which is late afternoon. And the lowest number of crimes occurred at 5 am in the morning.

Decision tree

Next, we wanted to predict what kind of violation could occur based on the day of the week and in hour value. So we started taking the dataset and split into train and test dataset onto 0.85 and 0.15 percent respectively. We applied first the ctree package in r with min_criterian 0.999 and split the value of 50,000. Which gives us this decision tree below. The rules of the decision tree tells us about the segregation rules for different crimes. Next, we have found out the misclassification error which is low for our confusion matrices.



The decision tree tells that clearly the crime occurrence rules of weekdays are separated from the rules of crime occurrence of weekends.

Below is the output for misclassification error:

```
> tab <- table(predict(tree),train2$tally)
> print(tab)

             low medium noviolation strong
  low          0      0           0      0
  medium       0      0           0      0
  noviolation  0      0           0      0
  strong    6055  40755        2349 222029
> sum(diag(tab))/sum(tab)
[1] 0.8187272
> 1-sum(diag(tab))/sum(tab)
[1] 0.1812728
```

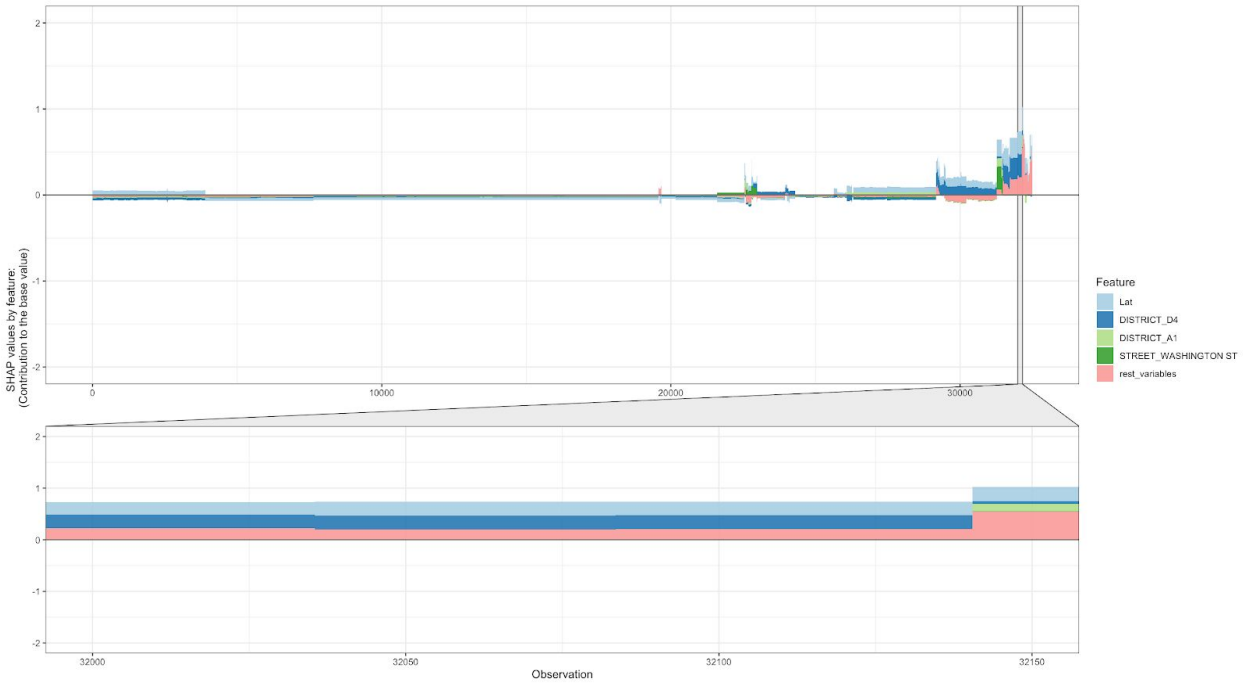
This was only our first approach for building a prediction model.

XGBoost Results

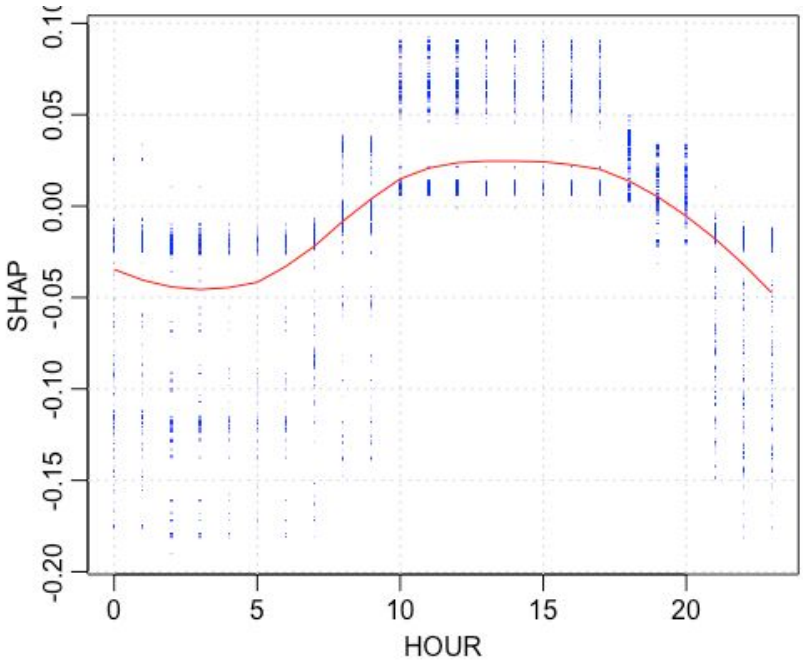
An Extreme Gradient Boosted Regression Tree model was used to predict whether a given police report was for the most frequent crime, Larceny. The model was trained on a randomly sampled subset of the data (n = 32489) and had a prediction accuracy of 91.7% when predicting occurrences of Larceny on the test data (n = 13924).

The above features above reflect their overall importance (i.e., Gain) on model predictions. Evidently, location-related variables like Latitude, specific districts and streets were most informative, with Hour being the lone temporal variable related to Larceny.

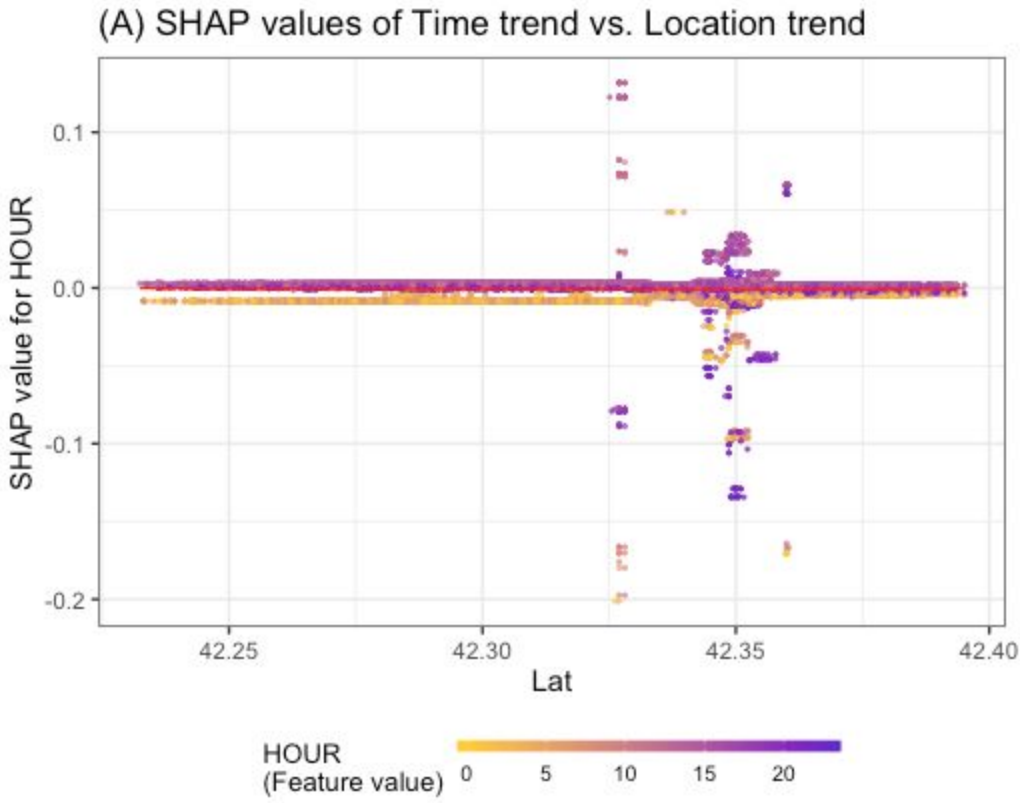
SHAP values reflect the influence that predictors have on the resultant predictions of y for individual observations of **X**. The Force Plot below represents the relative influence that each predictor had across all 32489 observations in the training set.



Despite being the most frequently occurring crime, most observations in the dataset were not pertaining to Larceny, this is evident in the above plot, which is largely flat across non-occurrences. SHAP plots are also useful in understanding the individual relationships that variables have with the outcome. For example, Hour's influence appears negligible in the above plot, but an individual plot of Hour of the day and its SHAP value reveals the times in which Larceny is more common.



It appears that Hour's influence on predictions was positive during the day, while its influence on predictions were negative at night. Simply put, Larceny is more likely to occur mid-day rather than the early morning and evening.



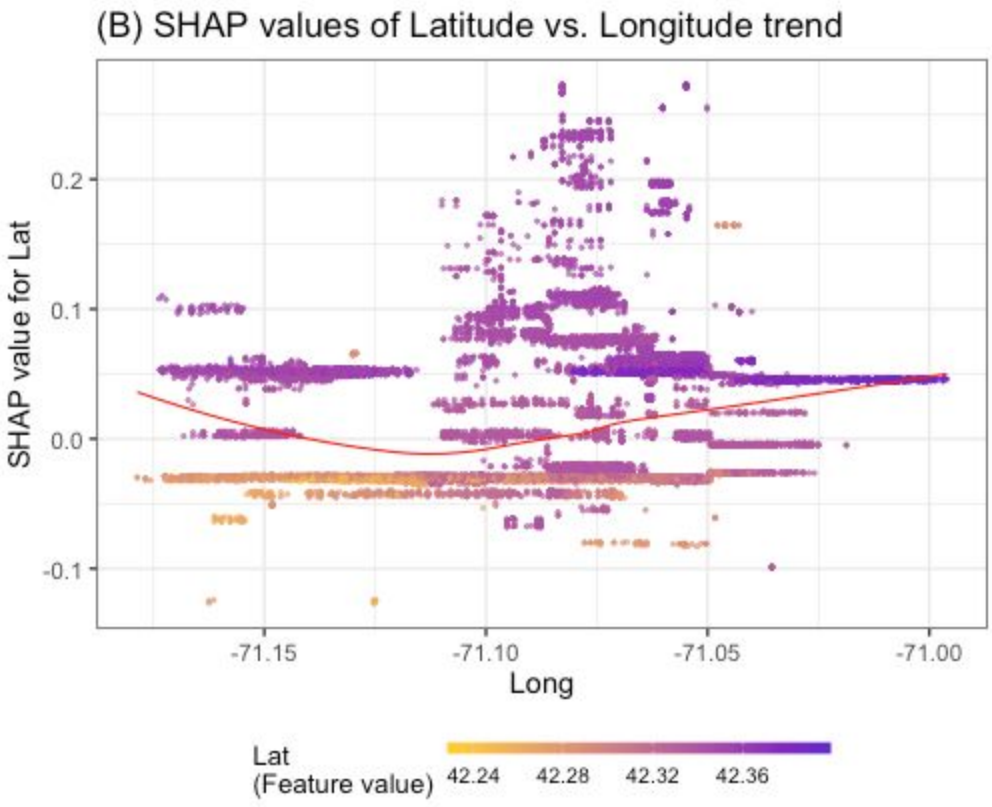
In regard to Latitude, it appears that Hour has differential effects at different values of Lat, meaning that there may be differences in when Larceny occurs, depending on what part of town you are in. It appears that further north (i.e., closer to downtown, the financial district and the airport), crime appears to be most common during mid-day, while there is more variability further south of town.

Next Steps

Initial Results suggest that Spatial features (i.e., Latitude, Longitude districts, streets) are more related to occurrences of Larceny than most Temporal features (i.e., Day of Week, Month, Year). Thus, the next phase of exploration will be to assess other forms of crime and whether the trends regarding Larceny differ between different forms of crime.

Additionally, we will attempt to understand how different locations in the city differ from one another using Census Data to better understand why certain locations in the city appear prone to crime. Spatial location can be reverse coded using the Census API. Because of API limitations, a subset of the data (n = 5000) will be used. Variables like population density, resident demographics (i.e., age, gender, race) as well as median income from the American Community Survey (ACS). Specifically, the 2015 ACS will be used, as it provides a more granular level of measurement (block rather than district) than the 2018

data. The sacrifice of temporal recency for spatial granularity seems fitting given the primary results and the relative effects of spatial features compared to temporal features.



Additional Plans:

We will identify additional modelling techniques as well as different evaluation metrics to for greater insights to our present results.

We are going to add in confusion matrices and attempt to compensate for issues of imbalance in our Dependent Variable to describe the performance of our classifier on the test data for which the true values are known.

We will add a learning curve plot for our final Decision Tree Models to show the differences in training error and validation set error at different tuning parameters for our models.

We will merge Census data for the Boston Metropolitan area that we found online using the Census API using a sample of the full dataset (due to API limitations).

After Midterm Progress :

We had to find out the correlation of

In this stage we tried to do the feature selection for prediction of level_of_violation which i considered as a label in this case. In the First Iteration we have chosen “Month” and “UCR_part” as predictors and “Level_of_violation” as the label. I applied DecisionTreeClassifier to find out the decision rule for predicting the level_of_violation. The Test accuracy became Fair in this case.

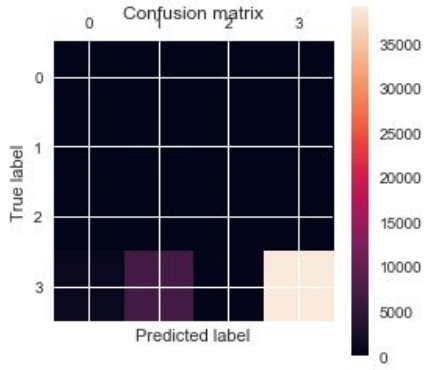
```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.8245381753987768
```

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_pred,y_test)
cm
```

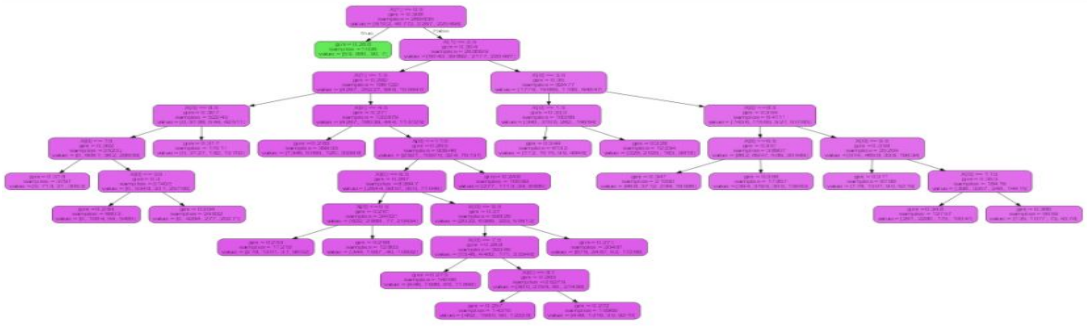
```
array([[    0,     0,     0,     0],
       [    7,   161,    14,     3],
       [    0,     0,     0,     0],
       [ 1025,  6948,   352, 39073]], dtype=int64)
```

```
plt.matshow(cm)
plt.title('Confusion matrix')
plt.colorbar()
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.show()
```



Below is the Decision Tree :

```
graph8.set_size('"9.5,9.5!"')
Image(graph8.create_png())
```

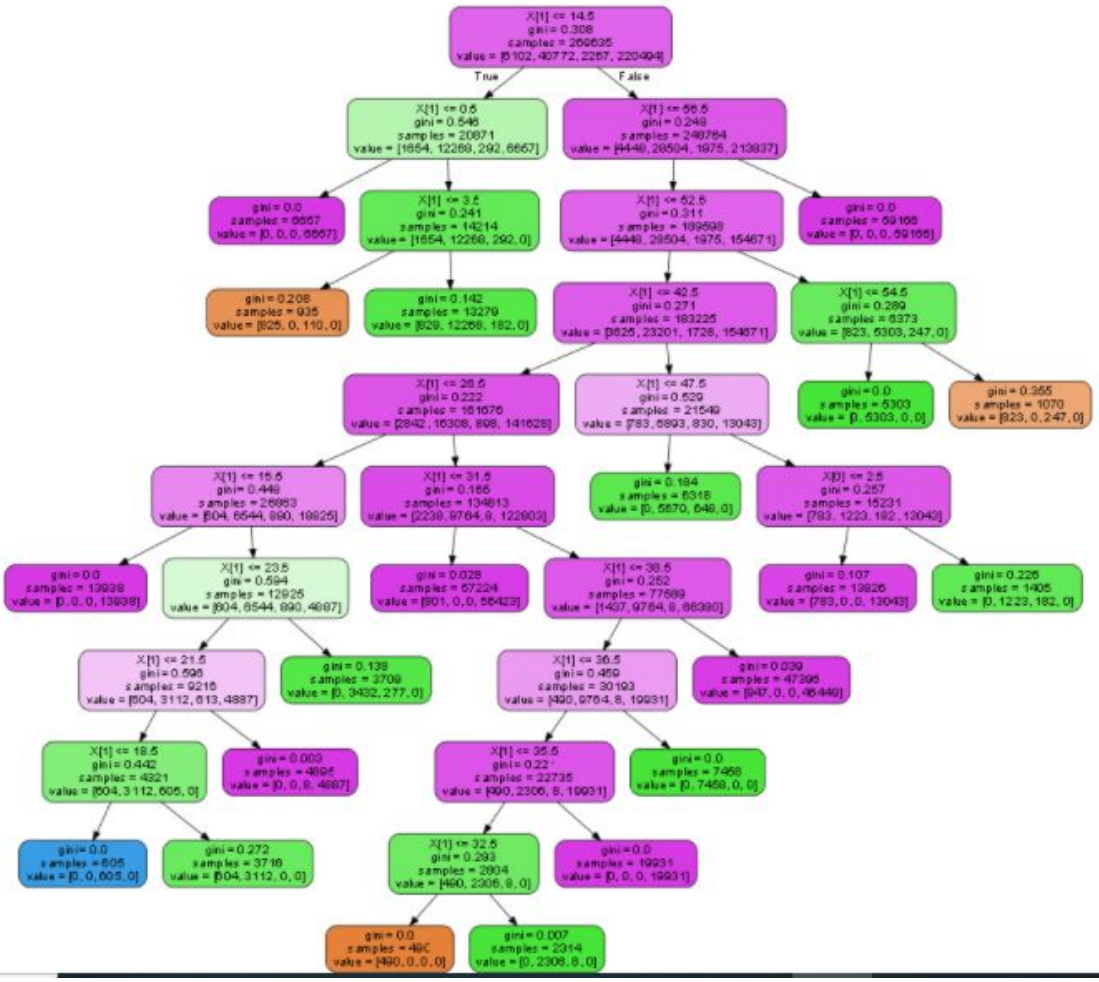


From this picture Its visible that although the accuracy came in a decent only one green colored True rule is created from this graph.

In the Next Iteration we wanted to choose better predictors means a better set predictors which will give better realistic results and more explainable. So , This time , i have chosen “Crime_type” , “Offense_code_group” an “District” as predictors and put the same previous label “Level_of_violation”. we have chosen “District” because it has also shown previously that it holds High feature importance. we had to apply encoding process to “District” and “Offense_code” s they were not in numeric types. This case The decision tree became more realistic and the accuracy became Also Higher.

```
In [786]: graph_11.set_size('"9.5,9.5!"')
          Image(graph_11.create_png())
```

Out[786]:



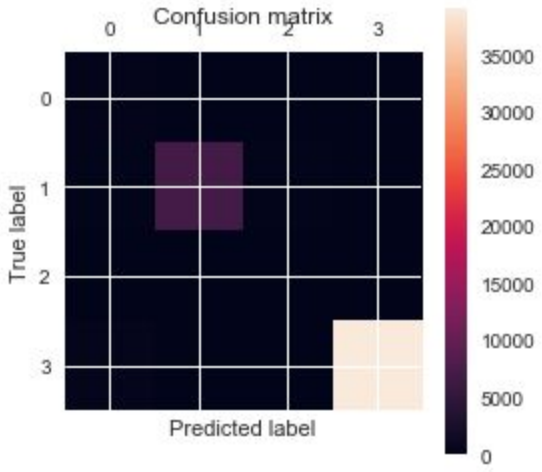
The test accuracy below shows an improvement in the process.

```
In [741]: print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

          Accuracy: 0.9802240295903999
```

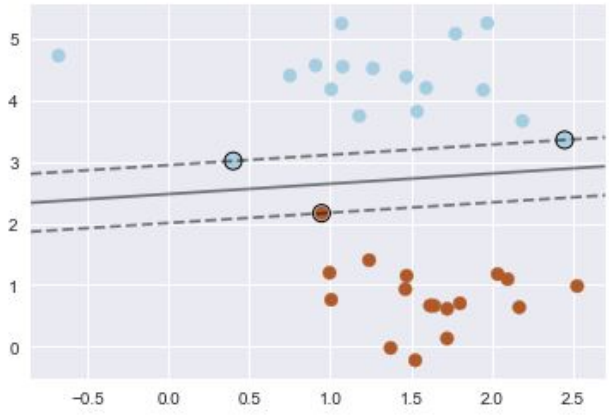
```
In [742]: from sklearn.metrics import confusion_matrix
          cm = confusion_matrix(y_pred,y_test)
          print(cm)
          plt.matshow(cm)
          plt.title('Confusion matrix')
          plt.colorbar()
          plt.ylabel('True label')
          plt.xlabel('Predicted label')
          plt.show()
```

```
[[  333     0    55     0]
 [  241  7109   187     0]
 [    0     0   124     0]
 [  458     0     0 39076]]
```



Then I tried to cluster the top predictors (like “District” etc) clustered into 0 and 1 points so that each clusters data point will belong to a class based on the impedance of crime occurrence. When i Introduced “District” the relationship between clusters held linearity. So we applied support vector machines to find the widest street approach between two clusters. The cros-svalidation accuracy became closest to 1.Apparently It looks like overfitting but it happened because the test data was the part of the training data which is our only limitation in this case.

```
<matplotlib.collections.PathCollection at 0x1f23f39ec88>
```

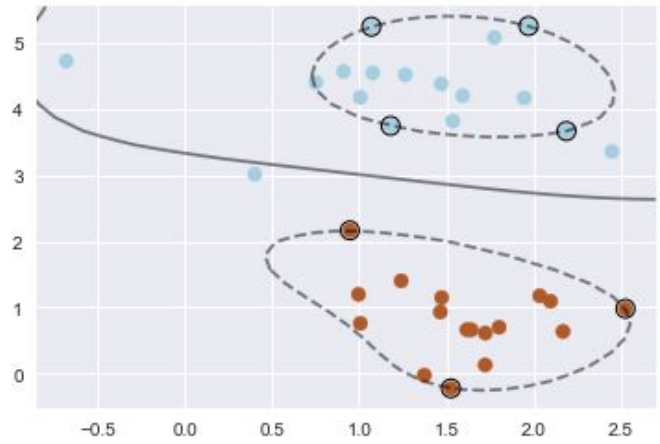



```
model.support_vectors_
```

```
array([[2.45161058, 3.35844964],
       [0.39920934, 3.01626962],
       [0.9461919 , 2.16566767]])
```

But When I introduced the “Offense_code_group” then the linearity property became lost. So, we need to add kernel function “rbf” to make the decision boundary.

```
<matplotlib.collections.PathCollection at 0x1f23f9d3710>
```



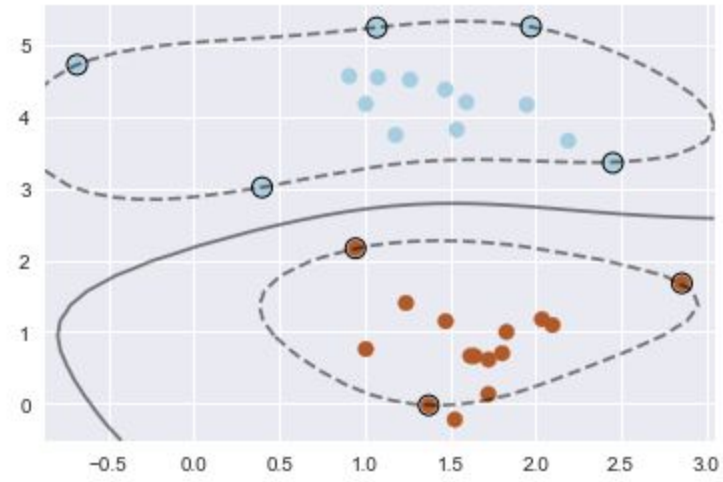

```
model.support_vectors_
```

```
array([[ 2.19018277,  3.66855671],
       [ 1.17976408,  3.7486251 ],
       [ 1.0698984 ,  5.24906511],
       [ 1.97257657,  5.25887053],
       [ 1.52672244, -0.22442003],
       [ 0.9461919 ,  2.16566767],
       [ 2.52917639,  0.98150255]])
```

The Amount of Non-linearity changed if we trained the model with more Data.

Below is the picture of changed decision boundary for changing test train split from 0.3 to 0.2

```
Out[233]: <matplotlib.collections.PathCollection at 0x2625c69c6a0>
```




```
In [234]: model.support_vectors_

Out[234]: array([[ 2.45161058,  3.35844964],
                 [ 0.39920934,  3.01626962],
                 [-0.6831733 ,  4.72863941],
                 [ 1.0698984 ,  5.24906511],
                 [ 1.97257657,  5.25887053],
                 [ 0.9461919 ,  2.16566767],
                 [ 1.37370809, -0.025348  ],
                 [ 2.85495646,  1.67921056]])
```

Below is the confusion matrix