

Phase 3 Intelligence Report—UMD Alert
Reporter: Agent Poltergeist (David Kuan-Yu Chen)
Collaborated Agent: None
Dataset: <http://alert.umd.edu/alerts> (UMD Alerts 2008-2016)

Main Threat

People are more likely to be victims of crimes in two areas, namely area of east College Park metro station circled by Baltimore Avenue, Hartwick Road, Columbia Avenue, and College Avenue and area circled by Baltimore Avenue, Berwyn Road, Potomac Avenue, and Lakeland Road, at around 6:00 PM to 6:00 AM in College Park, MD, according to my analysis.

Evidence

1. Fig 1 shows all crimes committed between 2008 and 2016, in which we can find that the northeast and the southeast sides of the school are two areas that had the densest colors, indicating that most crimes happened in these two areas.

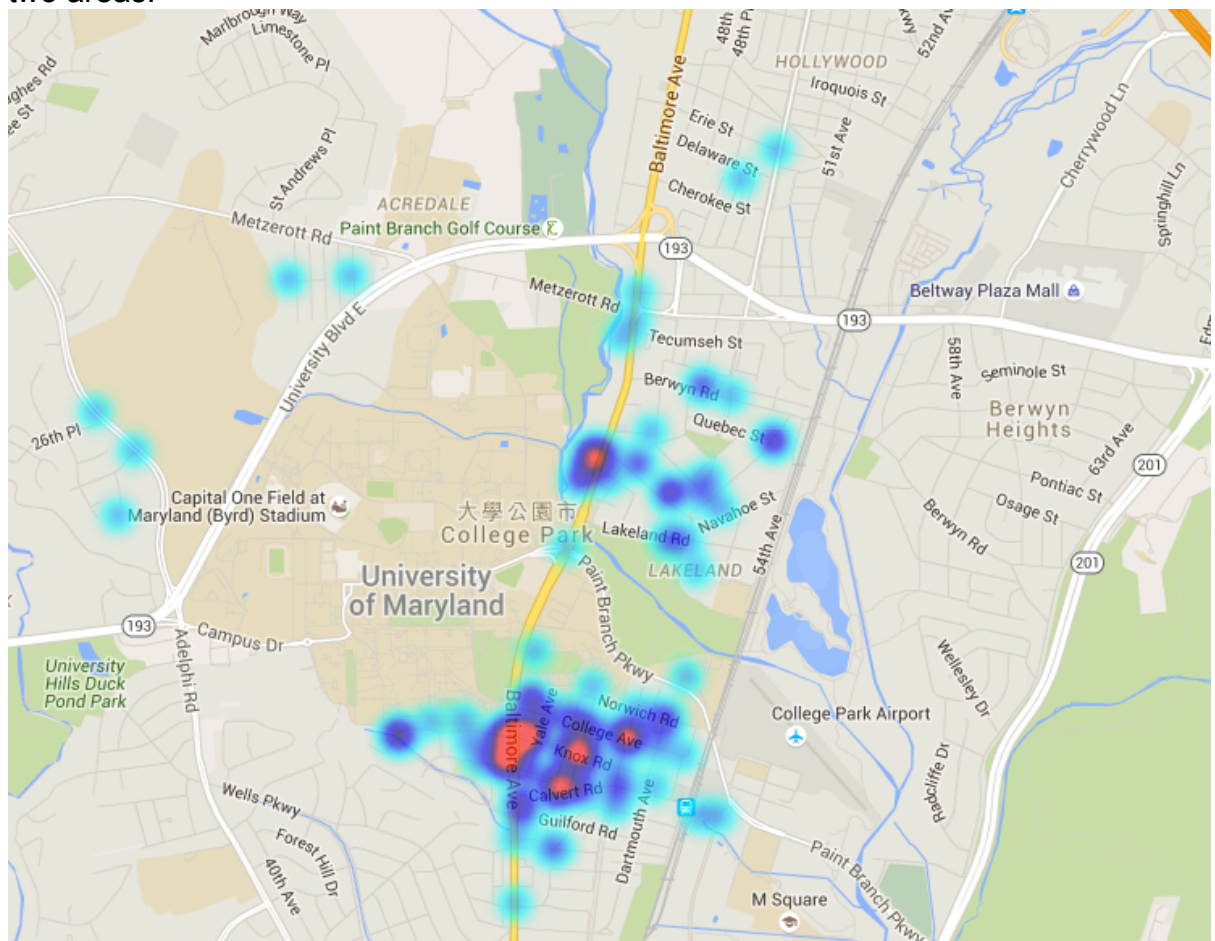


Fig 1. Crime Heat Map (2008-2016)

2. Fig 2 to Fig 10 indicate that criminals are less active between 6:00 AM to 6:00 PM, but after 6:00 PM more and more crimes were reported and 0-3 AM seems to be the peak time. Around 86.7% of the reported crimes on UMD Alert happened between 6:00 PM to 6:00AM.

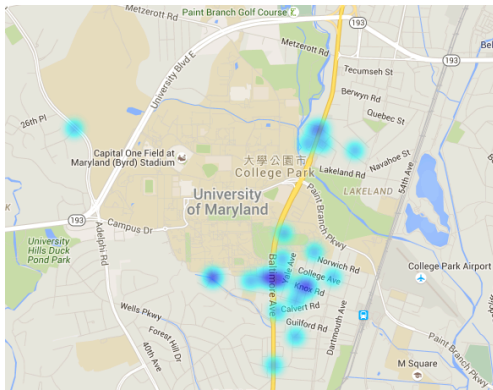


Fig 2. Crime Heat Map 0-3 AM

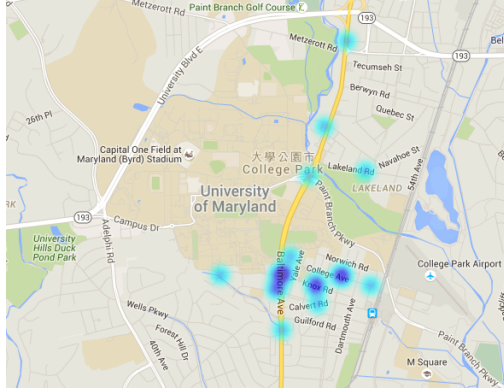


Fig 3. Crime Heat Map 3-6 AM

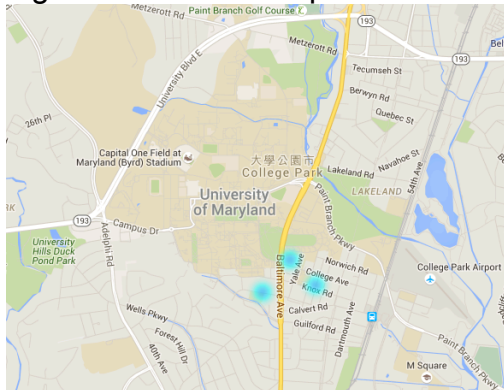


Fig 4. Crime Heat Map 6-9 AM

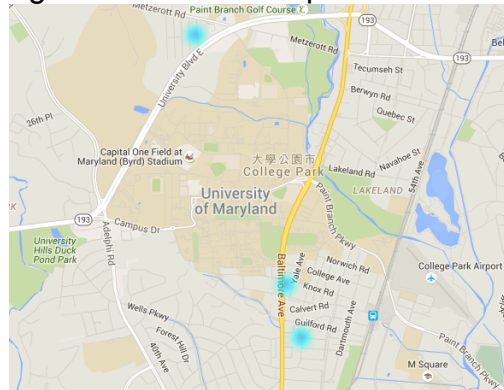


Fig 5. Crime Heat Map 9 AM-12 PM

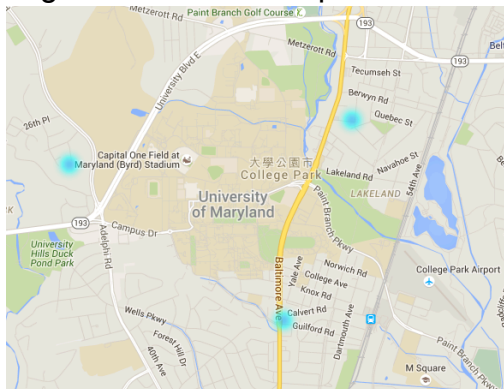


Fig 6. Crime Heat Map 12-3 PM

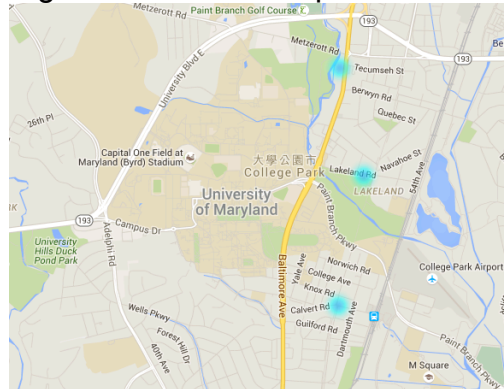


Fig 7. Crime Heat Map 3-6 PM

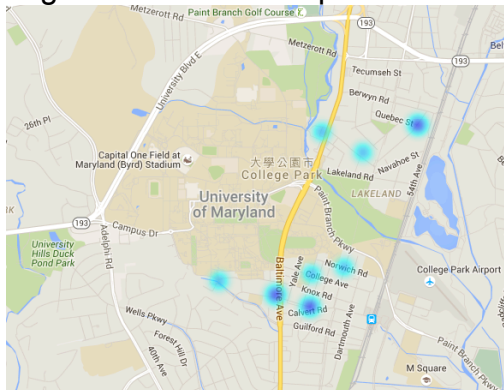


Fig 8. Crime Heat Map 6-9 PM

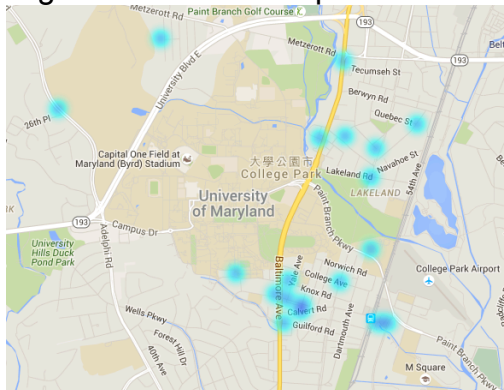


Fig 9. Crime Heat Map 9 PM-12 AM

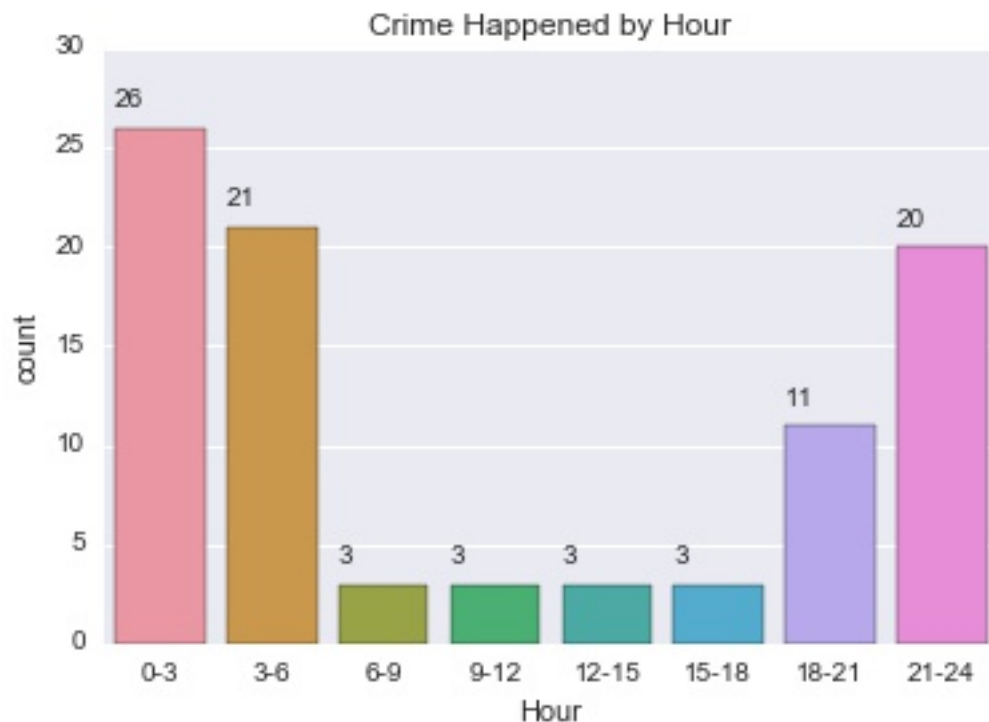


Fig 10. Crime Happened by Hours

Reliability

The data was from official record; however, the formats of records in UMD Alert are not consistent. Also, most of the locations of the crimes were not precisely disclosed; for example, a lot of the locations were mentioned as 4300 block of Baltimore Avenue or College Avenue. Therefore, the heat map can only show the approximate locations of those crimes. Furthermore, not all the records included the time the incidents happened. However, given these constraints, the crime heat map still provides a good overview of the crimes committed in College Park area and the time they happened.

Method

First, I wrote a Python script (Fig 11) to do the following tasks:

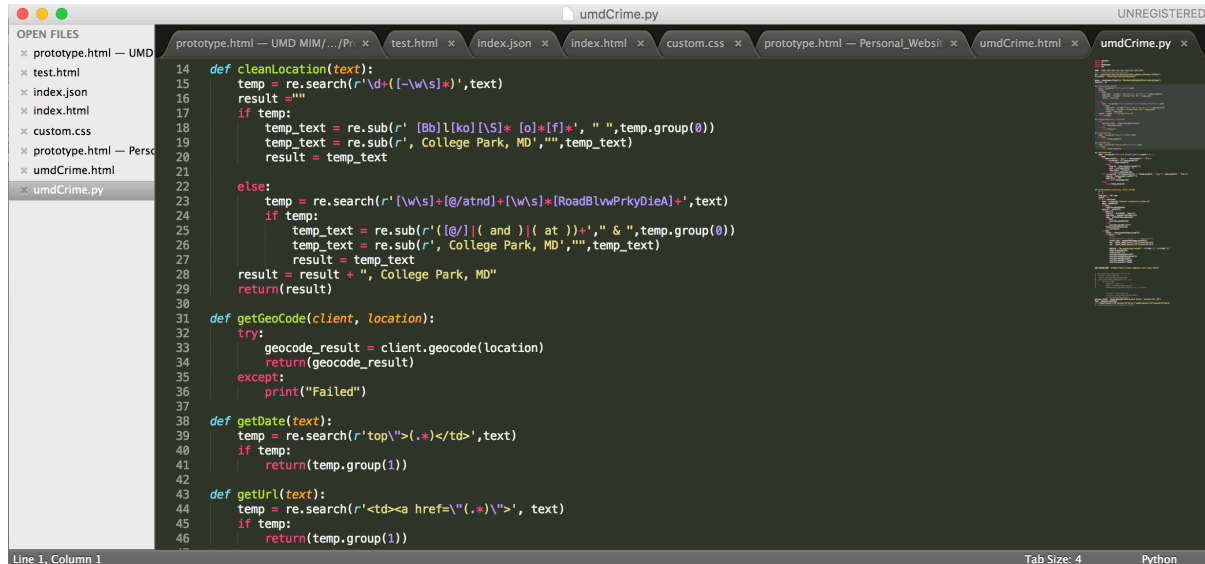
1. Scrape crime records from UMD Alert website
2. Extract the address from each record
3. Clean the address information, such as 4300 Block of Baltimore Avenue to 4300 Baltimore Avenue
4. Sent the cleaned address to Google Map API to get the latitude and longitude of the location
5. Extract other information from UMD Alert, such as date, crime, time
6. Store all relevant information in a csv file
7. Store only address information in a text file

Next, I put only the address information into an html file, which includes Google Map JavaScript API to draw the heat map.

Then, I used iPython notebook (Fig 12) to run some data analysis, such as binning time of crime to "0-3", "3-6", "6-9", "9-12", "12-15", "15-18", "18-21", and "21-24", and see the how many incidents happened in each time period. Also, I exported the

result to a json file so that I can put this json into the html file to enable showing crimes in different time periods.

Though, the result is very straightforward and easy to understand, I spent a lot of time on cleaning the data due to inconsistent data format. Some addresses in the records result in null values of the latitude and longitude.



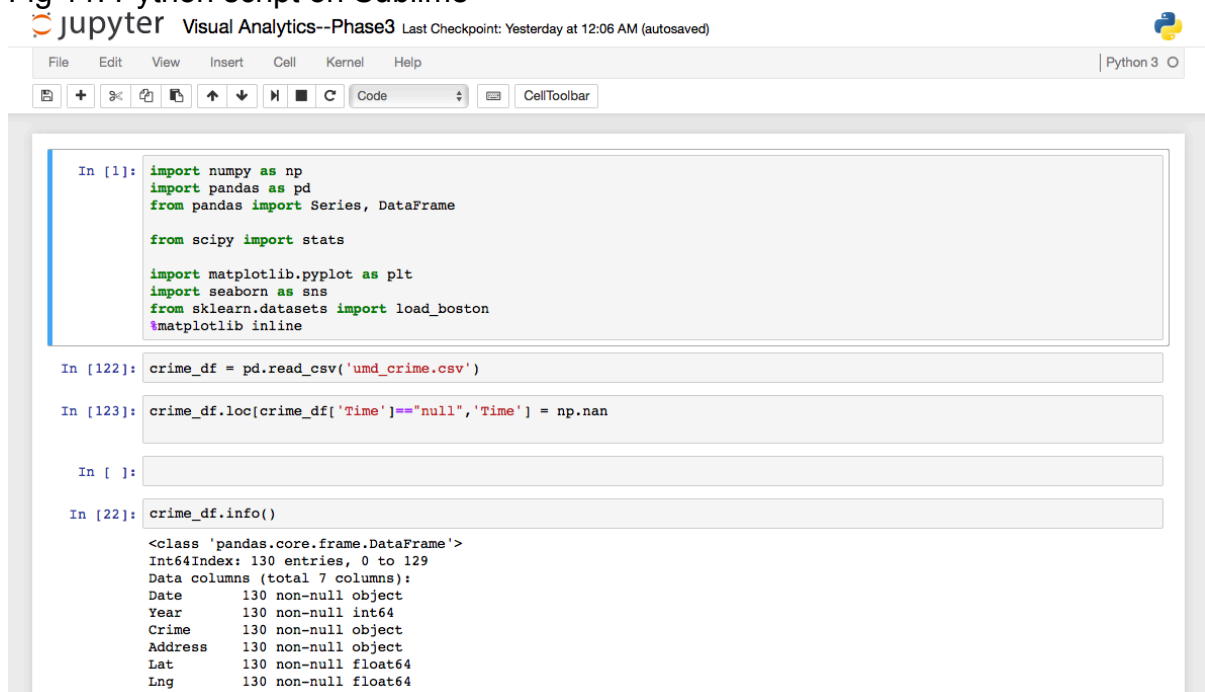
```
def cleanLocation(text):
    temp = re.search(r'd+([-\\s]*)',text)
    result = ""
    if temp:
        temp_text = re.sub(r' [Bb]l[ko][\\s]* [o]*[f]*', " ",temp.group(0))
        temp_text = re.sub(r', College Park, MD','',temp_text)
        result = temp_text
    else:
        temp = re.search(r'([\\s]*[a-zA-Z]+[\\s]*[Rr]oad[\\s]*[Pp]rky[\\s]*[Dd]ie)',text)
        if temp:
            temp_text = re.sub(r'([@/]|( and )|( at ))+', " & ",temp.group(0))
            temp_text = re.sub(r', College Park, MD','',temp_text)
            result = temp_text
        result = result + ", College Park, MD"
    return(result)

def getGeoCode(client, location):
    try:
        geocode_result = client.geocode(location)
        return(geocode_result)
    except:
        print("Failed")

def getDate(text):
    temp = re.search(r'top\\>(.*)</td>',text)
    if temp:
        return(temp.group(1))

def getUrl(text):
    temp = re.search(r'<td><a href=\\\"(.*)\\\">', text)
    if temp:
        return(temp.group(1))
```

Fig 11. Python script on Sublime



```
In [1]: import numpy as np
import pandas as pd
from pandas import Series, DataFrame

from scipy import stats

import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_boston
%matplotlib inline

In [122]: crime_df = pd.read_csv('umd_crime.csv')

In [123]: crime_df.loc[crime_df['Time']=='null','Time'] = np.nan

In [ ]:

In [22]: crime_df.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 130 entries, 0 to 129
Data columns (total 7 columns):
Date      130 non-null object
Year      130 non-null int64
Crime     130 non-null object
Address   130 non-null object
Lat       130 non-null float64
Lng       130 non-null float64
Time      90 non-null object
```

Fig 12. iPython (Jupyter) Notebook