CRIME DATASET ANALYSIS

CITY OF CHICAGO (2001-2018)

12.05.2019

Shubham Malik, Rohan Harode, Akash Singh Kunwar
Group Name: Data Pirates

Master's Program in Big Data
Simon Fraser University

Table of Contents

1. Introduction

Crimes in Chicago have been tracked by the Department's Bureau of Records since the beginning of the 20th century. The police department of the city of Chicago strives to improve its services and reduce crime in the city, this was our major motivation behind the project. We have obtained our data set from Chicago Data Portal (extracted from the Chicago Police Department) for the years 2001-2018, which is one of the richest open-source datasets. The data at 1.5GB contains over 6.6 million unique records with more than 20 features. This enormous amount of information has been scaled to our use and filtered extensively.

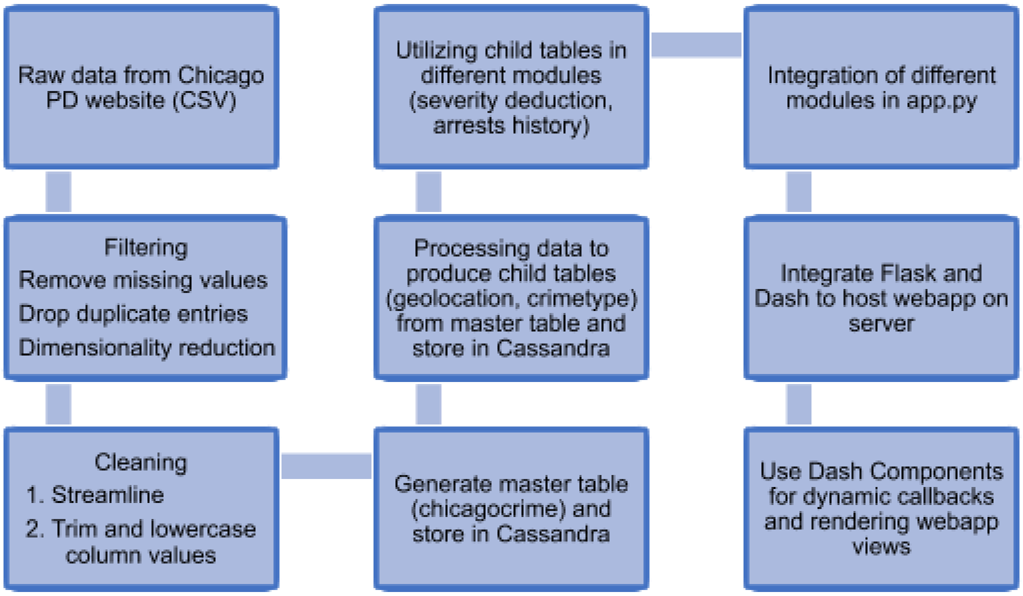
1.1. Problem Statement

By deep diving into the crime dataset, we aim to create interesting, interactive and intuitive visualizations to convey the story hidden in the data. We are particularly interested in understanding the pattern of crimes by implementing the following analytical results:

- **Deducing severity of the crime:** Classification of crime based on information from the Chicago PD website and crime description
- **Crime forecasting:** Training FB's Prophet on our data and forecasting crime for the year 2019 and examine crime rate trends over the years
- **Word-Cloud:** Generating word cloud for different crime types and locations to get a better understanding of major occurring crimes and the most affected location type
- **Geo Maps:** Understanding most frequent crime location using crime density of each crime on geo maps and interactively comparing it across the years
- **Analysis of data-set:** Analyzing the data to provide some answers to the following:
  - Crime trends over the years
  - Rate of each crime per year/month/hour
  - Locating high crime neighborhoods

The system can be implemented as an aid to supplement the Chicago PD experience and help them to prevent the occurrence of a type of crime at a particular location in the coming years.

2. Methodology


Raw data from Chicago PD website (CSV)
Filtering
Remove missing values
Drop duplicate entries
Dimensionality reduction
Cleaning
1. Streamline
2. Trim and lowercase column values
Generate master table (chicagocrime) and store in Cassandra
Processing data to produce child tables (geolocation, crimetype) from master table and store in Cassandra
Utilizing child tables in different modules (severity deduction, arrests history)
Integration of different modules in app.py
Integrate Flask and Dash to host webapp on server
Use Dash Components for dynamic callbacks and rendering webapp views

2.1. ETL and Data Operations

Cleaning is a crucial step before any analysis of the data. The extensive data obtained needed some data pre-processing such as Data Cleaning and Data Normalization. In the process, we cleaned and filtered the data to provide suitable data types. The operations like missing value checks, lower case conversion, duplicate check, and whitespace trimming, etc. are performed efficiently to make the data uniform and streamlined. Cleaned data is then filtered out to remove the features that are not relevant to our analysis (x coordinate, y coordinate, iucr).

After making sure the data is transformed efficiently in a structured manner and clean format, we then export the data to Cassandra DB. After cleaning, data size reduced from 1.5 GB to nearly 1.2 GB.

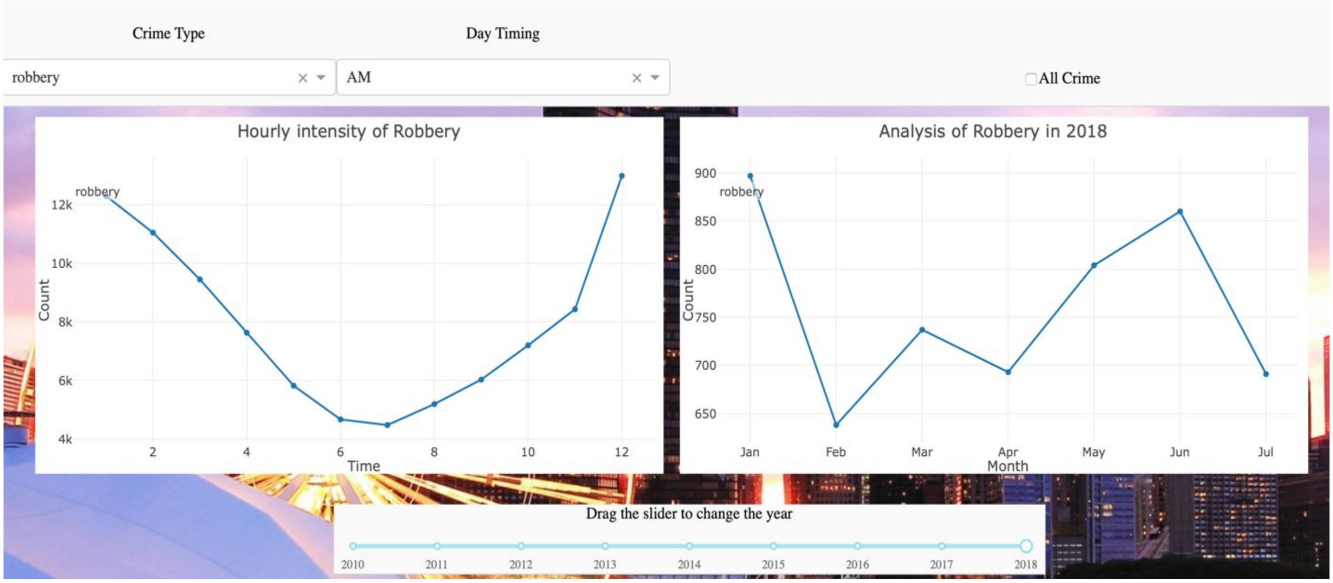
The master table is then used to perform various data operations where we have created functions conforming to standard practices and user-friendly for developers. Data is then used to create child tables using spark dataframes, which are required for the end user visualization and interactive plots, and are stored back in Cassandra. We have provided flexibility for data loading and saving in multiple formats like csv, Cassandra. Also added features for dropping/truncating table with convenience.

2.2 Exploratory Data Analysis

We performed Exploratory Data Analysis (EDA) to gain better insights into the data and have a better understanding of the main characteristics. Analyses below:

2.1.1. Interactive Analysis

To analyze crime trends (hourly, monthly, yearly) we have created an interactive application. Below is the illustration, whereby changing Crime Type and Day Timing, year slider options we can visualize dynamic crime count in the plots.



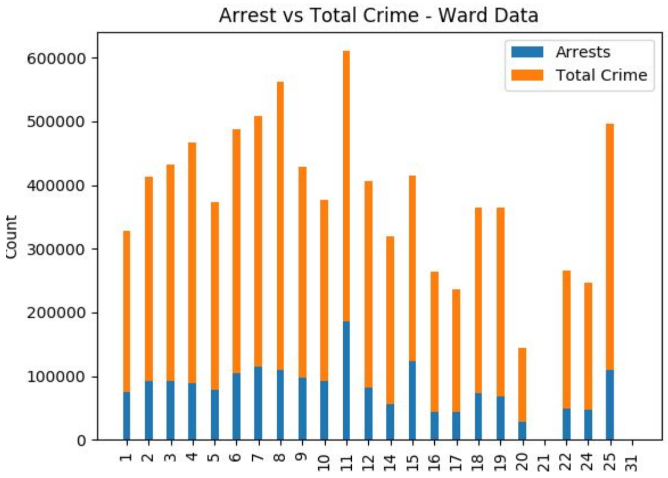
Crime Type: robbery

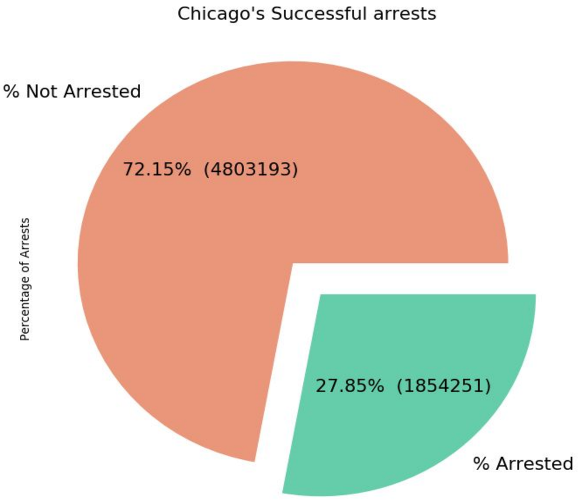
Day Timing: AM

□ All Crime

2.2.2. Static Analysis

After careful consideration of metrics, we narrowed down our results to focus on main parameters like arrest ratio, crime type, location, which influence overall safety and can result in increased security if controlled.





2.3. Module Development

2.3.1. Time Series Forecasting

Time Series analysis helps us to understand the pattern for a set of metrics with time. We used Facebook's Prophet Python library to forecast crime count for next year. It is based on an additive model where non-linear data can be fit in yearly, monthly, quarterly, seasonal trends.

When compared to classical forecast models like ARIMA and VARMA, fbprophet is robust to missing data and handles outliers. To increase the accuracy of predictions made using classical models, one has to spend a lot of effort in tuning each parameter whereas in fbprophet parameter tuning is easy. Along with forecasting, fbprophet provides additional features like historical trends which is quite useful to understand the overall trend (increasing, decreasing, persistent).

2.3.2. Deducing Crime Severity

We deduced crime severity based on crime type and crime description. For deductions, we studied how crime reporting agencies are classifying criminal incidents based on the primary and secondary description of the crime (IUCR codes). We found that crimes with similar descriptions are assigned IUCR code in almost the same range. Generally, homicide as crime type is by default considered as a severe crime but for many other crimes that cannot be categorized as severe just by looking at crime type (Narcotics) can also be further classified as severe and moderate on basis of crime type and crime description. For instance, if crime type is "Narcotics" and its description is "Manufacturing of cocaine" then it would fall under the severe category, whereas, if crime type is "Narcotics" and its description is "Possession of marijuana under 30 gms" then it would fall under the moderate category. Using this methodology, we classified data into severe and moderate categories and plotted bar charts of major crime types.

2.3.3. Word Cloud

To find the most frequent crime type and location, we used WordCloud as our data visualization technique. WordCloud is an image of words where the size of each word indicates its frequency/importance in data. We implemented WordCloud using Python's wordcloud library and confined the word cloud within the Chicago contour image.

2.4. Web Application and Visualization

2.4.1. Python and Flask

We chose Flask as our web application framework as it is lightweight and provides better compatibility with Python. Whereas, frameworks like Django is heavier and less flexible than Flask.

2.4.2. D3 js vs. Dash for web app

We preferred Dash over D3.js for interactive visualizations because it removes the hassle of JavaScript. Plotly (built on top of D3.js) is specifically a charting library whereas D3.js is just a framework to manipulate elements of document/DOM. Dash is ideal for building analytical web applications with a high custom user interface in pure python and is built on top of react.js and plotly.js.

2.4.3. Plotly vs Tableau

With Plotly and Dash, we don't need to embed dashboard/reports to webapp whereas with Tableau we need to embed using iframe and public URL.

3. Problem and Challenges

3.1. Cassandra and Dash Integration

Dash does not provide an environment to keep cassandra server up and running, once the data was loaded the connection to the cassandra table was getting lost ergo hindering our app to host the web app. Resultantly, we incorporated Flask framework over our app to run it as a server to host our application.

3.2. Spark dataframe and Dash

Data is highly diverse, hence we are using multiple normalized tables for different modules. To incorporate tabular data in dash, dash in itself does not possess feature to directly handle tables/dataframes, instead, it works on lists. Due to this, each of our loaded datasets has to be converted to a list, which is a computationally heavy task to perform. We identified two methods to do the underlying task and based on the minimum time taken we've used these methods accordingly.

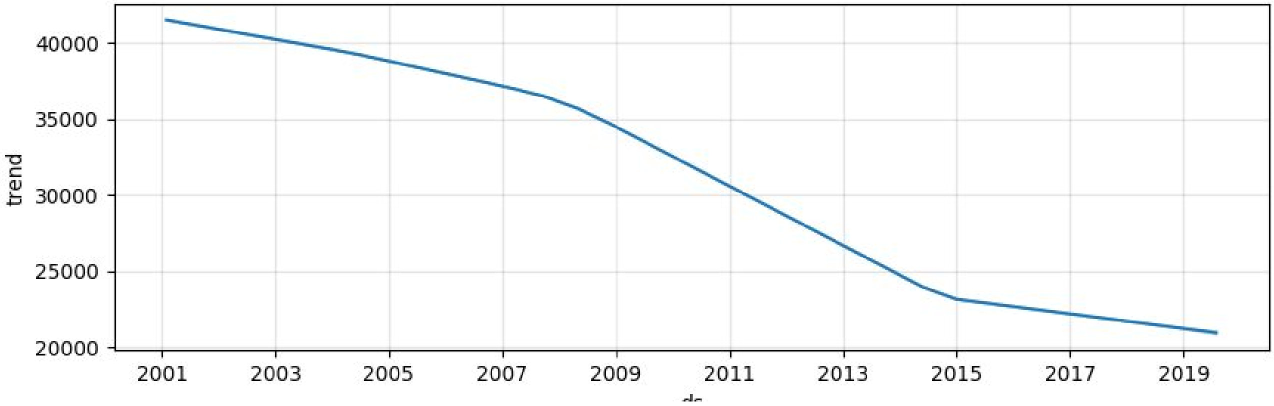
3.3. UI Visualizations

A major challenge we faced was switching between multiple tabs as dash doesn't allow switching between tabs outside the main layout. Another challenge we could face is while doing operations on a bigger dataset to get the data computed for interactive visualization tabs. The application runs swiftly for datasets sized between 200MB to 500MB, though, handling large scale data using dash is efficient however heavy visualizations require excessive computations to be done on limited resources, therefore it might result in a slight delay in loading our application.

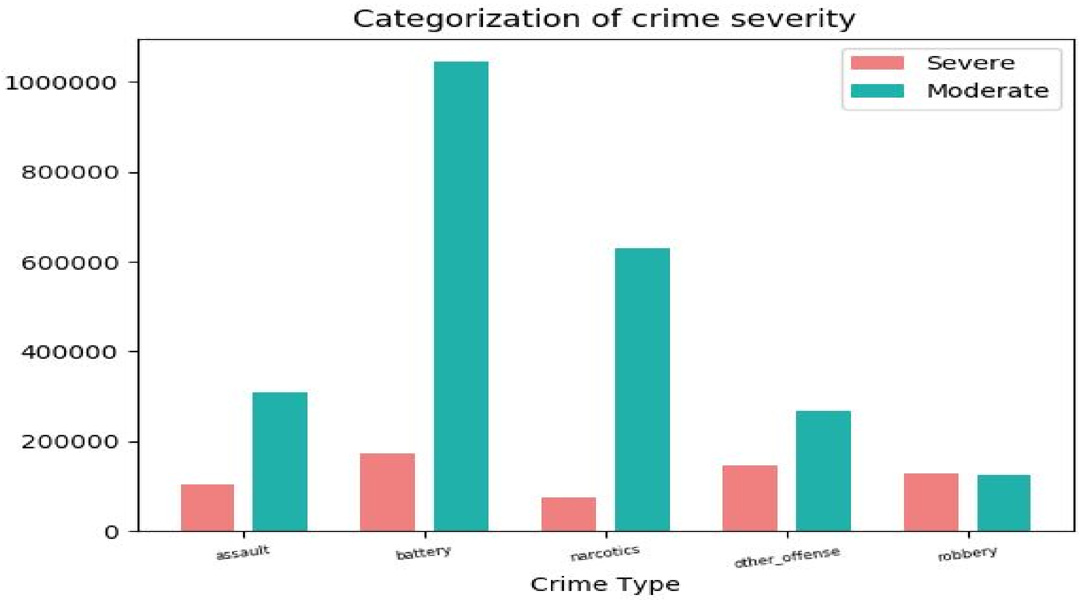
4. Results

The results of our projects shows that overall only 27.85% of the criminals that were reported, were arrested (Fig. 2 in section 2.2.2). Ward/district wise segregation can be seen in Fig. 1(section 2.2.2) indicating most affected regions which might require more security measures in the future. Amount of crimes reported in recent years has shown a declining trend with a considerable decrease in the years 2015-2018, which shows security in Chicago has increased over the years.

4.1. Forecasting: Originally, the predictions for the year 2019 were having some outliers due to which they seemed unrealistic. After getting feedback from Professor Gregory Baker during the demo, we trained our model on data of 2001-2017 years and predicted crime for the year 2018 to check how well the model output aligns with the original data of 2018, and then to check the accuracy of the predictions of 2019. Our model's historical trend predicts that the starting week of march has the maximum no of crimes happening.



4.2. Severity: Crimes not involving the use of physical force/ use of firearms on another person have major proportion categorized under moderate category than severe.


Categorization of crime severity

4.3. Crime Interpretation:

We added the feature to visualize the data based on timing of the day, which is useful to categorize crime more frequent at a particular hour. We observed, overall the months of June and July are more prone to crimes. The same can be observed from the interactive plots in the web app.

Robbery- It is more frequent in at night during 8pm-10pm and has high occurrence in the months of August to October (results can be seen in Fig. in section 2.1.1).

Assault- Assault is persistent around 9am-11am while recurrent in the May and June month across years.

5. Project Summary

Category	Summary	Points
Getting the data	Datasets from Chicago Data Portal, Kaggle https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2	1
ETL	Cleaned, Filtered and Transformed the dataset using Spark Dataframes and Spark SQL functions.	4
Problem	How predictive and exploratory data analysis can help in reducing crime.	1
Algorithmic work	Crime severity deduction using crime type and description, Time Series forecasting of crime data using FB Prophet, Successful crime arrests computation.	3
Bigness/Parallelization	1.5 GB data with 6.6M unique crime records.	2
UI	Developed an interactive web application hosted on Flask embedded with Dash dynamic plots.	3
Visualization	Interactive line charts and Geomap plots created using Dash and Plotly, Word Clouds for Crime type and location, Forecast and trend plots using FB Prophet, Bar and pie charts using Matplotlib, Seaborn.	4
Technologies	**New:** Dash, Plotly, Seaborn, FB Prophet, WordCloud, Integration of Flask and Dash server **Existing:** Cassandra, Spark, Pandas	2
Total		**20**

Link to Gitlab - https://csil-git1.cs.surrey.sfu.ca/akunwar/chicago-crime-analysis