# Location Analytics Project: New-York Crimes Prediction

Alaeddine DHAOUI & Mohamed CHIBOUB

Recognizing the patterns of criminal activity in a place is essential to help authorities prevent it or to alert and advise citizens during their travels. Law enforcement agencies can work effectively and respond faster if they have a better understanding of crime patterns in different geographies of a city. The objective of this project is to set up a web solution to fight and prevent crime. We were able to achieve a very high accuracy using an ANN model and predict which type of crime which will occur, depending on the position of a potential victim, and his/her personal information.

Additional Key Words and Phrases: Location Analytics, Open Layers, Artificial Neural Networks, Geolocation

## 1 INTRODUCTION

Crime rates in New York city have seen massive change since the 1980s, after officials starting adapting new ways to minimize the crimes that were spiking in rates at the time. So the aim of this paper is to propose a way of detecting what kind of crime that can occur to a potential victim based on multiple input information. For this matter, we will be using an artificial intelligence model using Artificial Neural Networks. Various methods have been used to generate a very high accuracy of the model used. Once the prediction model was tuned, we were able to create an intuitive interface, that allows a victim to select his/her position on the city of New York, enter his/her personal information and then predict the crime that can occur to him/her, in that location, at any particular time.

## 2 DATA SET

The data used to train the artifical intelligence model was brought from the NYC Open Data website. It is free public data published by New York City agencies and other partners. In our case, we used a database of more than 6 million different crime records dating back to 1915.

## 3 DATA PREPARATION

## 4 PREDICTION MODEL

## 5 RESULTS

## 6 WEB APPLICATION

In order to properly use our prediction model, we have to incorporate it within a web application. In this case, we used FastAPI and Vanilla Javascript in order to make it.

### 6.1 Frontend

The frontend was made using vanilla javascript and mainly used OpenLayers to use geolocation capabilities, such as displaying the map of New York, adding layers on top of it, etc.
In our case, the frontend consisted of two parts: the first part was mainly the map, where the user can click anywhere on the map. If the map lies within the surface of New York, then that point will be marked as the user location.

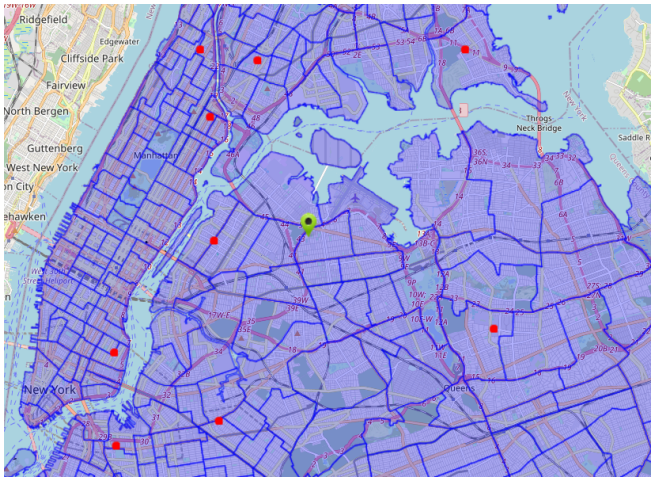


Fig. 1. Selecting User Location using Open Layers Map

Users can activate different layers. The first layer is the different boroughs of New york (marked with the blue color) and the second layers is the different locations of main police stations in New York.

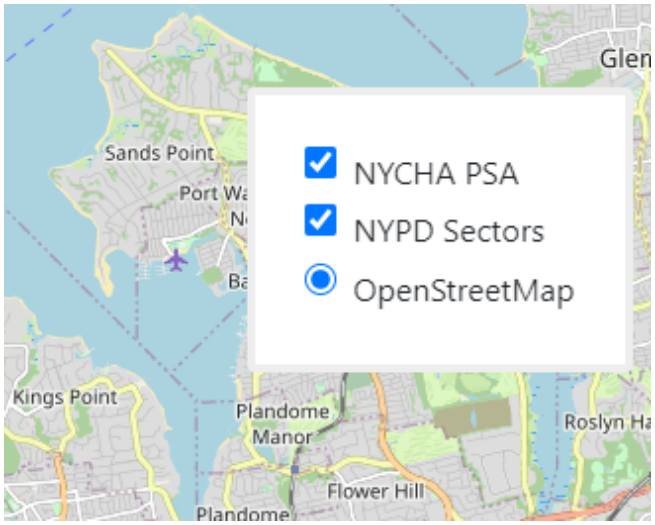


Fig. 2. Layer selection inside the map

Right after clicking on the map, we also save current location information, such as the borough code of the selected area, the precinct, etc. This information will be sent later when we want to predict the crime.
As for the second main part of the frontend, we have the prediction form, which must be filled in order to be able to send the prediction request to the backend. Several input fields are required, such as the

age of the victim, its race and sex, the date of the supposed crime, as well as the coordinates of the crime. But note that the latter is automatically filled, as long as the user chooses a location on the map.

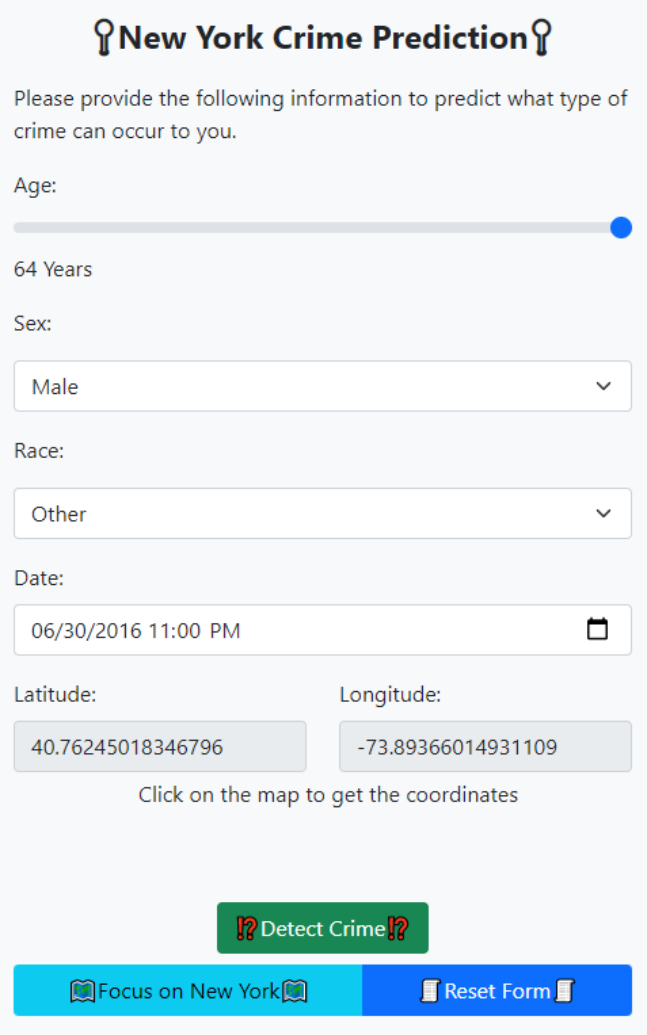


Fig. 3. Filling out the crime prediction form

After filling out all the information, the "Detect Crime" button will become clickable. So once it is clicked, a request will be sent to the backend with all the information of the victim. The result is then returned and we show it via a toast popup.

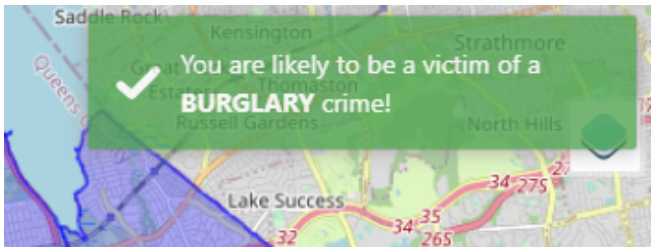


Fig. 4. Showing the Result Toast

## 6.2 Backend

As for the backend, we used FastAPI to create an CPI that will receive all the information from the frontend, call the prediction model using that info and finally get and return the predicted crime. For this, we loaded all the dictionaries necessary to convert the data that the model needs. We also enabled CORS. The API works correctly and the response time is very minimal. Testing using Postman was done in order to make sure the the API works correctly.

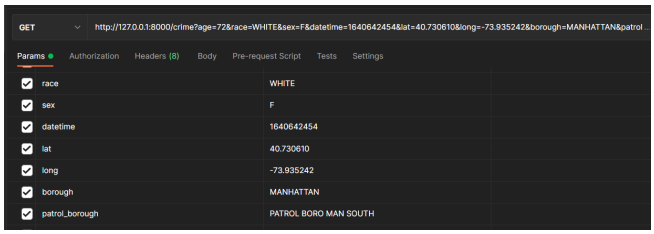


Fig. 5. Example of a request made using Postman

The result is sent as JSON and is parsed correctly in the frontend. It only contains one attribute which is the crime that is committed.

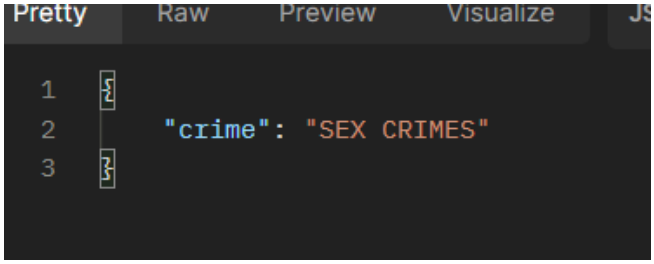


Fig. 6. The JSON Response from the prediction API

As an optimization, we added a FastAPI caching plugin, which uses redis in-memory database, in order to cache requests, this makes the prediction much faster for repetitive requests.