Predicting Crime Frequency in Prince George’s County, Maryland

Cole Bromfield, Edmund Park, Dennis Reyes

University of Maryland, College Park

DATA602 – PCS1, Fall 2023

Prof. Mohammad T. Hajiaghayi

**Predicting Crime Frequency in Prince George’s County, Maryland**

**Introduction**

Located in Maryland just outside of Washington, D.C., Prince George’s County offers comprehensive data on crime incidents within the county, going back to 2017. Using Python along with packages such as requests, Pandas, and Geopandas, this data can be put into a format that allows for easy analysis and visualization. Prediction models such as regression analysis allow for the next-month projections of the total quantity of crime occurrences, as well as more specific subcategories of crime such as homicide, violent crime, etc.

**Background**

The relationship between crime and the calendar is a long-discussed topic—it is common knowledge that murder rates increase in the summer, resulting in the famous correlation between ice cream sales and murder, oft-used by statistics professors in an attempt to explain the difference between correlation and causation. What other trends can we find when we examine monthly crime data over 5 years’ time? Are certain regions within the county more prone to certain types of crime than others? Using this past November as testing data, the goal of this project is to predict the number of crime incidents in Prince George’s County in December of 2023, with the ability to select for location and subcategory of crime.

**Acquiring the Data**

The historical data was found at data.princegeorgescountymd.gov and imported using requests and Json. We converted the data to a list of dictionaries, each containing the details of an individual crime occurrence. From there, we were able to construct a Pandas DataFrame consisting of every crime occurrence from February of 2017 up until July of 2023. For each crime instance, this DataFrame showed the case ID, date, crime type (e.g. theft, homicide, etc.), reporting area, sector of the county (represented by a letter), beat (subsections of each sector), street at which the incident took place, latitude, longitude, and street number. The data pertaining to July of 2023 through the present was found at a separate location on the same website, and the same process for importation was used to create a DataFrame from this data before the two DataFrames were merged into one. Only minor cleanup was required at this point—using Python’s Datetime package, the entries in the “Date” column were converted to Datetime objects, allowing for easier date-related analysis. This allowed for the creation of an additional column displaying the day of the week for each crime occurrence, potentially to be used as an extra feature in making our prediction. The head for the cleaned DataFrame can be seen in figure 1 A screenshot of a computer

Description automatically generatedbelow.

Figure

**Regression Analysis**

To generate predictive data about aggregate crime throughout the county, we turned to a logistic regression model, specifically polynomial regression. From the data we compiled for February of 2017 through November of 2023, we found the best method to be to sort all of the crime instances into one-month bins, displaying the total number of crimes occurring in any given month. This process required extensive effort to clean up and process the initial data, but after this initial processing we were able to put A graph of crime in pg county

Description automatically generatedthe monthly data into a scatter plot, providing a clear visual for the total number of instances of crime for each month in the specified period (Figure 2). To minimize complexity of regression analysis, the x-axis represents the number of months passed since February of 2017 (in other words, February of 2018 falls at , February of 2019 falls at , etc).

Figure 2

A graph with blue dots and red line

Description automatically generatedThe subsequent task was to fit this data to a curve. To avoid a polynomial that would overfit or underfit the data, we ran the regression calculation using 80% of the data as training data for every possible degree of polynomial, from 1 (a line) to 5, to determine which generated the lowest mean absolute error with the remaining 20% of the data, used as testing data. Finally, we used this model to predict the amount of total crime in November of 2023 to test the accuracy before using the model to predict December of 2023. Ultimately, the degree 4 polynomial (see figure 3) was found to generate the lowest mean absolute error, and predicted 2,789 total crime instances for the month of December 2023.

Figure 3

A graph with a red line and blue dots

Description automatically generatedAn identical process was done with various types of crime. After separating violent crime into its own DataFrame (using the categories “Homicide”, “Assault”, “Assault, Weapon”, and “Assault, Shooting”, we performed the same regression analysis, resulting in a 5th degree best fit polynomial (Figure 4). Under this model, we found a prediction of 124 violent crime occurrences for December of 2023.

Figure 4

A graph of a red line with blue dots

Description automatically generatedLikewise, car accidents can be tracked in a similar fashion. As we can see in Figure 5, there exists a weakly positive correlation between time and monthly accidents, and the model predicts 519 total car accidents for December of 2023. However, there is a caveat when predicting car accidents—their distribution across any given year is not uniform. During the winter, a greater number of car accidents will naturally occur, due to the greater likelihood of hazardous driving conditions (e.g snow, ice, sleet). Therefore, a regression model that inputs two features may prove more useful in the case of car accidents (the same can be said for violent crime and its higher likelihood in warmer months), with the second feature being the integer value of the month (e.g. 10 for October), not including the year. Interestingly, this value actually comes out lower. However, this is likely due to the overall upward trend in the data more than cancelling out the seasonal differences in accident quantity. As a result, we determined that it was not sensible to use the integer value of the month as an additional feature, and instead stuck with the more common, single input value linear regression calculation to predict total crime for the ensuing month.

Figure 5

A graph with blue lines and black dots

Description automatically generatedA graph with blue and black dots

Description automatically generatedThe next step was to narrow our prediction to an interval more compatible with a live prediction model—namely the next week as opposed to the next month. Although we attempted to use the same polynomial interpolation model as we had used for months, we found that it was considerably more computationally expensive, and opted instead to use Prophet, a Meta-created Python package intended to generate predictions based on time series data. Prophet’s prediction model carries the additional benefit of accounting for seasonality (e.g. month, season, day of the week, etc) and use of uncertainty intervals. Our new Prophet-generated predictive model can be seen in figure 6. For a more accurate model, we removed outliers (figure 7). As an example projection, the latter model predicts 650 total crime incidents for December 11th through December 17th. Finally, we generated a third Prophet model that incorporated holidays (figure 8). Prophet also allows for the generation of more detailed componentwise trend graphs (see A graph with blue and black dots

Description automatically generatedfigure 9). (Details on interpreting each of these graphs go here). To analyze the effectiveness of each method, we calculated the mean adjusted percentage error (MAPE) for each model (with the previous week as training data) and the A screenshot of a graph

Description automatically generatedresults were as follows: 25.31 for the first model (including outliers, not counting holidays), 13.25 for the second (excluding outliers, not counting holidays), and 13.21 for the third model, which excluded outliers and counted holidays.

Figure 6

Figure 7

Figure 8

Figure 9