Understanding Crime and Predicting Crime Rate in the City of Chicago

Parth Mishra
Computer Science Dept.
Michigan Technological University
parthm@mtu.edu

Abstract

With increasing amounts of data and computing power, we can answer more complex questions and identify underlying trends, in this project we try to forecast the crime rate in the city of Chicago. This is an interesting question because through this we can understand how the current policies and practices to prevent crime are working and what trajectory can we expect in the future. Our data will be time-series-based and to predict this data we will be using two different approaches: **FBProphet, ARIMA** and **LSTM**. FBProphet is a custom package created by Facebook for time-series prediction and ARIMA which stands for autoregressive integrated and moving average and is a classic time-series analysis method widely used in the stock market to validate predictions. LSTM is a sub-category of Recurrent Neural Network, which can mitigate the problems which are encountered while training recurrent neural networks; the vanishing gradient problem and the exploding gradient problem. To compare the three approaches we will use three metrics: RMSE (Root mean squared error), MAPE (Mean absolute percentage error), and MAE (Mean absolute error). We will see that ARIMA and LSTM perform very well and the LSTM also outperforms the ARIMA after training correctly.

Introduction

Data as we know is modern-day oil. Today companies like Google, Facebook, and Apple have immense amounts of data and in my personal opinion, their revenue line is in direct correlation with the amount of data they are collecting. The amounts of data generated by humans have skyrocketed in the past couple of years. Here are some facts on data generated in the world [1]:

During the year 2020, each person has produced 1.7MB of data per second. A staggering 90% of the world's data has been generated in the past two years alone. Every day, humans create 2.5 quintillion bytes of data through various channels.

But the key point in all this is, data is useless until it is mined properly to find out hidden trends and insights with which we can make viable, productive, and actionable data-driven decisions. This is where Data Mining comes into play. Data Mining is the incorporation of mathematics, statistical methods, and algorithms to mine the generated data and provide value to an organization through finding out key trends and insights [2].

Data mining can be applied to a plethora of possible places, one avenue which can benefit tremendously from data mining is the public services, more specifically, the police department [3]. Each day numerous crimes are committed in a city which can range from petty theft, burglary, embezzlement to more violent crimes. This requires a different degree of preparedness by the police department, there are various things a department has to identify like which areas of the city are more prone to crime, what types of crime are being committed more, what kinds of places are more prone to criminal activities, what strategies and policies seem to be working, how has been the overall trajectory of crime in the city over a while.

Helping answer these questions might not provide business values but it provides something much more important; better public service by increasing efficiency, improving response time, effective strategy evaluations, and resource allocation. All this leads to a happier and safer city. As more and more data is being collected by the police departments, more scope to mine this data is being created, and as mentioned above helping public services directly impacts the health of the public.

The city of Chicago has provided a very detailed dataset that is open for analysis and this was a very nice opportunity to work on public services data and use the different time-series analysis models and compare their performances. We will primarily be working on the questions of predicting the crime rate and visualizing the distribution of crime in the city through various techniques which will be detailed further.

Background

In this project we will be using the dataset provided by the City of Chicago This dataset reflects reported incidents of crime (with the exception of murders where data exists for each victim) that occurred in the City of Chicago from 2001 to 2017, the dataset contains more than 6,000,000

records/rows of data and cannot be viewed in full in Microsoft Excel. To be more accurate the shape of the data is 7941282 rows × 23 columns. We will start by preprocessing which would be required to make our data more model-friendly. I am not aware of any other usages of this data set and this dataset was sourced from Kaggle and is also available on the City of Chicago website [6].

In our objective to predict the crime rate, we would have converted our dataset into a more time-series-friendly format. As our problem is essentially a time-series prediction problem, we will have to use models which are more specifically tuned to these kinds of problems namely, FBProphet and ARIMA. We will discuss these two models below.

FBProphet

This model was first introduced by Facebook in 2017, which tried to define a realistic approach to large-scale forecasting that blends configurable models with an analyst-in-the-loop type of performance analysis. They also suggested a modular regression model with observable parameters that can be modified intuitively by analysts with domain knowledge of the specific time series [4].

Prophet is an open-source software introduced by the Facebook Core Data Science team and is available for R and Python. For python, Prophet follows an sklearn API style and implements the methods of fit and predict. There are various advantages of prophet; it uses a multi-component time series and various factors like seasonal trends, holidays, or events effect and error are taken into effect. The equation is given below.

$$f(x) = g(x) + h(x) + s(x) + e(t)$$

In the above equation $f(x)$ represent the combined effects of piecewise trend $g(x)$, holiday effects $h(x)$, seasonal trends $s(x)$ and noise $e(t)$.

FBProphet fits multiple regression lines to the model and uses time as a regressor. At first, it will try to fit a linear model but this can be changed to a non-linear model based on its arguments.

FBProphet is intriguing, sophisticated, and easy to use. Its major plus point that it has many different parameters and knobs to manage the model. It also gives the user the subcomponents which make up the final model and their effect on the final model can be tuned all these factors combined together can result in strong forecasting performance with little effort or domain information in time-series analysis.

ARIMA

ARIMA is a tried and tested simple yet powerful time-series prediction model [5]. It is known for analyzing the probabilistic properties of the variables and is based on univariate analysis Let us break down parts of ARIMA to better understand how works:

- **AR: A**uto-**R**egressive, this property helps predict future time points based on past time points and their effect. Here we calculate a final variable based on previous values.

  $Y_t = f(Y_{t-1}, Y_{t-2}, Y_{t-3}, \dots, \epsilon_t)$, where epsilon is the error and Y_t is the prediction which is based on the previous values of Y plus the error epsilon $f(Y_{t-1}, Y_{t-2}, Y_{t-3}, \dots, \epsilon_t)$.
- **I: I**ntegrated, If there is a theme, the time series is called non-stationary and seasonal. Integrated is a time series property that decreases seasonality. ARIMA versions vary to the point that seasonality is omitted. This is where the integrated property of ARIMA comes in.

- **MA: M**oving **A**verage, here the future predictions are based on the error terms of the previous time points.

  $Y_t = f(\epsilon_t, \epsilon_{t-1}, \epsilon_{t-2}, \dots)$, here Y_t is the output of the function which takes into effect the errors from the previous predictions $f(\epsilon_t, \epsilon_{t-1}, \epsilon_{t-2}, \dots)$.

We will use pmdarima which is a pre defined Python library which makes deploying ARIMA easy.

LSTM

Considering that we have a deep RNN, the inputs supplied to the different nodes of RNN produce some gradient, if the gradient is less than 1 then, in the future it can vanish completely, whereas if the gradient is greater than 1 it can explode. In both cases, the outcome accuracy of the model decreases and the training time increases. A solution to these problems is addressed by an extension of RNN called Long-Short Term Memory or the LSTM model.

LSTM addresses the vanishing and exploding gradient problem by introducing four neural network layers in each module As seen in the *fig 1*.

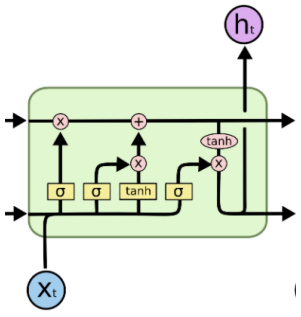

Fig 1. LSTM node

Each node introduces three different gates

• Forget gate: This gate decides which information to throw and how much to throw it takes into account h_{t-1} and x_t and finally returns a value between 0 and 1 where 0 is 'remove all' and 1 is 'keep all'.

• Input gate: This works in two parts, the first sigmoid layer decides which values should be updated, this is done by the sigmoid layer, second, the $tanh$ function creates a new vector and then combines it with the output values from the sigmoid layer and adds to the cell state.

• Output Gate: After passing h_{t-1} through the forget and input gate the output of these decides the cell state. The previous history is passed through another sigmoid layer which is then multiplied by the output from the previous two gates after passing it through another $tanh$ function to produce the next history.

Project Architecture

To implement the above two models we will have to preprocess our data and convert it into time series, and we will use this data to create visualizations to better understand the data and also predict the crime rate. We will follow the following pipeline for this project:

• **Data Preprocessing:** This part will include loading the datasets, concatenating different files, dropping columns that do not contribute to the model or are otherwise not needed, standardizing the date-time column using pd.to_datetime this is done because we are conducting a time-series analysis and we want the dates to be in a uniform format, checking for null values in the columns and dropping the rows with null values. Once we have our final datasets, for the first two models, we can use a common train/test dataset in which we will use the last 12 months as the test dataset. For the LSTM, since it is recurrent in nature, we will have to create batches of data, and for each batch, we will take into account the previous 12 months, so each element in the X_train is a list of 12 values which represents the last 12 months and the 13th-month value is stored in the y_train.

• **Data Visualization:** This involves visualizing the processed data and understanding how the distribution of crime throughout the data, we try to understand which types of crimes are most common, where are the crimes most commonly committed, and the general trend line over the years. We will use a count plot to demonstrate our findings

• **Deploy FBProphet:** Here we try to implement our first model, the FBProphet, since it is similar to the sklearn model methods we will be fitting the model.

• **Deploy ARIMA:** Here we try to implement our second model, the ARIMA, we will be using the pmdarima for our purposes.

• **Deploy LSTM:** Here we try to implement our third model, the LSTM, we will be using the Tensorflow for our purposes and more specifically, we will be using the Keras API.

• **Report Results:** Once we are done with building all the three models we will be comparing the RMSE and the MAPE values from all the three models to identify which one performs the best.

Model Specifics

In this project we are using three different models and will be comparing their performance in predicting the crime rate, below are the specific models parameters which we will using and their corresponding output graphs which can provide some more context to their accuracies.

FBProphet Model Details

We will be using prophet from the Prophet library. The input to the prophet models has to be very specific and it can contain only two columns; one for the date and the second one to the count, these columns are required to be specifically named by this module before being fed into the forecast, the is done for 12 successive periods and in this case, each period is a month. The forecast will intern give us a final data frame with all the additive terms contributing to making the final prediction. *Fig 2.* Represents the trend line produced by the model (yellow), the green line represents the final prediction value.

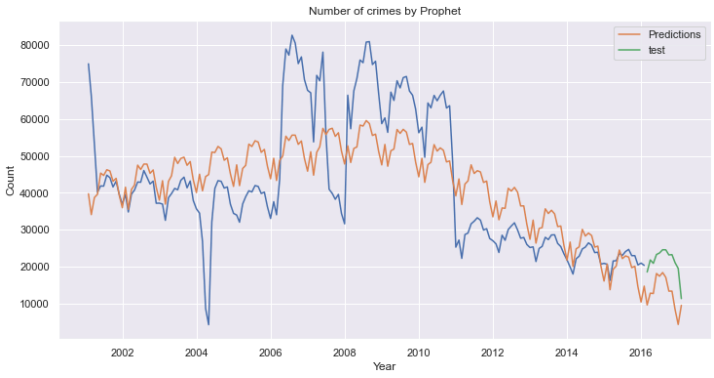

Fig 2. Representing the output of the Prophet

ARIMA Model Details

Here we use the pmdarima module very efficiently determines the best flavor of ARIMA based on our data, in this case, the output model turns out to be a SARIMAX model which also makes sense, since our data has season-

al-components to it, the SARIMAX model is an extension of the ARIMA model which takes into account the seasonality of the model which is signified by the "S". *Fig 3.* Shows the test data in yellow and the predictions in green.

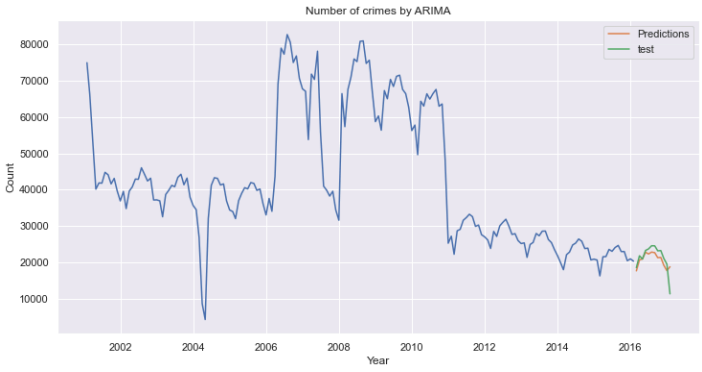

Fig 3. Representing output of the ARIMA

LSTM Model Details

We will be using the very famous TensorFlow to deploy our LSTM model, for this model we will be using the train test from the earlier models but we will have to modify the dataset before we can feed it to the LSTM. The first step would be to scale the data between 0 and 1, as this approach of scaling helps the optimizer to restrict the data range and has shown to produce optimal results.

Then we will have to create a new training set for the LSTM, for this, we will take 13 values, the first 12 we append to a list which is added to the training set, the 13th value is added to the test set this way we have a mapping of 12 values to the 13th value, and this is what we want because in our LSTM we will be taking in account the effect of the previous 12 months. The same is done for the testing dataset.

In this LSTM we will be using 264 nodes, the $tan(h)$ activation function, we will also be using a dropout layer to increase regularization which will make 10% of nodes ineffective in each epoch, we will be using this function as this was shown to perform better than ReLu while testing, we *"adam"* optimizer while compiling and we will train the model to 600 epochs. *Fig 4.* Shows the output graph in which test data is represented by yellow and predictions are represented by green.

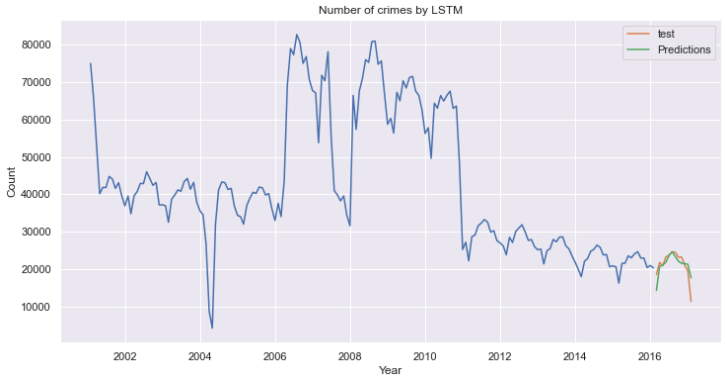

Fig 4. LSTM prediction output

Results

In the final dataset, we will only be keeping 6 columns we are keeping these columns as based on their descriptions they are the best features to answer the questions we ask in the project; types of crime, locations of crime, and the crime rate. Following is the description of each of the features.

- ID: Unique identifier for the record.
- Date: Date when the incident occurred. this is sometimes a best estimate.
- Primary Type: The primary description of the IUCR code.
- Location Description: Description of the location where the incident occurred.
- Arrest: Indicates whether an arrest was made.
- Domestic: Indicates whether the incident was domestic-related as defined by the Illinois Domestic Violence Act.

Date	ID	Date	Primary Type	Location Description	Arrest	Domestic
2004-01-01 00:01:00	4786321	2004-01-01 00:01:00	THEFT	RESIDENCE	False	False
2003-03-01 00:00:00	4676906	2003-03-01 00:00:00	OTHER OFFENSE	RESIDENCE	False	True
2004-06-20 11:00:00	4789749	2004-06-20 11:00:00	OFFENSE INVOLVING CHILDREN	RESIDENCE	False	False
2004-12-30 20:00:00	4789765	2004-12-30 20:00:00	THEFT	OTHER	False	False
2003-05-01 01:00:00	4677901	2003-05-01 01:00:00	THEFT	RESIDENCE	False	False

Figure 5: Dataset description

Visualizing Data

Here we visualize the distribution of the types of crimes being committed, the location where they are committed at and the overall trend. After visualizing the data the following graphs can be seen. Each graph represents some aspect of the insight we wanted to find out.

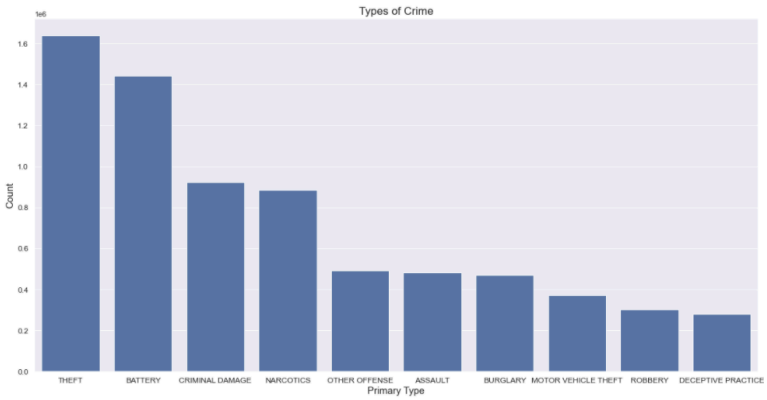

Fig 6: Ten most commonly committed crimes

In the graph above you can see the 10 most commonly committed crimes in the City of Chicago. Here as you can see theft is the most common type of crime followed by battery, criminal damage, and narcotics.

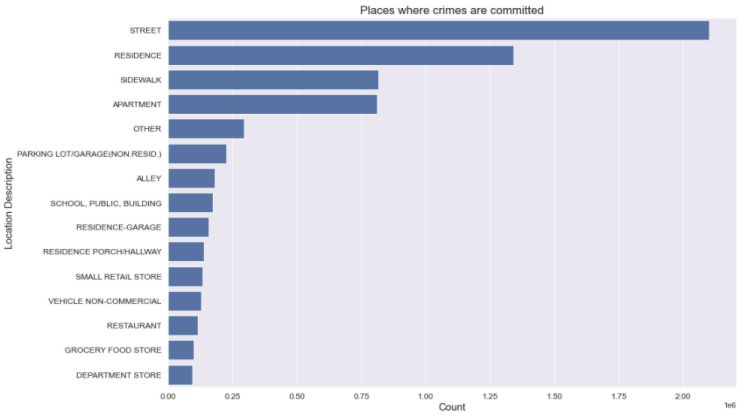

Fig 7: Fifteen most common places where crimes are committed.

The above graph represents the 15 most common places to commit a crime. As you can see in the above figure, the street is the most common place where crimes are committed followed by.

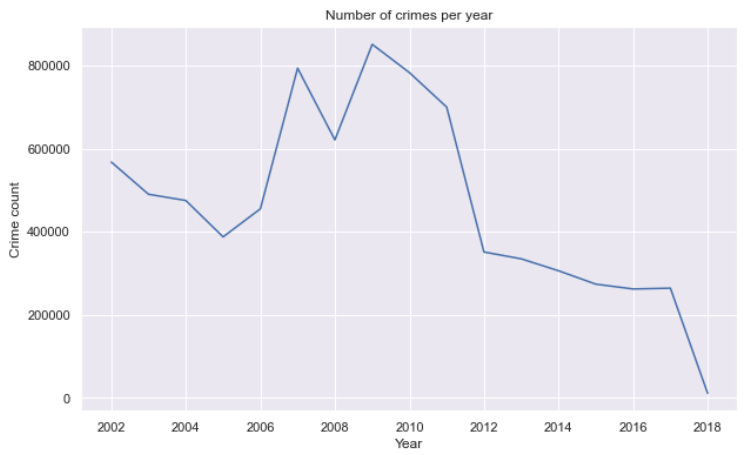

Fig 8. Overall trend from 2002 to 2018

The above graph clearly shows that there has been a significant drop in crime in recent years, although there was a huge bump between 2007 to 2011, it was quickly controlled.

Crime Rate Prediction

Now, we will see the results of the three models that we have used, in the table below we can see our output.

RMSE & MAPE Table		
Model	***RMSE***	***MAPE***
FBProphet	39.336	9026.302
ARIMA	11.113	2588.719
LSTM	9.901	2409.730

Table 1. RMSE & MAPE for different models

As you can see in the above table the LSTM outperforms the other models with an RMSE of 2409.7 and MAPE of 9.9% followed by the ARIMA which is followed by the FBProphet

Conclusion

In this project, we understood the crimes committed in the City of Chicago, the types of crime committed, the overall trend. We also demonstrated various models to analyze time series like FBProphet, ARIMA, and LSTM. Each model posed its own set of challenges. It is also very important to understand that no model is best for everything, and each model has its pros and cons. It would be better to understand the problem at hand understand the underlying data and apply these models. In recent research, the BERT transformer has been shown to perform exceptionally well, which can also be implemented here.

Police forces interact with the masses more directly and physically than any other public service, a good policing system can lead to a very well-organized and happy society, data-driven decisions can have decent gains in capacity planning, resource allocation, and preparedness. but as with all cases, with great power comes great responsibility, a police force cannot be completely data-driven because the masses it interacts with are not data-driven, there is a human element at play here, and no data can encompass empathy and basic common sense.

Note that this project in no manner is complete and able to make any fundamental decisions regarding the police but rather, it gives a very good starting point about where

to go. There have been cases where predictive policing severe consequences and unnecessary harassment. So, it becomes all the more important for the police to use its data judiciously and identify the various areas where it can be useful. We should keep in mind that wrong things done for the wrong reasons are still wrong, and in the words of John F. Kennedy *"A police state finds that it cannot command the grain to grow."*

References

1. https://techjury.net/blog/how-much-data-is-created-every-day/#gref

2. Marriboyina, Venkatadri & Reddy, Lokanatha C. (2011). A Review on Data mining from Past to the Future. International Journal of Computer Applications. 15. 10.5120/1961-2623.

3. Brayne S. Big Data Surveillance: The Case of Policing. American Sociological Review. 2017;82(5):977-1008. doi: 10.1177/0003122417725865

4. Taylor SJ, Letham B. 2017. Forecasting at scale. PeerJ Preprints 5:e3190v2 https://doi.org/10.7287/peerj.preprints.3190v2

5. MAKRIDAKIS, S. and HIBON, M. (1997), ARMA Models and the Box–Jenkins Methodology. J. Forecast., 16: 147-163. https://doi.org/10.1002/(SICI)1099-131X(199705)16:3<147::AID-FOR652>3.0.CO;2-X

6. http://data.cityofchicago.org/Public-Safety/Chicago-Police-Department-Illinois-Uniform-Crime-R/c7ck-438e