Impact of Covid-19 Pandemic on Criminal Activities

Survey of Programming Languages
ITCS-4102/5102

Final Project Report

June 27th, 2022

Group 6:

1. Sumati Bele
2. Rucha Pramod Visal
3. Dhvani Patel

Table of Contents

1. Introduction

The COVID-19 pandemic has been the most serious public health emergency in the 20th century. It has spread through every continent and affected the lives of all individuals in every country at some level. From strict travel restrictions to mandatory testing requirements, COVID-19 has drastically altered human social behavior and destabilized an individual's financial condition.

Every society has a significant amount of crime. Its costs and consequences affect almost everyone to some extent. To investigate whether this pandemic also influences crime, the differences of crime incidents numbers before and during the pandemic in one of the largest cities i.e. Chicago is investigated. The evidence suggests that while a negative impact has been observed, crime rates or types significantly vary across the city. Overall, fewer crime incidents have been reported than before the pandemic began[4].

Although prediction algorithms have previously been used in COVID-19, elections, economy, industry, and other fields, they have rarely been questioned when it comes to predicting crime rates. The purpose of the project is to use crime-identifying trends to predict the future crime rate, as crime has a major impact on public policy, the economy, and politics in general. Specifically, the project will cover several patterns of Chicago's crime rates by pre-processing data and comprehensive modeling of the crime datasets. The machine learning model accurately included the data, allowing for the prediction of Chicago's future crime rates[3].

1.1 Purpose

The objective of this project is to find the correlation between criminal activities (example: homicide, weapon violations) and, financial crisis (2007-2008) and COVID-19 pandemic and create a model to further predict the trends in criminal activities if such a disaster might ever occur in future. The dataset comprises incidents reported in the City of Chicago from 2001 to the present. CRISP-DM methodology will be used to conduct this study that consists of a problem and data understanding, data preparation, modeling, evaluation, and deployment phases. Our aim is to uncover rules for quantifying relationships between attributes (example: public peace violation during both the events).

Thus, the main purpose of this Project is to analyze the effects of the COVID-19 pandemic and how it has affected the crime rates present in Chicago over the years of 2001 and 2021. There is limited criminal research that investigates the connection to pandemics, and how it can be used to reduce crime rates in similar situations. So, Our goal in this project is to reduce crime rates and provide possible policy implications.

1.2 Scope

This project is expected to obtain deeper insights and their visualization of the relationships between the pandemic and crime in the city, and to provide new attempts for crime prediction during the pandemic. From the datasets, we found that a turning point existed, which can divide periods into those before and during the COVID-19 pandemic.

1.3 Definitions, Acronyms, and Abbreviations

(1) COVID-19: Coronavirus Disease 2019

The virus responsible for COVID-19 (previously known as "2019 novel coronavirus") and the disease it causes is a coronavirus called SARS-CoV-2. This is a novel coronavirus. Anyone can get COVID-19 through contact with another person who has the virus. It is predominantly a respiratory illness that can affect various organs. Wide range of symptoms are reported, ranging from mild symptoms to severe illness for infected patients.

(2) CRISP-DM: Cross-Industry Standard Process for Data Mining

An industry-proven method to guide data mining efforts. As a methodology, it includes descriptions of the typical phases of a project, the tasks involved with each phase, and an explanation of the relationships between these tasks."

(3) LSTM: Long Short Term memory

Long short-term memory is an artificial neural network used in the fields of artificial intelligence and deep learning. It is a recurrent neural network capable of learning order dependence in sequence prediction problems. Moreover, it contains feedback connections, i.e., it is capable of processing the entire sequence of data, apart from single image data points[1].

1.4 Project Team Details

1.4.1 Introduction

We are a team of three people. Our strength is that the team members are experienced in the field of software development and Data Analysis. Our team members are also continuous learners and good problem solvers.

1.4.2 Team Profile

Work Distribution:

- **Dhvani Patel:** Dhvani worked on Business Understanding and Data Understanding.

  Business Understanding Stage:
  There are different types of crimes, some of which have a clear link to pandemic, while others do not. It is crucial to think about the plausible directions of causality. Chicago has higher crime rates than any other city. There is an overall declining trend in the number of crimes that occurred between 2001 and 2021. Although, certain types of crimes show an upward trend when disasters or pandemic occur.

  Data Understanding Stage / Exploratory Data Analysis:
  The dataset comprises 7427119 observations and 22 features with int, float, bool and object type values.

- **Rucha Visal:** Rucha worked on Data Preparation and Modeling.

  Data Preparation Stage:
  1. Importing Crimes dataset into Google Colaboratory.
  2. Understanding the information provided in the dataset and looking for types of columns, number of unique values in each column and missing values.
  3. Converted datatype of ‘Date’ column to DATETIME from OBJECT type.
  4. Check for NULL values in the data frame. No null values observed.
  5. Drop the features which are not necessary.
  6. Check for a unique ‘Primary type’ having a count of 36.
  7. Merged the similar crimes into a generalized type for easy processing.

8. Plotted the various trends observed with respect to thefts, assaults, weapon violations, homicide, gambling and narcotics, and burglary and robbery crimes against date to further predict the future trends.

Modeling Stage:
We have used LSTM based deep learning techniques to make time series predictions. For this project, we are considering the top two districts with the highest crime and predicting each crime for a period of one year. For prediction based on the pandemic, we are considering data from 2019-2020 for training and data till Nov 2021 for prediction. It is important to step because that's how we can make appropriate resource allocation decisions.

- **Sumati Bele:** Sumati worked on Evaluation and Designing and Deployment of the project.

Evaluation Stage:
We have used the following metric to evaluate the model- RMSE: The RMSE for training and test sets should be very similar if you have built a good model. If the RMSE for the test set is much higher than that of the training set, it is likely that you've badly overfitted the data, i.e. you've created a model that tests well in a sample, but has little predictive value when tested out of sample.Our baseline evaluation metric for Long Short-Term Memory (LSTM) model was around Root Mean Squared 130.10.

For a pandemic period the RMSE score is:
For District 1 (considering all 7 types of crimes) we achieved an RMSE score of : 22.630
For District 2 (considering all 7 types of crimes) we achieved RMSE score of : 36.367

Design and Deployment Stage:
Designed and deployed well structured python code using various machine learning libraries(keras, PrettyTable etc) and evaluated and developed step by step algorithms according to project requirements and objectives. Apart from that**,**Went through every task with every resource, so we know what to do and when. Make sure to know how to check/ report the tasks as ready, what to do when there are problems, and who to contact if decisions are needed.

2. Programming Language Specifications

In this Project, We have chosen Python as our programming language because of its portability, robustness, flexibility to use Data Science and Machine Learning libraries.

2.1 Why Python?

- Python is an object-oriented high-level programming language that contains data-oriented features for faster data-processing. It has a simple and clear syntax structure and emphasizes clarity and helps troubleshooting code.
- As a result, python is used in a wide variety of places and scenarios: software creation, script-writing, and web-development. These additional functionality compared to HTML, CSS, or JavaScript make it a popular programming language for scientific computing. Moreover, it comes with a regularly updated and well maintained program library that contains functions for Data Analysis.
- A marked difference that separates Python from high-level languages such as C or Java is the feature of dynamic typing. Dynamic typing allows programmers to avoid declaring variables during initialization.
- Fields of data mining, data processing, and data modeling heavily utilize this feature in their code structures. All of these features make Python an incredibly useful and easy programming language to learn and implement.

2.2 History of Python

- Python was created by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in the Netherlands.
- An interesting fact is that naming nomenclature is connected to the famous show “Monty Python’s Flying Circus
- Many different programming languages such as C, C++, Algol-68, ABC, etc we amalgamated to originally form the Python language.
- Python has undergone several major version upgrades from Python 1.0 (Release on 20 February 1991), Python 2.0 (16 October 2000), and finally Python 3.0 (3 December 2008).
- In the course of development many new features such as Cycle Detecting Garbage Collector to a community backed review process have been integrated. It is currently maintained by a core development team at the National Research Institute for Mathematics and Computer Science.
- Currently, Google plans to upgrade that programming language that isn't clear to the public.

2.3 Features of Python

- **Modularity:** Low-level modules can be easily added to the Python interpreter. This allows programmers to develop modular code.
- **Portability:** Various hardware platforms and interfaces support the Python Language. Hence, code is platform independent and portable across devices.
- **Usage:** Due to its simple and clearly defined syntax structure, Python has a shorting learning curve and quicker troubleshooting time.
- **Clarity:** The elegant syntax structure also makes it easier for users to understand the code even though they may not have programmed it.
- **Support:** Various GUI applications can run and be ported to different operating systems such as Macintosh or Windows. Major commercial databases can also run Python.
- **Scalability:** Compared to shell scripting Python's syntax structure allows for code to be scaled upwards or downwards easily. Large programs are efficiently set up and run.
- **Library Resources:** Python has a vast, portable, and easy to maintain library across all platforms.
- **Interactive usage:** Programmers can utilize features such as dynamic code troubleshooting not present in other programming languages.
- **Databases:** Python provides interfaces to all major commercial databases.

3. General Project Description

3.1 Project Perspective

The purpose of this project is to find patterns in the crime rates for some districts of chicago during the pandemic period and then determine the future crime rates through the LSTM model.

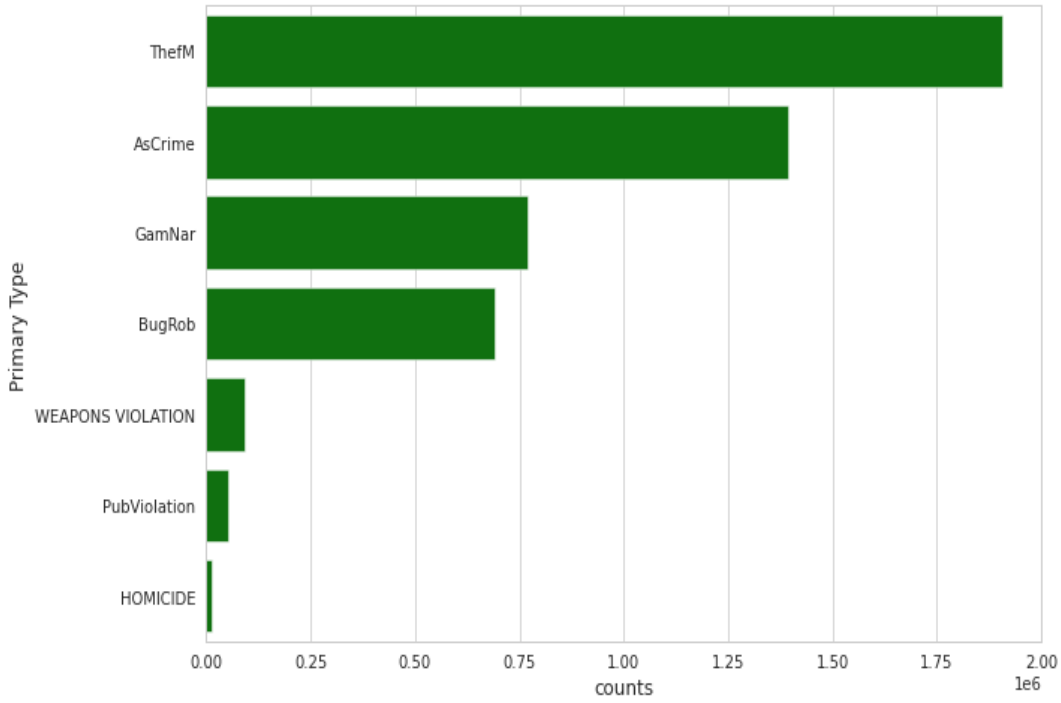
3.2 Project Components

The project heavily used CRISP-DM methodology. This methodology consists of descriptions of typical phases of the project, the task involved with each phase, and an explanation of the relationship between the tasks.

3.3 Specific Goals:

Following images show the trends during different years for the mentioned criminal activity.

- X-axis: represents years.
- Y-axis: represents count of the particular criminal activity.



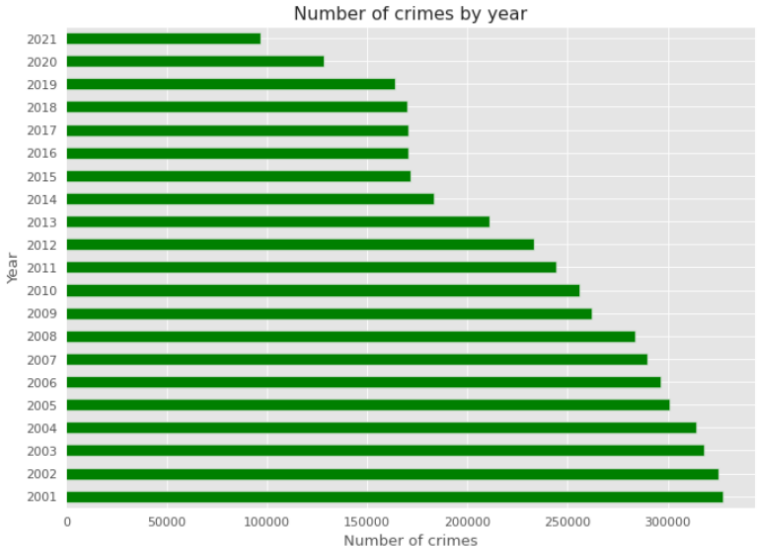
Specific Goal: By the use of the openly available data, we tried to find patterns in these criminal activities (patterns like, is one kind of crime more frequent than other, is one area of Chicago more unsafe than the others, etc.) and came up with models(LSTM) that will help us in predicting crimes.

Here is sample visualization of year wise crime rate plot:

Plotting Data

Year wise crime rate plot:

```
plt.style.use('ggplot')
plt.figure(figsize=(10, 7), dpi=80)
crimes.groupby([crimes.Year]).size().plot(kind='barh', color='green')

plt.ylabel('Year')
plt.xlabel('Number of crimes')
plt.title('Number of crimes by year')
plt.show()
```



3.4 Overview of Programs Related to Specific Goals:

The overall project code is divided into multiple subtasks such as:

1. **Peeking into the data:** This task includes loading the dataset from the google drive to the colab using python's mount() function. Read the chicago crime dataset with the help of python panda's read_csv() function.
2. **Data preparation:** This stage includes verifying the uniformity of the dataset, deleting redundant data etc.
3. **Prediction and modeling:** This stage includes dividing/splitting the dataset into the train and test dataset respectively and applying LSTM model(python library used -keras.layers)
4. **Evaluation:** Evaluating the correctness of the machine learning model used - Root mean square error metric is used.

3.5 Assumptions and Dependencies:

There were no assumptions made for this project. The project heavily relies on data preparation,visualization and python libraries such as numpy, pandas, matplotlib, seaborn, LSTM etc.

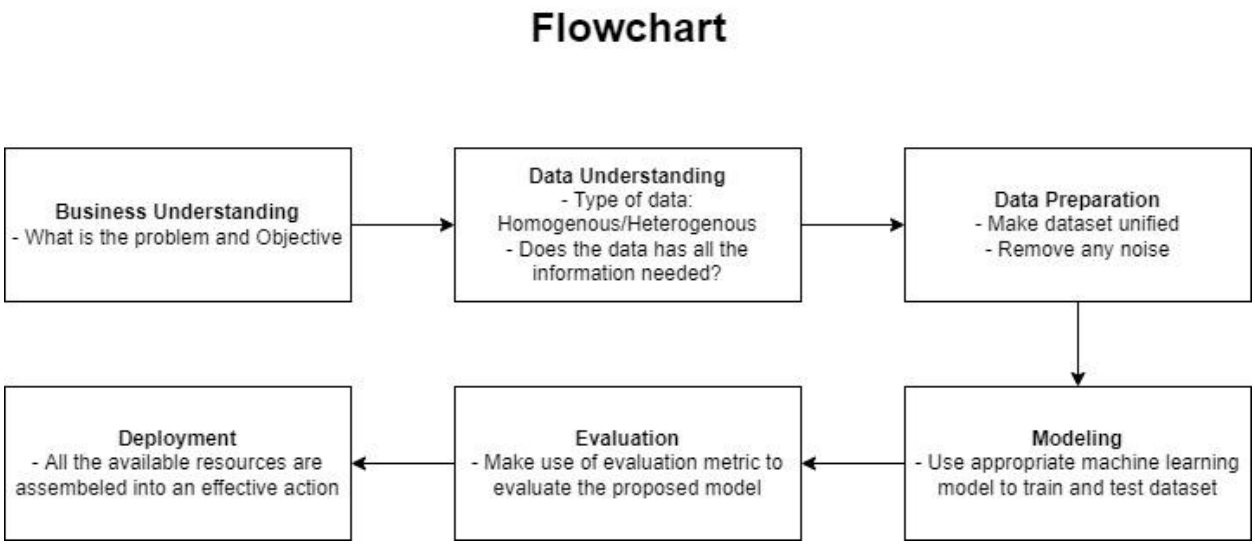
4. Programs Developed

4.1 Program Specifications

4.1.1 User Interfaces:

Our project is aimed to provide visualization and forecasting/prediction of future events so we haven't specifically used any UI for the same.We used python's matplotlib module and its libraries to plot graphs for data visualization.

4.1.2 Flow Chart



4.1.3 Program Logic and Code Explanations

As per CRISP-DM methodology the program logic is developed and divided into 3 main stages-

1. Data Understanding/Exploratory Data Analysis:

This stage involved - dropping the columns or attributes from dataset which are not related,checking for NULL values and then dropping the missing values(methods used- isnull(), dropna()etc.)

Code:

Removing the columns after analysis of their content, which are not useful further.

```
# Drop columns which are not related
crimes = data.drop(columns=['Ward', 'Community Area','Location','Block','Domestic','Beat','Updated On','IUCR','FBI Code','ID','Location Description','
```

```
crimes.head()
```

Date	Case Number	Date	Primary Type	Arrest	District	X Coordinate	Y Coordinate	Year	Latitude	Longitude
2001-01-01	G000070	01/01/2001 12:00:00 AM	CRIMINAL DAMAGE	False	22.0	1165016.0	1832136.0	2001	41.694977	-87.671448
2001-01-01	G177802	01/01/2001 12:00:00 AM	ASSAULT	False	24.0	1168051.0	1941384.0	2001	41.994699	-87.657187
2001-01-01	HK408994	01/01/2001 12:00:00 AM	OFFENSE INVOLVING CHILDREN	False	20.0	1164565.0	1939287.0	2001	41.989020	-87.670069
2001-01-01	HW586150	01/01/2001 12:00:00 AM	CRIM SEXUAL ASSAULT	False	11.0	NaN	NaN	2001	NaN	NaN
2001-01-01	HN727099	01/01/2001 12:00:00 AM	THEFT	False	22.0	1175615.0	1829550.0	2001	41.687651	-87.632719

```
crimes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 7427777 entries, 2001-01-01 00:00:00 to 2021-10-23 23:50:00
Data columns (total 10 columns):
 #   Column        Dtype
---  ------        -----
 0   Case Number   object
 1   Date          object
 2   Primary Type  object
 3   Arrest        bool
 4   District      float64
 5   X Coordinate  float64
 6   Y Coordinate  float64
 7   Year          int64
 8   Latitude      float64
 9   Longitude     float64
dtypes: bool(1), float64(5), int64(1), object(3)
```

Checking for Null Values

Check for null values and drop the missing values.

```
crimes.isnull().sum()
```

```
Case Number         4
Date                0
Primary Type        0
Arrest              0
District           47
X Coordinate    74539
Y Coordinate    74539
Year                0
Latitude        74539
Longitude       74539
dtype: int64
```

```
crimes.dropna()
```

Date	Case Number	Date	Primary Type	Arrest	District	X Coordinate	Y Coordinate	Year	Latitude	Longitude
2001-01-01 00:00:00	G000070	01/01/2001 12:00:00 AM	CRIMINAL DAMAGE	False	22.0	1165016.0	1832136.0	2001	41.694977	-87.671448
2001-01-01 00:00:00	G177802	01/01/2001 12:00:00 AM	ASSAULT	False	24.0	1168051.0	1941384.0	2001	41.994699	-87.657187
2001-01-01 00:00:00	HK408994	01/01/2001 12:00:00 AM	OFFENSE INVOLVING CHILDREN	False	20.0	1164565.0	1939287.0	2001	41.989020	-87.670069
2001-01-01 00:00:00	HN727099	01/01/2001 12:00:00 AM	THEFT	False	22.0	1175615.0	1829550.0	2001	41.687651	-87.632719
2001-01-01 00:00:00	HL265830	01/01/2001 12:00:00 AM	THEFT	False	5.0	1175936.0	1829795.0	2001	41.688316	-87.631536
...	...	...	...	...	...	...	...	...	...	...
2021-10-23 23:30:00	JE421923	10/23/2021 11:30:00 PM	BATTERY	False	9.0	1170500.0	1880881.0	2021	41.828622	-87.649951
2021-10-23 23:40:00	JE423502	10/23/2021 11:40:00 PM	LIQUOR LAW VIOLATION	True	22.0	1167266.0	1828441.0	2021	41.684790	-87.663315
2021-10-23 23:41:00	JE420327	10/23/2021 11:41:00 PM	BATTERY	False	19.0	1165254.0	1916459.0	2021	41.926364	-87.668187
2021-10-23 23:45:00	JE420326	10/23/2021 11:45:00 PM	BATTERY	True	14.0	1154533.0	1908136.0	2021	41.903747	-87.707805
2021-10-23 23:50:00	JE424399	10/23/2021 11:50:00 PM	BATTERY	False	24.0	1159451.0	1945538.0	2021	42.006280	-87.688706

7353188 rows × 10 columns

2. Data Processing:

There are 36 criminal types mentioned in the dataset. Upon further analysis and study of various criminal activities, we could further narrow down most of the activities under specific categories. Some criminal types had multiple labels for the same type (e.g. Crim Sexual Assault, Criminal Sexual Assault). We have excluded certain criminal activities which had a very negligible count of data and could not be classified down to a particular class of criminal activity.

Following are the codes assigned with the criminal activities they include :

- BugRob: includes types Burglary and Robbery.
- AsCrime: includes types Assault, Crim Sexual Assault, Criminal Sexual Assault, Sex Offence, Offence Involving Children, Battery.
- PubViolation: includes types Public Peace Violation, Interference with the officer.
- ThefM: includes types Theft, Motor Vehicle Theft
- Gambar : includes types Gambling, Narcotics, Other Narcotics, Liquor law violation
- Homicide: includes only Homicide crime type.

We will be renaming the Primary types of criminal activities to the above codes that fall under the particular code assigned for the set of those similar criminal activities.

Code:

```
crimes['Primary Type'].replace(['BURGLARY','ROBBERY'],'BugRob',inplace=True)
```

```
crimes['Primary Type'].replace(['CRIMINAL SEXUAL ASSAULT','CRIM SEXUAL ASSAULT','SEX OFFENCE','OFFENCE INVOLVING CHILDREN','BATTERY'],'AsCrime',inplac
```

```
crimes['Primary Type'].replace(['PUBLIC PEACE VIOLATION','INTERFERENCE WITH OFFICER'],'PubViolation',inplace=True)
```

```
crimes['Primary Type'].replace(['THEFT','MOTOR VEHICLE THEFT'],'ThefM',inplace=True)
```

```
crimes['Primary Type'].replace(['GAMBLING','OTHER NARCOTIC VIOLATION','NARCOTICS','LIQUOR LAW VIOLATION'],'GamNar',inplace=True)
```

```
crimes = crimes[crimes['Primary Type'].isin(['AsCrime', 'ThefM', 'GamNar','BugRob','WEAPONS VIOLATION','PubViolation','HOMICIDE'])]
```

```
crimes['Primary Type'].unique()
```

```
array(['AsCrime', 'ThefM', 'BugRob', 'WEAPONS VIOLATION', 'GamNar',
       'PubViolation', 'HOMICIDE'], dtype=object)
```

```
c = crimes[['District','Primary Type']]
x = c.groupby(['District']).count()
d = x.sort_values('Primary Type',ascending=False)
print(d)
dist_list = list(d.index[:2])
```

```
          Primary Type
District
11.0            349731
8.0             316355
7.0             295451
6.0             287255
25.0            277334
4.0             267696
```

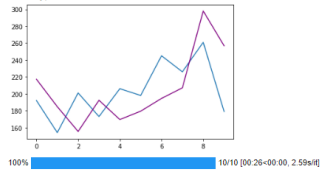
3. Modeling and Evaluation:

In this part we have divided the dataset into a train and test set to verify the efficiency of the LSTM model for prediction.Here is an example of 1-layer LSTM model for district1.

```
[ ]  ## Comparing the actual and the predicted values using our LSTM model for District 1

     # prepare situation
     window = 3
     predTot = list()
     testTot = list()

     # walk forward over time steps in test
     print('District 1')
     for distNum in tqdm_notebook(range(len(train_d_crimes))):

         history = train_d_crimes[distNum]
         test = test_d_crimes[distNum]
         preds = []
         for t in tqdm_notebook(range(len(test)), leave=False):

             length = len(history)

             # split into samples
             X, y = split_sequence(history, window)

             # reshape from [samples, timesteps] into [samples, timesteps, features]
             n_features = 1
             X = X.reshape((X.shape[0], X.shape[1], n_features))

             # define model
             model = Sequential()
             model.add(LSTM(50, activation='relu', input_shape=(window, n_features)))
             model.add(Dense(1))
             model.compile(optimizer='adam', loss='mse')

             # fit model
             model.fit(X, y, epochs=200, verbose=0)

             X_test = array([history[i] for i in range(length-window, length)])
             X_test = X_test.reshape((1, window, n_features))
             yhat = model.predict(X_test, verbose=0)

             obs = test[t]
             preds.append(yhat.reshape((1,)))
             history.append(obs)

         # plot
         print('Crime Type: {}'.format(crime_type[distNum]))
         plt.plot(test)
         plt.plot(preds, color='purple')
         plt.show()

         testTot = testTot + test
```

```
        model.add(Dense(1))
        model.compile(optimizer='adam', loss='mse')

        # fit model
        model.fit(X, y, epochs=200, verbose=0)

        X_test = array([history[i] for i in range(length-window, length)])
        X_test = X_test.reshape((1, window, n_features))
        yhat = model.predict(X_test, verbose=0)

        obs = test[t]
        preds.append(yhat.reshape((1,)))
        history.append(obs)

    # plot
    print('Crime Type: {}'.format(crime_type[distNum]))
    plt.plot(test)
    plt.plot(preds, color='purple')
    plt.show()

    testTot = testTot + test
    predTot = predTot + preds
error = mean_squared_error(predTot, testTot) ** .5
print('Test RMSE: %.3f' % error)
```

```
District 1
100%|██████████| 7/7 [03:17<00:00, 28.01s/it]
100%|██████████| 10/10 [00:28<00:00, 2.76s/it]
WARNING:tensorflow:5 out of the last 5 calls to <function Model.make_predict_function.<locals>.predict_function at 0x7fcd730444d0> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.f
WARNING:tensorflow:6 out of the last 6 calls to <function Model.make_predict_function.<locals>.predict_function at 0x7fcd724b2cb0> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.f
Crime Type: TheftM
```




```
100%|██████████| 10/10 [00:26<00:00, 2.59s/it]
```

Root mean Square Error(RMSE):

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

RMSE equation, source: Google Images

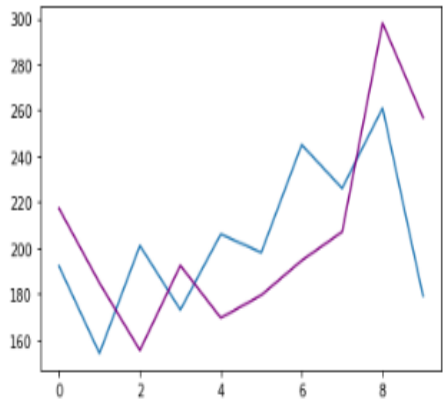
Here yi(hat) is the prediction and y(i) is the actual. 'n' is the total number of predictions which is 12 here in our case (one for each month). RMSE tells us how *off* are the predictions from actual values therefore lower the RMSE the better. Now we've also defined a way to compare our predictions

For district1 Test RMSE score obtained is - 23.620

4.1.4 Inputs/Outputs:

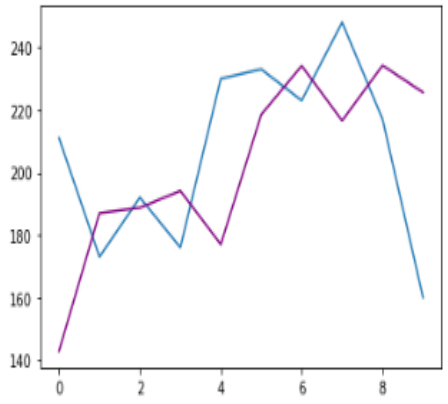
Input : Chicago crime dataset

Output: Visualization graphs of LSTM for district1 and district2



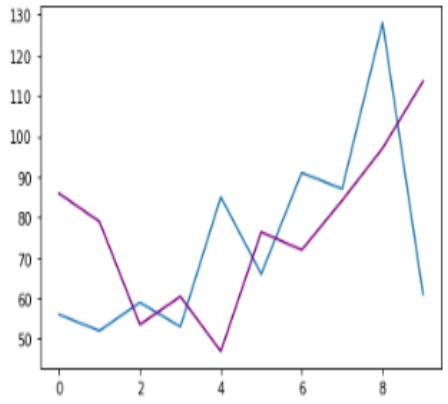
100% 10/10 [00:26<00:00, 2.59s/it]

Crime Type: AsCrime

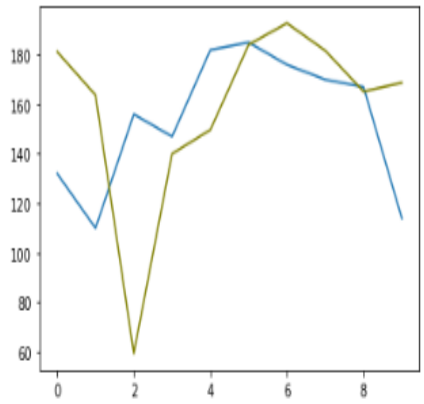


100% 10/10 [00:27<00:00, 2.84s/it]

Crime Type: BugRob

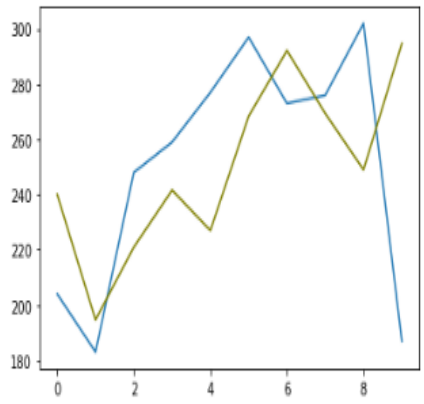


100% 10/10 [00:32<00:00, 3.81s/it]



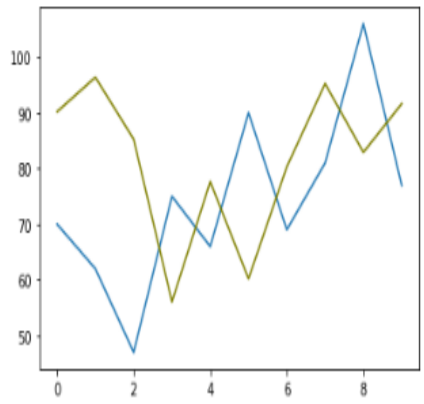
100% 10/10 [00:27<00:00, 2.65s/it]

Crime Type: AsCrime



100% 10/10 [00:26<00:00, 2.46s/it]

Crime Type: BugRob



100% 10/10 [00:28<00:00, 2.89s/it]

4.2 Clarity of Project

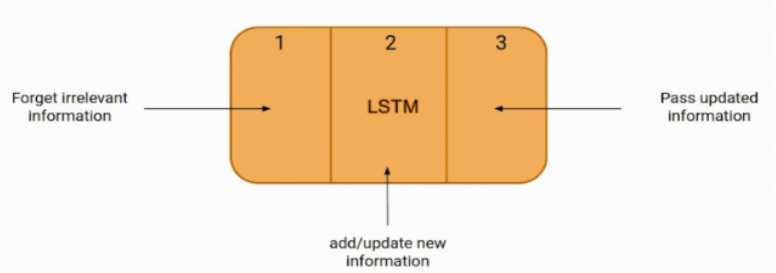
4.2.1 Coding Quality:

The code we developed is of high quality as it meets all the requirements. The code follows some non-functional attributes such as clarity, maintainability, refactoring, reusability, and well-tested efficiency.

4.2.2 Algorithms: used LSTM

Long Short Term Memory Network is an advanced RNN, also known as a sequential network, that allows information to be stored. It can solve the vanishing gradient problem that RNN has. A recurrent neural network, also known as an RNN, is a type of persistent memory system[1].

Architecture of LSTM:

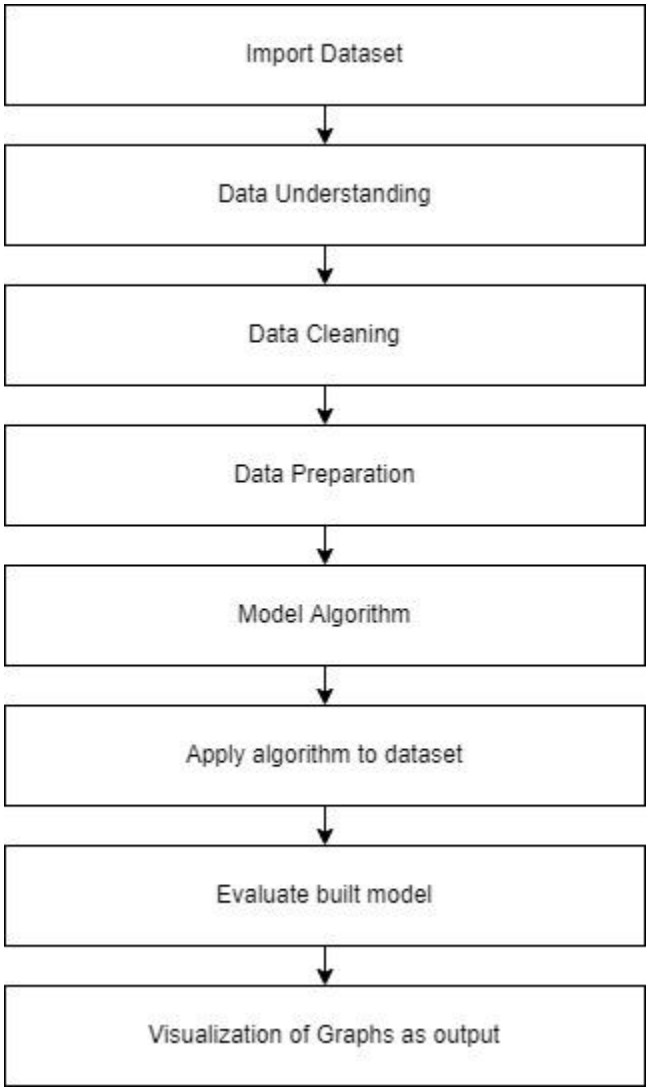


Source: Reference[1]

4.2.3 Performance

We have used LSTM which gives RMSE score of 36.367 for district2 and 23.620 for district 1.This algorithm gives best performance compared to Linear regression(423.102) and XG Boost(100.763).

4.3 Project Workflow



5. Tools & Environment:

- Google colaboratory was used to develop and deploy the project.
- Github as a version control tool.
- Python Libraries such as pandas,matplotlib, numpy,seaborn etc.

6. Discussions and conclusion

6.1 Conclusion

In particular, we have created several visualizations to dig deeper into the dataset to find various correlations between the features and the target variable. The unique feature in our data is that it doesn't contain any NULL values. The columns chosen for processing have no null values. Hence, there is no need for imputation or handling of null values. We used the Date feature to derive other attributes like 'day' of the activity, 'month' of the activity and to plot corresponding graphs. Also, we removed rows where the year in 'Date' does not match the year in the 'Year' column. Plus, any duplicates were removed based on ID and Case Number pair. Finally, the Root Mean Square Error(RMSE) method was used for error measure.

6.2 References

6.2.1 Dataset Reference

Link for the dataset: Chicago Crime Dataset

(https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-Present/ijzp-q8t2/data)

This dataset reflects reported incidents of crime that occurred in the City of Chicago from 2001 to the present. This dataset gets updated on a weekly basis. Data is extracted from the Chicago Police Department's CLEAR (Citizen Law Enforcement Analysis and Reporting) system.

Other references:

[1]https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/
[2]https://www.sciencedirect.com/science/article/pii/S2666449622000135
[3]https://machinelearningmastery.com/time-series-forecasting-with-prophet-in-python/
[4]http://reach4ml.org/using-data-science-along-with-machine-learning-to-predict-the-future-theft-crime-rates-in-chicago/

6.2.2 Reference Document

Below are the list of documents attached along with this report:

1. **SPL_Project.zip** – This file contains the Source code.
2. **README.txt** – This file contains the instructions to run the code.

6.2.3 Github links

- https://github.com/ruchavisal/Impact-of-Covid-19-Pandemic-on-Criminal-Activities