



JOMO KENYATTA UNIVERSITY OF AGRICULTURE AND TECHNOLOGY

COLLEGE OF PURE AND APPLIED SCIENCES

SCHOOL OF COMPUTING AND INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTING

BSc. COMPUTER SCIENCE

ICS2402: COMPUTER SYSTEMS PROJECT

REF: JKU/2/83/022

**Title: PREDICTIVE ANALYSIS OF CRIME INCIDENCES BASED ON
SOCIO-ECONOMIC FACTORS IN KENYA**

Written By: Odhiambo Paul Erick
SCT211-0094/2016

Supervisor: Dr. Lawrence Nderu

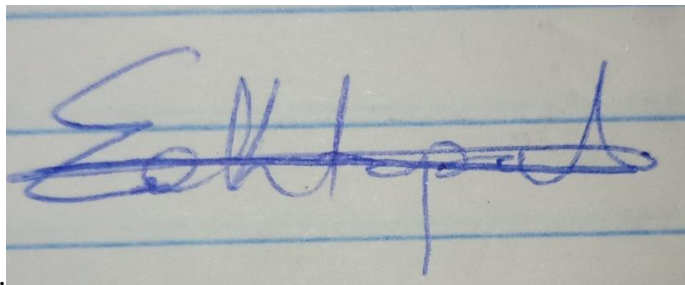
Supervisor 2: Ms. Joan Gichuru

DECLARATIONS

I declare that this work presented to Jomo Kenyatta University of Agriculture and Technology as a requirement for the award of bachelor's degree in Computer Science, is my original work that has not been presented to any institution for an award or any attainment whatsoever, with duly acknowledged material cited and referenced.

NAME: **ODHIAMBO PAUL ERICK**

REG NO: **SCT211-0094/2016**



SIGNATURE:

DATE: **10TH JANUARY 2021**

SUPERVISORS' APPROVAL

The project presented has been supervised and approved by the School of Computing and Information Technology, Department of Computing, through our supervision and confirmation of the work.

1ST SUPERVISOR

NAME: **PROF. LAWRENCE NDERU**

SIGNATURE:

DATE: **10TH JANUARY 2021**

2ND SUPERVISOR

NAME: **MS. JOAN GICHURU**

SIGNATURE:

DATE: **10TH JANUARY 2021**

ACKNOWLEDGEMENT

I thank God for His protection and provision of good health and wisdom to see me through the completion of this project successfully. I equally appreciate the insights and information received from the school throughout the course and my supervisors for their guidance to relevant resources I obtained from scholarly sites. I extend my appreciation to my family and friends who have been very instrumental in believing in me and supporting me with the best environment to see the success of this project.

ABSTRACT

This project aims to counter crime by developing a scientific model that will help in predicting crime incidence or simply the number of crimes for different types of crime that will happen soon in Kenya, based on proposed variables that are believed to directly influence various criminal activities. These variables are age group, employment, economy, education, and gender. With the lack of a detailed dataset that is a backbone for this work, Economic growth which is measured through Gross Domestic Product, and directly influences most criminal activities, has been used as the main metric to offer predictive analysis of crime.

The research area of this project is motivated by “Goal 16 - Peace, Justice, and Strong Institutions“ which is one of the fundamental Sustainable Development Goals vastly known as SDGs that have been stipulated by the United Nations, and Kenya being one of the member states of United Nations, champions this course.

Table of Contents

CHAPTER 1: INTRODUCTION	8
1.0 BACKGROUND INFORMATION	8
1.1 PROBLEM STATEMENT	8
1.2 JUSTIFICATION	9
1.3 OBJECTIVES	9
1.3.1 Goal/Main Objectives	9
1.3.2 Sub-Objectives	9
CHAPTER 2: LITERATURE REVIEW	10
CHAPTER 3: RESEARCH METHODOLOGY	11
CHAPTER 4: SYSTEMS ANALYSIS AND DESIGN	12
4.1 Introduction	12
4.2 Input Requirements	12
4.2.1 Source of Data	12
4.2.2 Functional Requirements	12
4.2.3 Non-Functional Requirements	12
4.3 Output Requirements	12
4.3.1 Functional Requirements	12
4.3.2 Non-Functional Requirement	12
4.4 Processing Requirements	13
4.5 Hardware Requirements	13
4.6 Software Requirements	13
4.7 Design Methodology	13
4.7.1 High-Level Design	14
CHAPTER 5: SYSTEM IMPLEMENTATION	16
5.1 Business Understanding	16
5.2 Data Collection	16
5.2.1 Mining the first data set and saving it as a CSV file	17
5.2.2 Mining second data set and saving it as a CSV file	18
5.3 Data Preparation	18
5.3.1 Working on the first dataset with crime records from 2005 up to 2012	19
5.3.2 Working on the second dataset with crime records from 2010 up to 2019	23
5.3.3 Merging the two datasets into a single dataset	28

5.3.4 Working on the merged dataset -- Crime records from 2005 up to 2019 – to create a new dataset with types of criminal records and corresponding yearly GDP values.	31
5.3.5 Merging the single yearly datasets to create a new dataset with yearly GDPs.....	48
5.4 Exploratory Data Analysis	52
5.4.1 Loading the data set, reading it as data frame, and indexing the 'Year' column.....	52
5.4.2 Studying the trend in the Gross_Domestic_Product_GDP	53
5.4.3 Studying the trend in the Number_of_Crimes_Reported	53
5.5 Modeling	54
5.5.1 Data Preparation for LSTM	54
5.5.2 One-Hot Encoding.....	57
5.5.3 Normalizing the attributes.....	59
5.5.4 Framing the problem into a Supervised Learning problem	60
5.5.5 Removing the columns not to be predicted	61
5.5.6 Splitting the data set into Train and Test set.....	61
5.5.7 Design and Fitting the Neural Network Model.....	62
5.5.8 Plotting the Loss Chart	64
5.6 Model Evaluation	65
5.6.1 Evaluating the model using Root Mean Squared Error (RMSE).....	67
5.6.2 Plotting Actual Values against Forecast values.....	68
5.7 Model Deployment	68
CHAPTER 6: CONCLUSION, CHALLENGES, AND RECOMMENDATIONS.....	72
6.1 Conclusion.....	72
6.2 Challenges faced during the Project	72
6.3 Project Recommendations.....	72
CHAPTER 7: REFERENCES	73

CHAPTER 1: INTRODUCTION

1.0 BACKGROUND INFORMATION

Introduced in the year 1945, The United Nations was formed to mainly ensure that peace and security are maintained across the globe through cohesive collaboration and interrelation amongst its member states. By 2017, the United Nations made an affirmation that 68.5 million people had been violently displaced due to violence, persecution, conflict, or violation of human rights across the globe. The organization's research also goes ahead to conclude that high corruption, theft, tax evasion, and a weak judicial system costs developing countries US\$1.26 trillion per year. This is a big drawback in the economy to the developing countries and Kenya as an example has witnessed a series of economic crises over the past decades.

In a country where economic crises have been looming, criminal activities are guaranteed to be reported from time to time. Some of the major types of criminal activities that have indeed been witnessed in our country are corruption, robbery, and theft by servants. Continuous occurrence of these criminal activities leads to the continued suppression of development in the different sectors in our country like health, education, and even the judicial systems. This in turn leads to inequality amongst people in gaining access to these resources and afford a decent living. In the long run, our societies and the country at large grows with little job opportunities in which employees averagely earn little wages. An example of the effects of this is rendering a lot of families poor, and a purported 'majority of Kenyans' being in the middle class, yearly boycotting, holding strikes and demonstrations to fight for payment of wages or promised salary increase.

It is also sad to note that several economic crimes lead to social crimes. Cases such as suicide and other offenses against morality such as prostitution have all been witnessed in our country. Crime has been a major concern in our country and economic growth is a key concern to be kept under close supervision to ensure that crime is curbed if not done away with.

1.1 PROBLEM STATEMENT

The number of criminal activities in Kenya has been increasing over years. According to the 2018 Annual Crime Report made by the National Police Service, there were 88,268 reported cases to the police as compared to 77,992 in 2017. This was an increase of 10,276 (13%) cases. In 2016 the reported cases were 76,986 which grew to 77,992 in 2017, interpreting an increase of 1,006 (1%).

The incidence that has had an increase include Assault, Creating Disturbance, Defilement, General Stealing, Malicious Damage, and Possession of drugs mainly Cannabis Sativa, bhang. Amongst the major attempts that have been made by the relevant stakeholders within the government to help solve this problem include reorganizing the National Police Service but this has not yet made any notable effect.

1.2 JUSTIFICATION

Crime is one of the problems affecting our country Kenya. I settled for this research topic since it is in line with the Sustainable Development Goals (SDGs), and more specifically “Goal 16 - Peace, Justice and Strong Institutions“. These goals have been stipulated by United Nations, after being discussed and agreed by the United Nations member states, and Kenya is one of the member states that is on an everyday hunt to achieve economic stability, fight corruption, and other economic and social crimes as a whole, to guarantee a safe environment for productivity and living for her citizens.

1.3 OBJECTIVES

1.3.1 Goal/Main Objectives

- The objective of this project is to build a predictive model that forecasts the number of different types of crime and provide relevant bodies fighting crime with analytical insights that will enable them to take informed actions, thereby promoting Peace, Justice, and Strong Institutions – SDG number 16.

1.3.2 Sub-Objectives

- Establish correlation between the various metrics of crime and therefore make a more accurate prediction on the incidence of crime
- Use predictive analytics to model the problem
- Provide actionable insights to relevant stakeholders so that correctional measures can be undertaken

CHAPTER 2: LITERATURE REVIEW

This study has been achieved through vast research on the purpose topic. To start with, the United Nations and its member states have set aside the Sustainable Development Goals (SDGs) which are 17 in total to be achieved in all her member states to see to it that poverty comes to an end, safeguard the planet, and to guarantee that by the year 2030, everyone in the planet lives in a peaceful and prosperous world[1]. One of these goals is ‘Goal 16 - Peace, Justice and Strong Institutions’ which is purposed in this project [2].

Crime is described as any action against the law that is carried out purposefully by a person or persons. Its results are both physical torture and psychological distress including the damage of property and loss of property as a whole[3]. In a move to combat crime, the United Nations advises that organizational crimes and drug-related crimes have been on the increase. This leads to the failure of many businesses in most sectors hindering progressive economic growth in many regions of the world[4]. Violent crimes have also proven to threaten many people’s security in places across many countries in Africa. This forces the affected people to relocate to safer or better places.

Notably, most people in Kenya do not report cases of criminal activities to the police. This is a result of the citizens’ lack of trust in the police to deliver justice. The police have in turn been depicted as being incapacitated to perform their duties accordingly. This is attributed to their lack of resources to carry out investigations to completion and corruption cases within the authorities[5]. This has seen a trend in the growth of the private security sector. Sadly, their service is only limited to the few citizens who are capable of affording such.

In the capital city, Nairobi, for example, organized crimes that are mostly gang-related have been on increase from time immemorial[6]. Cases of mugging, rape, money laundering, drug and alcohol abuse, robbery, and shoplifting have been witnessed in many places around the city. The main cause of such criminal acts is related to unemployment, overpopulation, and peer influence amongst the offenders[4]. Unemployment in most cases has been the cause of most crimes if not all. The lack of jobs for many in the city has seen the desperate move of resulting into crime for survival means.

The majority of an unemployed group of people and most crime offenders in the country are the youth. The International Labour Organization (ILO) lists the main causes of unemployment to be increased population to match the limited number of job opportunities, unskilled population, and slow economic growth[7]. Following the economic theory, as the economy grows and opportunities expand, crime should decrease. This implies that crime will increase as the economy drops and the opportunities reduce in number[8].

The literature exploration and evidence backs up the hypotheses for this study which is

Hypothesis: Crime and economic growth are correlated implying that an increase in economic growth results in a decrease in the number of crimes and vice versa.

CHAPTER 3: RESEARCH METHODOLOGY

Qualitative and quantitative research methods are the two types of research. Quantitative methods are scientific methodologies that collect quantitative data and analyze it statistically, producing results from a population sample of interest. Qualitative approaches, on the other hand, are interpretive and are concerned with uncovering the underlying causes of a phenomenon as well as how people perceive the world around them. This research relies heavily on quantitative methodologies, which entail evaluating numerical and statistical data to determine causal linkages and anticipate outcomes.

I used the secondary data gathering and analysis approach in quantitative analysis, which comprises evaluating existing data and quantitatively manipulating the statistical data. In this light, the majority of the data included in this study is official statistics collected from the Kenya National Bureau of Statistics published annually. The consistency and correctness of official statistics, which allows for valuable comparisons over time, affected the choice of this strategy. Furthermore, secondary data could serve as a foundation for additional investigation and analysis.

The Long Short-Term Memory algorithm was used in this investigation. Working with Recurrent Neural Networks, this approach is useful for time-series predictions. Recurrent Neural Networks (RNNs) are memory-gated neural networks that are well-suited to supervised learning issues with time-series data[9]. The RNNs memory allows them to remember patterns, which is important in problems involving sequence dependency, such as prediction.

The LSTM algorithm features a three-gate design that manages the memory contents. The input gate as well as the forget gate are in charge of the cell state and is the long-term memory that is employed for sequence dependence[10]. The output gate generates the hidden state, or output vector, which would be the memory centered to be used. I chose LSTM-RNN because it models the supervised machine learning problem from beginning to end[11]. Second, the approach makes it simple to include metrics whose values are generated outside of the model. Finally, the technique does autonomous feature extraction, which aids in the predictive modeling feature engineering process[12]. The Root Mean Square Error is used to evaluate the model's accuracy in prediction (RMSE).

CHAPTER 4: SYSTEMS ANALYSIS AND DESIGN

4.1 Introduction

This section entails all the analytical aspects that led to the design and implementation of the desired predictive model. The processing requirements, input, and output, software, and hardware requirements were explored too. The design methodology of the systems is also captured in diagrams.

4.2 Input Requirements

4.2.1 Source of Data

The data that has been used throughout this project has been mined from the Yearly Statistical Abstracts that are provided by the Kenya National Bureau of Statistics.

4.2.2 Functional Requirements

- i) Make use of the Camelot-py module to draw out tables from the Statistical Abstracts files and export these tables into CSV files.
- ii) Data Cleansing.
- iii) One-hot encoding the Type of Crime column.

4.2.3 Non-Functional Requirements

- i) The correctness of the data.
- ii) Consistency of the Data Format – The time series data.
- iii) The integrity of the data.

4.3 Output Requirements

4.3.1 Functional Requirements

- i) The system developed should give a prediction of the number of crimes that will occur for the different types of crime.
- ii) A display of the number of different types of crime should be made by the system.

4.3.2 Non-Functional Requirement

The main requirement here is achieved through reproducibility by providing the coding documentation and providing analysis reports resulting in a documented system.

4.4 Processing Requirements

- i) Preprocessing the input data.
- ii) Perform an Exploratory Data Analysis.
- iii) Training the Recurrent Neural Network model using the Long Short-Term Memory algorithm.
- iv) The model created should be able to collect data, both while building the model and during its deployment to give predictions.
- v) Model deployment to give predictions on an interface - web.

4.5 Hardware Requirements

- i) CPU
- ii) RAM
- iii) GPU

4.6 Software Requirements

- i) Working Environment – Google Colab
- ii) Programming language – Python
- iii) Data collection – Camelot.py
- iv) Data preparation – Pandas
- v) Model deployment – Flask

4.7 Design Methodology

The design methodology for this project is the Agile methodology. I chose this methodology because of its faster implementation of ideas and quick identification of errors while developing. The errors are equally quick to eliminate and continue with production.

Another advantage of this methodology that led me to choose it is managing iterations during development and this improves the productivity of the work.



Fig A. Agile Methodology

4.7.1 High-Level Design

The project is also governed by the data science project life cycle making the high-level design of the system to be that of a data science project. This is shown below

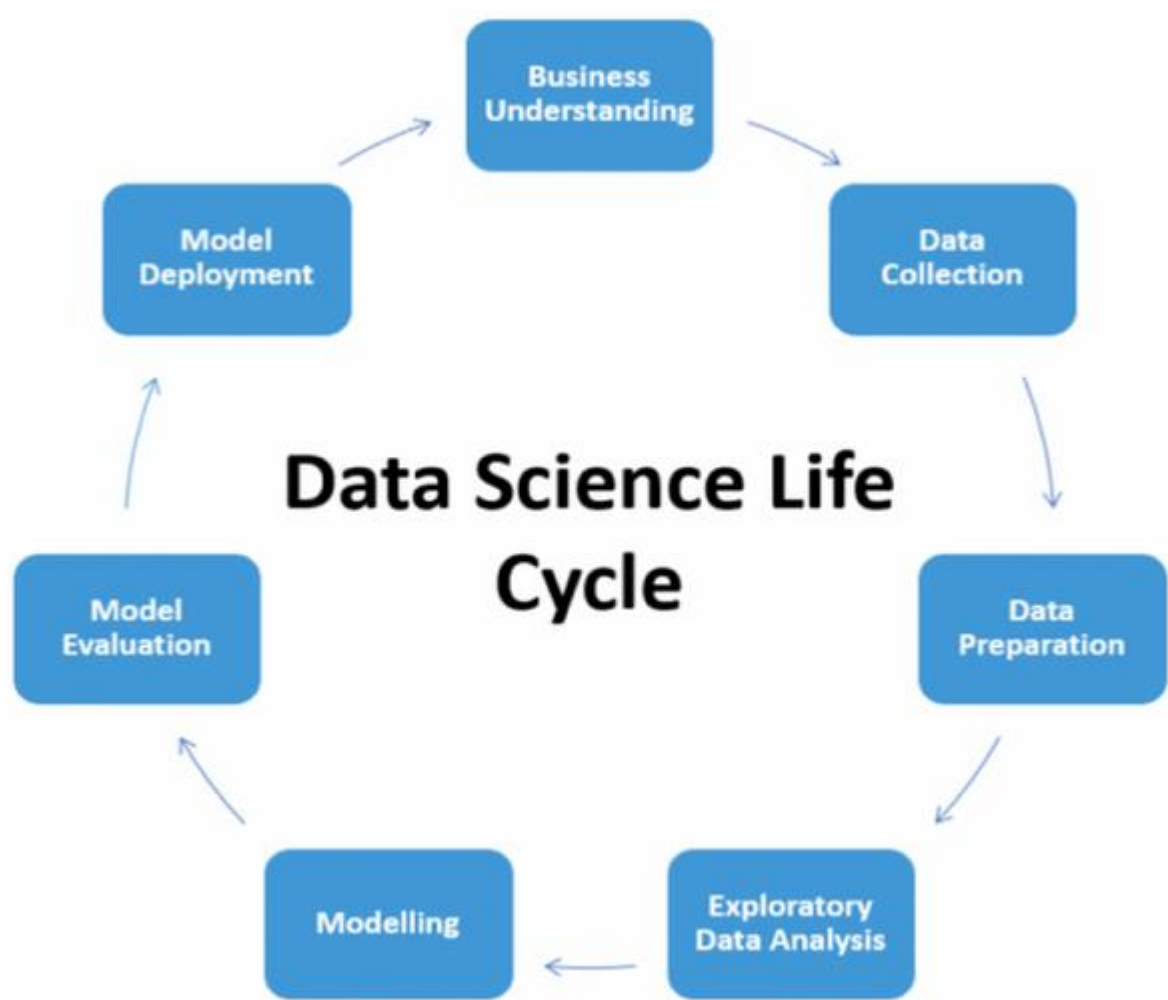


Fig B. Data Science Project Life Cycle

CHAPTER 5: SYSTEM IMPLEMENTATION

This implementation of this project's system was achieved through the adherence of the Data Science Life Cycle which involves:

- i) Business Understanding
- ii) Data collection
- iii) Data preparation
- iv) Exploratory Data Analysis
- v) Modeling
- vi) Model Evaluation
- vii) Model Deployment

5.1 Business Understanding

There are many theories that surround crime. For example, when there is an increase in Economic growth more money is believed to be generated and so crime cases are likely to go high. Examples of these crimes are theft, corruption e.t.c. When there is a drop in economic growth, less amount of money is generated and cases of crimes are likely to increase.

Ideally, when the economic growth is stable or increasing, less crime should be recorded and when economic growth is dropping, more crimes are expected. This led to the data collection phase.

5.2 Data Collection

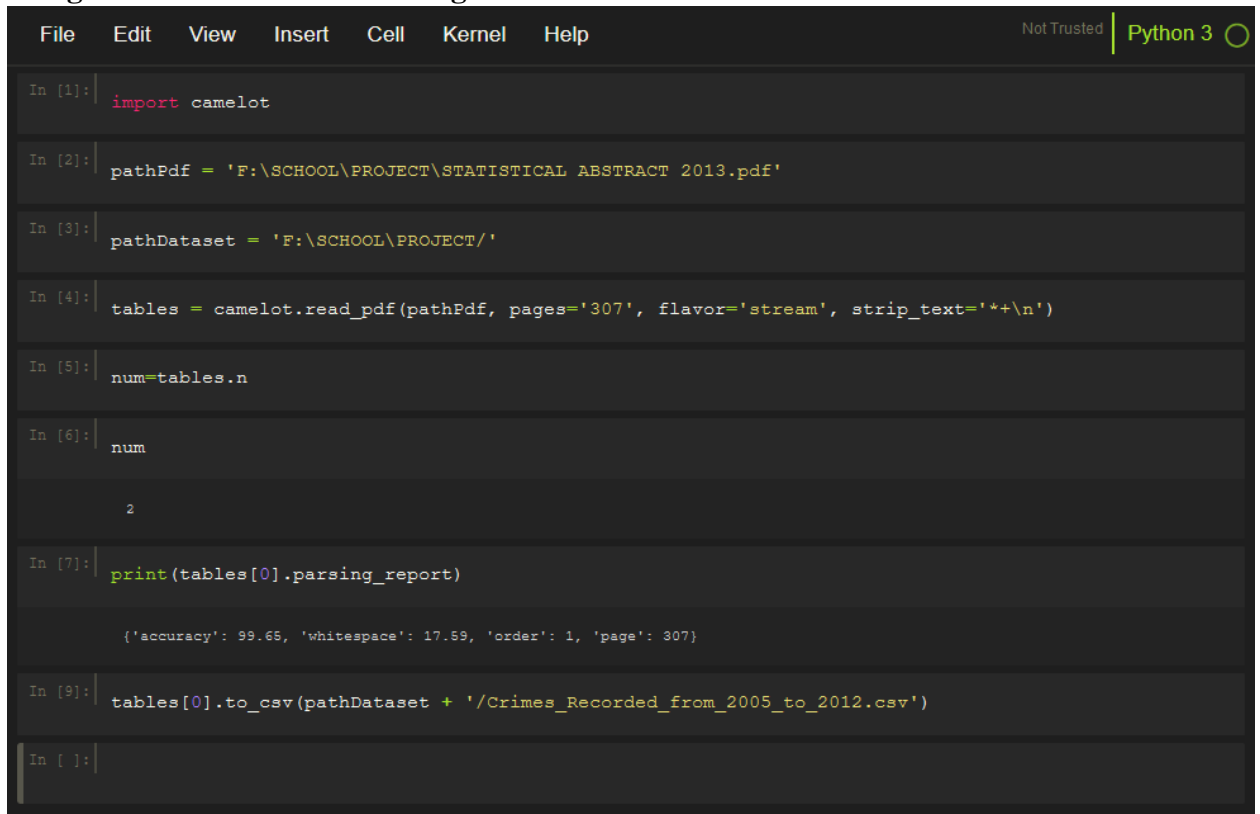
At this phase, I was looking for datasets that had different types of crime with variables that are believed to affect crime like gender, age group, level of education, and employment or economy. Getting such a dataset with all listed variables was a challenge.

I proceeded to mine datasets from the Yearly Statistical Abstracts by the Kenya National Bureau of Statistics and create a dataset to be used in this project. The data sets were:

1. STATISTICAL ABSTRACT 2013.pdf which had crime records from 2005 to 2012.
2. Statistical Abstract 2020.pdf which had crime records from 2010 to 2019.

I mined the datasets locally using Anaconda's Jupyter Notebook.

5.2.1 Mining the first data set and saving it as a CSV file



```
File Edit View Insert Cell Kernel Help Not Trusted Python 3 ○

In [1]: import camelot

In [2]: pathPdf = 'F:\SCHOOL\PROJECT\STATISTICAL ABSTRACT 2013.pdf'

In [3]: pathDataset = 'F:\SCHOOL\PROJECT/'

In [4]: tables = camelot.read_pdf(pathPdf, pages='307', flavor='stream', strip_text='*+\n')

In [5]: num=tables.n

In [6]: num

2

In [7]: print(tables[0].parsing_report)

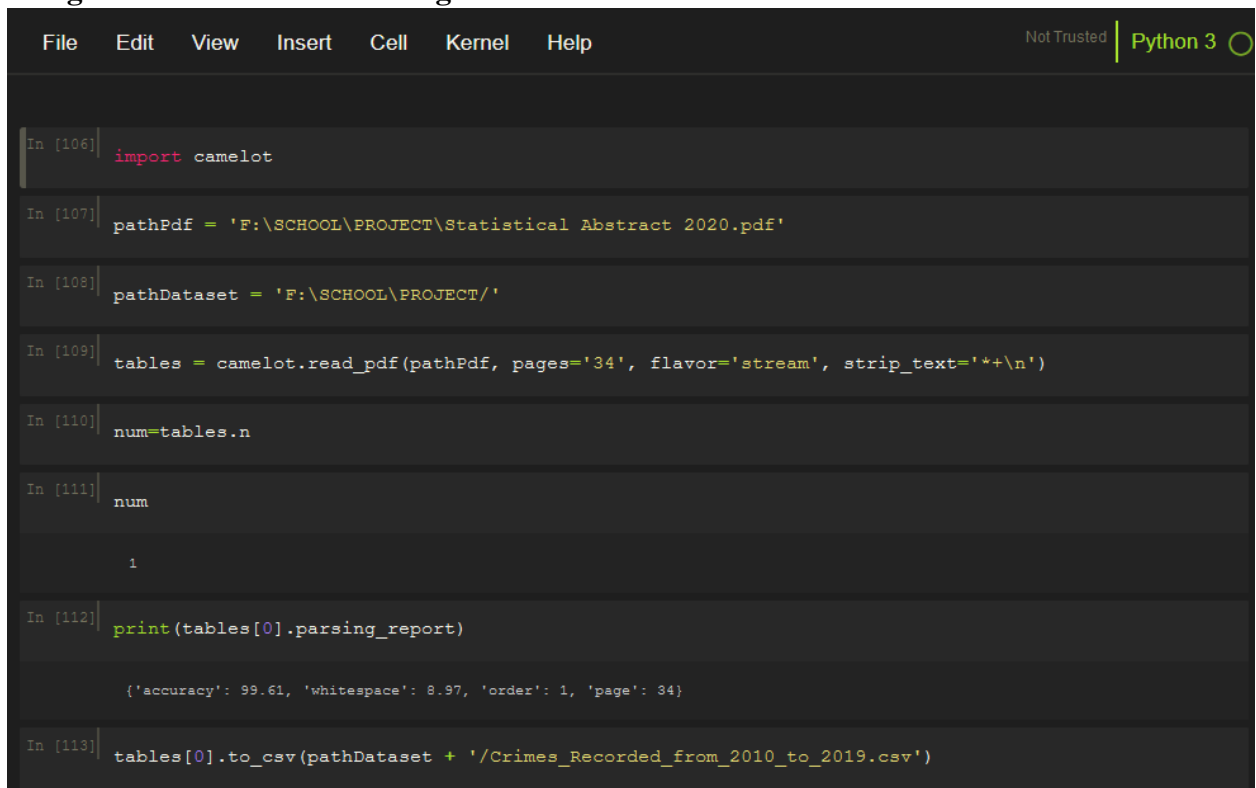
{'accuracy': 99.65, 'whitespace': 17.59, 'order': 1, 'page': 307}

In [9]: tables[0].to_csv(pathDataset + '/Crimes_Recorded_from_2005_to_2012.csv')

In [ ]:
```

Fig 1. Mining data set with crime records from 2005 to 2012

5.2.2 Mining second data set and saving it as a CSV file



```
File Edit View Insert Cell Kernel Help Not Trusted Python 3

In [106]: import camelot

In [107]: pathPdf = 'F:\SCHOOL\PROJECT\Statistical Abstract 2020.pdf'

In [108]: pathDataset = 'F:\SCHOOL\PROJECT/'

In [109]: tables = camelot.read_pdf(pathPdf, pages='34', flavor='stream', strip_text='**+\n')

In [110]: num=tables.n

In [111]: num

1

In [112]: print(tables[0].parsing_report)

{'accuracy': 99.61, 'whitespace': 8.97, 'order': 1, 'page': 34}

In [113]: tables[0].to_csv(pathDataset + '/Crimes_Recorded_from_2010_to_2019.csv')
```

Fig 2. Mining data set with crime records from 2010 to 2019

The two datasets did not have variables affecting crime. This led to the data preparation phase where I added a new column of Gross Domestic Product (GDP) since the Economy is one of the crime metrics and GDP is the best measure of Economic growth.

5.3 Data Preparation

In this phase, I was preparing the datasets that I had mined to form more meaningful data. This involved dropping columns and rows that were unnecessary from each dataset, renaming some columns, adding new columns, and finally merging the two datasets at the year column.

From this phase onwards, I carried out my project on Google Colaboratory, vastly known as Google Colab, since it is opportune for data analysis and machine learning tasks. I first moved the two CSV datasets that I had mined locally, to my drive, then mounted the drive on Google Colab to enable me to access the files online.

The first import I did on the Google Colab environment was importing Tensorflow, which is an open-source AI library that enables machine learning to be practiced largely and with ease, alongside training models and also performing numerical

computations on presented data. I then installed Python, the programming language I used for the data-related task.

```

• Working environment - Google Colab

[ ] import tensorflow as tf
    print(tf.__version__)

2.7.0

[ ] !python -c "import sys; print(sys.version)"

3.7.12 (default, Sep 10 2021, 00:21:48)
[GCC 7.5.0]

```

Fig 3. Importing Tensorflow and setting up Python

I then proceeded to import the necessary dependencies (libraries and packages) that were needed for the different tasks in the different phases ahead.

```

Import all necessary libraries and packages

[ ] import sys # give access to variables and functions that interact strongly with interpreter
    import matplotlib.pyplot as plt #to plot graphs
    import numpy as np #for linear algebra
    from scipy.stats import randint
    import pandas as pd #processing the datasets, CSV file I/O
    import seaborn as sns #to plot interactive graphs

```

Fig 4 Importing dependencies

5.3.1 Working on the first dataset with crime records from 2005 up to 2012

I first loaded the dataset I intended to use and read it as data frame (df) from it is a Comma Separated Value (CSV) file. This would allow easier modification of the dataset. I printed the head section of the data frame and checked its columns

```

1. Preparing dataset 2005 to 2012

[ ] #loading the dataset
    pathDataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Crimes_Recorded_from_2005_to_2012.csv'

[ ] #Reading the dataset
    df = pd.read_csv(pathDataset)
    df.head()

```

	Deaths (excluding executions)	651	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8
0	Source: Kenya Prisons Department	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	PENAL CODE CASES, 2005 - 2012	NaN	NaN	NaN	NaN	NaN
2	Table:209	NaN	NaN	NaN	NaN	NaN	Number	NaN	NaN
3	Category of offences	2005	2006	2007	2008	2009	2010	2011	2012
4	Homicide	2,313	2,090	1,912	2,037	2,214	2,239	2,641	2761

```

[ ] df.columns

Index(['Deaths (excluding executions)', '651', 'Unnamed: 2', 'Unnamed: 3',
      'Unnamed: 4', 'Unnamed: 5', 'Unnamed: 6', 'Unnamed: 7', 'Unnamed: 8'],
      dtype='object')

```

Fig 5. Preparing a first data set

The data frame's initial length was 23. Upon checking the data frame, the first preparation I did on it was dropping all fields that had null (NaN) values.

```
[ ] len(df)

23

[ ] #Dropping the fields with null(NaN) values
df = df.dropna()

[ ] print(df)
```

	Deaths (excluding executions)	651	...	Unnamed: 7	Unnamed: 8
3	Category of offences	2005	...	2011	2012
4	Homicide	2,313	...	2,641	2761
5	Offences against morality	3,153	...	4,703	4806
6	Other offences against persons	17,304	...	20,144	20698
7	Robbery	6,936	...	3,262	3,262
8	Breakings	8,454	...	7,325	7,578
9	Theft of stock	2,219	...	2,269	2,377
10	Stealing	12,589	...	13,797	14,111
11	Theft by servant	2,874	...	2,889	2,984
12	Vehicles and other thefts	1,718	...	1,768	1,663
13	Dangerous drugs	6,356	...	4,649	4181
14	Traffic offences	38	...	100	66
15	Criminal damage	3,236	...	3,345	3769
16	Economic crimes	1,390	...	3,036	3369
17	Corruption	107	...	52	49
18	Offences involving police officers	29	...	27	69
19	Offences involving tourists	32	...	0	0
20	Other penal code offences	6,652	...	5,726	6109
21	Sub-Total	75,400	...	75,733	77,852

```
[19 rows x 9 columns]
```

Fig 6. Dropping Nan values from the first dataset

I then dropped the data frame's header and made the first row our new header.

```
[ ] #Drop the dataframe's header; make the dataframe's first row new header

df.columns = df.iloc[0]

[ ] print(df)
```

3	Category of offences	2005	2006	...	2010	2011	2012
3	Category of offences	2005	2006	...	2010	2011	2012
4	Homicide	2,313	2,090	...	2,239	2,641	2761
5	Offences against morality	3,153	3,525	...	4,817	4,703	4806
6	Other offences against persons	17,304	18,723	...	20,012	20,144	20698
7	Robbery	6,936	5,234	...	2,843	3,262	3,262
8	Breakings	8,454	7,420	...	6,453	7,325	7,578
9	Theft of stock	2,219	2,209	...	2,244	2,269	2,377
10	Stealing	12,589	10,874	...	11,986	13,797	14,111
11	Theft by servant	2,874	2,700	...	2,591	2,889	2,984
12	Vehicles and other thefts	1,718	1,660	...	1,365	1,768	1,663
13	Dangerous drugs	6,356	5,821	...	5,081	4,649	4181
14	Traffic offences	38	62	...	103	100	66
15	Criminal damage	3,236	3,518	...	3,327	3,345	3769
16	Economic crimes	1,390	1,873	...	2,662	3,036	3369
17	Corruption	107	252	...	62	52	49
18	Offences involving police officers	29	76	...	37	27	69
19	Offences involving tourists	32	84	...	1	0	0
20	Other penal code offences	6,652	6,104	...	4,956	5,726	6109
21	Sub-Total	75,400	72,225	...	70,779	75,733	77,852

```
[19 rows x 9 columns]
```

Fig 7. Dropping data frame's header

```
[ ] # drop first row of dataframe

df = df.iloc[1: , :]
```

```
[ ] print(df)
```

	Category of offences	2005	2006	...	2010	2011	2012
3	Homicide	2,313	2,090	...	2,239	2,641	2761
5	Offences against morality	3,153	3,525	...	4,817	4,703	4806
6	Other offences against persons	17,304	18,723	...	20,012	20,144	20698
7	Robbery	6,936	5,234	...	2,843	3,262	3,262
8	Breakings	8,454	7,420	...	6,453	7,325	7,578
9	Theft of stock	2,219	2,209	...	2,244	2,269	2,377
10	Stealing	12,589	10,874	...	11,986	13,797	14,111
11	Theft by servant	2,874	2,700	...	2,591	2,889	2,984
12	Vehicles and other thefts	1,718	1,660	...	1,365	1,768	1,663
13	Dangerous drugs	6,356	5,821	...	5,081	4,649	4181
14	Traffic offences	38	62	...	103	100	66
15	Criminal damage	3,236	3,518	...	3,327	3,345	3769
16	Economic crimes	1,390	1,873	...	2,662	3,036	3369
17	Corruption	107	252	...	62	52	49
18	Offences involving police officers	29	76	...	37	27	69
19	Offences involving tourists	32	84	...	1	0	0
20	Other penal code offences	6,652	6,104	...	4,956	5,726	6109
21	Sub-Total	75,400	72,225	...	70,779	75,733	77,852

```
[18 rows x 9 columns]
```

Fig. 8 Dropping data frame's header

```
#checking the dataframe columns
df.columns
```

```
Index(['Category of offences', '2005', '2006', '2007', '2008', '2009', '2010',
      '2011', '2012'],
      dtype='object', name=3)
```

```
[ ] #renaming the first column

df.rename(columns={'Category of offences': 'Type of Crime'}, inplace=True)
```

```
[ ] df.columns
```

```
Index(['Type of Crime', '2005', '2006', '2007', '2008', '2009', '2010', '2011',
      '2012'],
      dtype='object', name=3)
```

Fig. 9. Renaming first column 'Category of offenses' to 'Type of Crime'

I continued to drop the last row of the data frame. The data frame now had 17 rows and 9 columns.

```
[ ] #drop last row
df = df.iloc[:-1,:]

[ ] print(df)
```

3	Type of Crime	2005	2006	...	2010	2011	2012
4	Homicide	2,313	2,090	...	2,239	2,641	2761
5	Offences against morality	3,153	3,525	...	4,817	4,703	4806
6	Other offences against persons	17,304	18,723	...	20,012	20,144	20698
7	Robbery	6,936	5,234	...	2,843	3,262	3,262
8	Breakings	8,454	7,420	...	6,453	7,325	7,578
9	Theft of stock	2,219	2,209	...	2,244	2,269	2,377
10	Stealing	12,589	10,874	...	11,986	13,797	14,111
11	Theft by servant	2,874	2,700	...	2,591	2,889	2,984
12	Vehicles and other thefts	1,718	1,660	...	1,365	1,768	1,663
13	Dangerous drugs	6,356	5,821	...	5,081	4,649	4181
14	Traffic offences	38	62	...	103	100	66
15	Criminal damage	3,236	3,518	...	3,327	3,345	3769
16	Economic crimes	1,390	1,873	...	2,662	3,036	3369
17	Corruption	107	252	...	62	52	49
18	Offences involving police officers	29	76	...	37	27	69
19	Offences involving tourists	32	84	...	1	0	0
20	Other penal code offences	6,652	6,104	...	4,956	5,726	6109

```
[17 rows x 9 columns]
```

Fig. 10 Data frame after dropping the last row

I renamed the data frame and saved it as a new CSV file to my drive.

```
[ ] #renaming and saving the updated file of Crimes_Recorded_from_2005_to_2012.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Updated_Crimes_Recorded_from_2005_to_2012.csv')
```

Fig. 11 Renaming and saving data frame as CSV file

5.3.2 Working on the second dataset with crime records from 2010 up to 2019

On this part, I first loaded the CSV dataset to be used and read it as a data frame. I printed the head section of the data frame to check its columns

2. Preparing dataset 2010 to 2019

```
[ ] pathDataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Crimes_Recorded_from_2010_to_2019.csv'
```

```
[ ] #Reading the dataset
df = pd.read_csv(pathDataset)
df.head()
```

	Governance, Peace and Security	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	Unnamed: 10
0	Table 17.2: Crimes Reported to the Police, 201...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Number
2	Crimes1	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
3	Homicide	2,239	2,641	2,761	2,878	2,649	2,648	2,751	2,774	2,856	2,971
4	Offences against morality	4,817	4,703	4,806	4,779	5,184	6,164	6,228	5,492	7,233	8,051

Fig. 12 Preparing a second data set

Upon checking the data frame's head, the first preparation I did on this data set too was dropping all fields that had null (NaN) values.

```
[ ] #Dropping the fields with null(NaN) values
df = df.dropna()
```

```
[ ] print(df)
```

	Governance, Peace and Security	Unnamed: 1	...	Unnamed: 9	Unnamed: 10
2	Crimes1	2010	...	2018	2019
3	Homicide	2,239	...	2,856	2,971
4	Offences against morality	4,817	...	7,233	8,051
5	Other offences against persons	20,012	...	25,049	27,196
6	Robbery	2,843	...	2,935	2,858
7	Breakings	6,453	...	5,970	5,976
8	Theft of stock	2,244	...	2,077	1,962
9	Stealing	11,986	...	12,845	13,954
10	Theft by servant	2,591	...	2,477	2,226
11	Theft of Vehicles and other thefts	1,365	...	1,370	1,298
12	Dangerous drugs	5,081	...	8,021	8,011
13	Traffic offences	103	...	213	341
14	Criminal damage	3,327	...	4,783	4,852
15	Economic crimes	2,662	...	4,100	4,786
16	Corruption	62	...	119	130
17	Offences involving police officers	37	...	174	77
18	Offences involving tourists	1	...	93	48
19	Other penal code offences	4,956	...	7,953	8,674
20	Total	70,779	...	88,268	93,411

[19 rows x 11 columns]

Fig 13. Dropping fields with Nan values

I then dropped the data frame's header, making the first row our new header. I dropped its last row too.

```
[ ] #drop df header; making first row of dataframe be header

df.columns = df.iloc[0]

[ ] # check changes
print(df)
```

2	Crimes1	2010	2011	...	2017	2018	2019
2	Crimes1	2010	2011	...	2017	2018	2019
3	Homicide	2,239	2,641	...	2,774	2,856	2,971
4	Offences against morality	4,817	4,703	...	5,492	7,233	8,051
5	Other offences against persons	20,012	20,144	...	22,515	25,049	27,196
6	Robbery	2,843	3,262	...	2,713	2,935	2,858
7	Breakings	6,453	7,325	...	6,131	5,970	5,976
8	Theft of stock	2,244	2,269	...	2,136	2,077	1,962
9	Stealing	11,986	13,797	...	11,656	12,845	13,954
10	Theft by servant	2,591	2,889	...	2,632	2,477	2,226
11	Theft of Vehicles and other thefts	1,365	1,768	...	1,404	1,370	1,298
12	Dangerous drugs	5,081	4,649	...	5,565	8,021	8,011
13	Traffic offences	103	100	...	69	213	341
14	Criminal damage	3,327	3,345	...	4,262	4,783	4,852
15	Economic crimes	2,662	3,036	...	3,695	4,100	4,786
16	Corruption	62	52	...	75	119	130
17	Offences involving police officers	37	27	...	86	174	77
18	Offences involving tourists	1	0	...	15	93	48
19	Other penal code offences	4,956	5,726	...	6,772	7,953	8,674
20	Total	70,779	75,733	...	77,992	88,268	93,411

```
[19 rows x 11 columns]
```

Fig 14. Dropping data frames header, making the first row be the new header

I continued with the data preparation by dropping the first row and the last row from the data frame

```
[ ] #drop unwanted dataframes rows - first and last

#first row
df = df.iloc[1: , :]

#last row
df = df.iloc[:-1, :]

[ ] print(df)
```

2	Crimes1	2010	2011	...	2017	2018	2019
3	Homicide	2,239	2,641	...	2,774	2,856	2,971
4	Offences against morality	4,817	4,703	...	5,492	7,233	8,051
5	Other offences against persons	20,012	20,144	...	22,515	25,049	27,196
6	Robbery	2,843	3,262	...	2,713	2,935	2,858
7	Breakings	6,453	7,325	...	6,131	5,970	5,976
8	Theft of stock	2,244	2,269	...	2,136	2,077	1,962
9	Stealing	11,986	13,797	...	11,656	12,845	13,954
10	Theft by servant	2,591	2,889	...	2,632	2,477	2,226
11	Theft of Vehicles and other thefts	1,365	1,768	...	1,404	1,370	1,298
12	Dangerous drugs	5,081	4,649	...	5,565	8,021	8,011
13	Traffic offences	103	100	...	69	213	341
14	Criminal damage	3,327	3,345	...	4,262	4,783	4,852
15	Economic crimes	2,662	3,036	...	3,695	4,100	4,786
16	Corruption	62	52	...	75	119	130
17	Offences involving police officers	37	27	...	86	174	77
18	Offences involving tourists	1	0	...	15	93	48
19	Other penal code offences	4,956	5,726	...	6,772	7,953	8,674

```
[17 rows x 11 columns]
```

Fig 15. Data frame after dropping the first row and last row

I then renamed the first column, that is, 'Crimes1' to 'Type of Crime'

```
[ ] #Renaming the first column

df.rename(columns={'Crimes1': 'Type of Crime'}, inplace=True)

[ ] df.columns
```

```
Index(['Type of Crime', '2010', '2011', '2012', '2013', '2014', '2015', '2016',
      '2017', '2018', '2019'],
      dtype='object', name=2)
```

Fig 16. Renaming the first column

The last data preparation that would be done was joining the two data sets to form one data set with criminal records that traverse 2005 up to 2019. The first data set that was prepared had crime records from 2005 to 2012. The second data set had crime records from 2010 to 2019. To join the two data sets, I dropped a few year columns from one data set since they were present in the other data set.

For instance, in this case, I dropped the first four columns from the data frame with crime records from 2010 to 2019. These columns were 'Type of Crime', '2010', '2011' and '2012' which were all present in the dataset with crime records from 2005 to 2012.

```
[ ] #dropping columns

df.drop(['Type of Crime', '2010', '2011', '2012'], axis=1 , inplace=True)
```

```
df.columns
print(df)
```

	2013	2014	2015	2016	2017	2018	2019
2							
3	2,878	2,649	2,648	2,751	2,774	2,856	2,971
4	4,779	5,184	6,164	6,228	5,492	7,233	8,051
5	19,344	19,911	21,174	22,295	22,515	25,049	27,196
6	3,551	3,011	2,865	2,697	2,713	2,935	2,858
7	6,397	5,656	5,591	5,621	6,131	5,970	5,976
8	1,965	1,848	1,961	1,918	2,136	2,077	1,962
9	11,455	10,042	9,528	10,361	11,656	12,845	13,954
10	2,702	2,279	2,184	2,440	2,632	2,477	2,226
11	1,631	1,239	1,111	1,355	1,404	1,370	1,298
12	4,316	4,850	5,525	6,160	5,565	8,021	8,011
13	45	100	120	139	69	213	341
14	3,603	3,709	3,983	4,307	4,262	4,783	4,852
15	2,750	3,038	3,244	3,503	3,695	4,100	4,786
16	57	138	79	92	75	119	130
17	95	53	71	57	86	174	77
18	14	21	19	15	15	93	48
19	6,250	5,648	6,223	7,047	6,772	7,953	8,674

Fig. 17 Data frame after dropping unnecessary columns

I then renamed the data frame, now containing only columns from 2013 to 2019 with their respective yearly entries, and saved it as a new CSV file.

```
[ ] #renaming and saving the updated file of Crimes_Recorded_from_2010_to_2019.csv

df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Updated_Crimes_Recorded_from_2010_to_2019.csv')
```

Fig. 18 Saving the data frame as a new CSV file

5.3.3 Merging the two datasets into a single dataset

I read the datasets as different data frames (df1 and df2) then concatenated the two data frames into one data frame (df3). This data frame now had 17 rows and 18 columns, crime records from 2005 up to 2019. I then saved the data frame as a CSV file

```
Joining the two Datasets

[ ] #reading the datasets to be merged as df1 and df2 respectively
    df1 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Updated Crimes Recorded from 2005 to 2012.csv')
    df2 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Updated Crimes Recorded from 2010 to 2019.csv')
```

Fig. 20 Reading the two data sets to be merged as data frames df1 and df2

```
#joining the dataframes on column

df3 = pd.concat([df1, df2], axis=1)

[ ] print(df3)
```

Unnamed: 0	Type of Crime	...	2018	2019
0	4	Homicide	2,856	2,971
1	5	Offences against morality	7,233	8,051
2	6	Other offences against persons	25,049	27,196
3	7	Robbery	2,935	2,858
4	8	Breakings	5,970	5,976
5	9	Theft of stock	2,077	1,962
6	10	Stealing	12,845	13,954
7	11	Theft by servant	2,477	2,226
8	12	Vehicles and other thefts	1,370	1,298
9	13	Dangerous drugs	8,021	8,011
10	14	Traffic offences	213	341
11	15	Criminal damage	4,783	4,852
12	16	Economic crimes	4,100	4,786
13	17	Corruption	119	130
14	18	Offences involving police officers	174	77
15	19	Offences involving tourists	93	48
16	20	Other penal code offences	7,953	8,674

[17 rows x 18 columns]

Fig. 21 Joining data frames df1 and df2, as a new data frame df3

Our new data frame would now have 17 rows and 18 columns.

```
[ ] #renaming and saving df3 to Crimes_Recorded_from_2005_to_2019.csv
df3.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Crimes_Recorded_from_2005_to_2019.csv')

End of joining the two datasets
```

Fig 22. Saving df3 as a new CSV file with crime records from 2005 to 2019

I then checked the newly created dataset to see if it was correctly created and modified it if there were any anomalies. I had to do a few necessary adjustments.

Updating and modifying the new dataset

```
[ ] #Updating the new dataframe - df3
#loading the data set
pathdataset='/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Crimes_Recorded_from_2005_to_2019.csv'

[ ] #Reading the dataset
df = pd.read_csv(pathdataset)
df.head()
```

	Unnamed: 0	Unnamed: 0.1	Type of Crime	2005	2006	2007	2008	2009	2010	2011	2012	Unnamed: 0.2	2013	2014	2015	2016
0	0	4	Homicide	2,313	2,090	1,912	2,037	2,214	2,239	2,641	2761	3	2,878	2,649	2,648	2,751
1	1	5	Offences against morality	3,153	3,525	3,673	3,116	4,068	4,817	4,703	4806	4	4,779	5,184	6,164	6,228
2	2	6	Other offences against persons	17,304	18,723	17,831	16,496	20,539	20,012	20,144	20698	5	19,344	19,911	21,174	22,295
3	3	7	Robbery	6,936	5,234	3,492	3,401	2,939	2,843	3,262	3,262	6	3,551	3,011	2,865	2,697
4	4	8	Breakings	8,454	7,420	6,337	6,626	7,053	6,453	7,325	7,578	7	6,397	5,656	5,591	5,621

Fig. 23 Checking the newly created data set for any anomalies

I noticed that the data frame had ‘Unnamed’ columns which should not be present. I, therefore, went ahead and dropped these columns.

```
[ ] df.columns

Index(['Unnamed: 0', 'Unnamed: 0.1', 'Type of Crime', '2005', '2006', '2007',
      '2008', '2009', '2010', '2011', '2012', 'Unnamed: 0.2', '2013', '2014',
      '2015', '2016', '2017', '2018', '2019'],
      dtype='object')

[ ] #dropping unwanted columns

df.drop(['Unnamed: 0', 'Unnamed: 0.1', 'Unnamed: 0.2'], axis=1, inplace=True)

[ ] df.columns

Index(['Type of Crime', '2005', '2006', '2007', '2008', '2009', '2010', '2011',
      '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019'],
      dtype='object')
```

Fig. 24 Display of the data frame's column before and after dropping the unwanted columns

```
[ ] df.head()
```

	Type of Crime	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
0	Homicide	2,313	2,090	1,912	2,037	2,214	2,239	2,641	2,761	2,878	2,649	2,648	2,751	2,774	2,856	2,971
1	Offences against morality	3,153	3,525	3,673	3,116	4,068	4,817	4,703	4,806	4,779	5,184	6,164	6,228	5,492	7,233	8,051
2	Other offences against persons	17,304	18,723	17,831	16,496	20,539	20,012	20,144	20,698	19,344	19,911	21,174	22,295	22,515	25,049	27,196
3	Robbery	6,936	5,234	3,492	3,401	2,939	2,843	3,262	3,262	3,551	3,011	2,865	2,697	2,713	2,935	2,858
4	Breakings	8,454	7,420	6,337	6,626	7,053	6,453	7,325	7,578	6,397	5,656	5,591	5,621	6,131	5,970	5,976

Fig. 25 Display of the data frames head with wanted columns after dropping the unwanted columns

Below is a pictorial representation of the data frame in CSV format, prepared after cleaning and merging the two datasets that I had mined and updating the newly created data frame after merging.

		1 to 17 of 17 entries															Filter
	Type of Crime	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	
0	Homicide	2,313	2,090	1,912	2,037	2,214	2,239	2,641	2,761	2,878	2,649	2,648	2,751	2,774	2,856	2,971	
1	Offences against morality	3,153	3,525	3,673	3,116	4,068	4,817	4,703	4,806	4,779	5,184	6,164	6,228	5,492	7,233	8,051	
2	Other offences against persons	17,304	18,723	17,831	16,496	20,539	20,012	20,144	20,698	19,344	19,911	21,174	22,295	22,515	25,049	27,196	
3	Robbery	6,936	5,234	3,492	3,401	2,939	2,843	3,262	3,262	3,551	3,011	2,865	2,697	2,713	2,935	2,858	
4	Breakings	8,454	7,420	6,337	6,626	7,053	6,453	7,325	7,578	6,397	5,656	5,591	5,621	6,131	5,970	5,976	
5	Theft of stock	2,219	2,209	1,568	2,269	2,876	2,244	2,269	2,377	1,965	1,848	1,961	1,918	2,136	2,077	1,962	
6	Stealing	12,589	10,874	10,749	11,435	11,972	11,986	13,797	14,111	11,455	10,042	9,528	10,361	11,656	12,845	13,954	
7	Theft by servant	2,874	2,700	2,169	2,387	2,732	2,591	2,889	2,984	2,702	2,279	2,184	2,440	2,632	2,477	2,226	
8	Vehicles and other thefts	1,718	1,660	1,221	1,387	1,439	1,365	1,768	1,663	1,631	1,239	1,111	1,355	1,404	1,370	1,298	
9	Dangerous drugs	6,356	5,821	5,401	4,407	5,541	5,081	4,649	4,181	4,316	4,850	5,525	6,160	5,565	8,021	8,011	
10	Traffic offences	38	62	46	120	59	103	100	66	45	100	120	139	69	213	341	
11	Criminal damage	3,236	3,518	2,770	3,760	3,417	3,327	3,345	3,769	3,603	3,709	3,983	4,307	4,262	4,783	4,852	
12	Economic crimes	1,390	1,873	1,908	1,898	2,324	2,662	3,036	3,369	2,750	3,038	3,244	3,503	3,695	4,100	4,786	
13	Corruption	107	252	177	133	158	62	52	49	57	138	79	92	75	119	130	
14	Offences involving police officers	29	76	32	33	56	37	27	69	95	53	71	57	86	174	77	
15	Offences involving tourists	32	84	10	6	5	1	0	0	14	21	19	15	15	93	48	
16	Other penal code offences	6,652	6,104	3,732	3,994	4,864	4,956	5,726	6,109	6,250	5,648	6,223	7,047	6,772	7,953	8,674	

Show 50 per page

Fig. 26 Display of the merged data set in CSV format containing crimes records from 2005 to 2019 on Google Colab

5.3.4 Working on the merged dataset -- Crime records from 2005 up to 2019 – to create a new dataset with types of criminal records and corresponding yearly GDP values.

This part aimed to prepare a data set with GDP values for each year from 2005 up to 2019. I first loaded the data set to be used, which was the merged data set with crime records from 2005 to 2019, and then read it as a data frame.

```

Creating new yearly datasets with corresponding yearly GDP values

1. Loading the dataset

[ ] pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

```

Fig. 27 Reading the CSV data set to be used

5.3.4.1 Creating dataset 2005 with GDP value

2.1 Creating dataset 2005 with GDP value

```
[ ] # Creating yearly datasets from Master datasets by  
# splitting dataset yearly  
# and adding corresponding yearly GDPs as new column  
  
df.head()
```

Unnamed: 0	Type of Crime	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
0	Homicide	2,313	2,090	1,912	2,037	2,214	2,239	2,641	2,761	2,878	2,649	2,648	2,751	2,774	2,856	2,971
1	Offences against morality	3,153	3,525	3,673	3,116	4,068	4,817	4,703	4,806	4,779	5,184	6,164	6,228	5,492	7,233	8,051
2	Other offences against persons	17,304	18,723	17,831	16,496	20,539	20,012	20,144	20,698	19,344	19,911	21,174	22,295	22,515	25,049	27,196
3	Robbery	6,936	5,234	3,492	3,401	2,939	2,843	3,262	3,262	3,551	3,011	2,865	2,697	2,713	2,935	2,858
4	Breakings	8,454	7,420	6,337	6,626	7,053	6,453	7,325	7,578	6,397	5,656	5,591	5,621	6,131	5,970	5,976

Fig. 28 Display of the data frame to be used head part

To create the single yearly datasets I took the following steps:

(I'll use preparation of 2005 data set with Crime records and that year GDP value as an example of preparing the single data sets from the data set containing crime records from 2005 to 2019)

i. Drop the columns not to be used from the data frame

```
[ ] # Creating df2005 -> Crimes_Recorded_in_2005_and_Corresponding_GDP  
  
# drop unwanted columns  
df.drop(['Unnamed: 0', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019'])  
  
df.head()
```

	Type of Crime	2005
0	Homicide	2,313
1	Offences against morality	3,153
2	Other offences against persons	17,304
3	Robbery	6,936
4	Breakings	8,454

Fig.29 Dropping unwanted columns and a display of the head part that remains of the data frame

ii. **Rename the column e.g '2005' in this case to 'Number of Crimes Reported'**

```
[ ] # Rename column '2005' to 'Number of Crimes Reported'
df.rename(columns={'2005': 'Number of Crimes Reported'}, inplace=True)
```

df.head()

	Type of Crime	Number of Crimes Reported
0	Homicide	2,313
1	Offences against morality	3,153
2	Other offences against persons	17,304
3	Robbery	6,936
4	Breakings	8,454

Fig.30 Data frame after renaming '2005' to 'Number of Crimes Reported'

iii. **Add two new columns 'Year' and 'Gross Domestic Product (GDP)'**

I add two new columns 'Year' with 17 entries which are similar dates i.e '2005-01-01' in this case. This is an assumption that the records were all entered on the same day. The other new column added is 'Gross Domestic Product (GDP)' with 17 entries too which are GDP values i.e '5.9' which was the GDP value in 2005.

```
[ ] # add new column 'Year' with date entries
df.insert(0, 'Year', (["2005-01-01"]*17))

# add new column 'GDP' and value
df.insert(3, 'Gross Domenstic Product (GDP)', ([5.9]*17))
```

df.head()

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2005-01-01	Homicide	2,313	5.9
1	2005-01-01	Offences against morality	3,153	5.9
2	2005-01-01	Other offences against persons	17,304	5.9
3	2005-01-01	Robbery	6,936	5.9
4	2005-01-01	Breakings	8,454	5.9

Fig.31 Display of data frame head after adding two columns i.e Year with date entries and Gross Domestic Product (GDP) with value '5.9'

iv. Saving the data frame as a new CSV file dataset

```
#saving the new df to CSV file -> 2005_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2005_Crimes_and_GDP.csv')
```

Fig.31.1 Saving the newly created single data set with 2005 Crime records and the year's GDP

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2005-01-01	Homicide	2,313	5.9
1	2005-01-01	Offences against morality	3,153	5.9
2	2005-01-01	Other offences against persons	17,304	5.9
3	2005-01-01	Robbery	6,936	5.9
4	2005-01-01	Breakings	8,454	5.9
5	2005-01-01	Theft of stock	2,219	5.9
6	2005-01-01	Stealing	12,589	5.9
7	2005-01-01	Theft by servant	2,874	5.9
8	2005-01-01	Vehicles and other thefts	1,718	5.9
9	2005-01-01	Dangerous drugs	6,356	5.9
10	2005-01-01	Traffic offences	38	5.9
11	2005-01-01	Criminal damage	3,236	5.9
12	2005-01-01	Economic crimes	1,390	5.9
13	2005-01-01	Corruption	107	5.9
14	2005-01-01	Offences involving police officers	29	5.9
15	2005-01-01	Offences involving tourists	32	5.9
16	2005-01-01	Other penal code offences	6,652	5.9

Fig.31.2 2005 data frame snap

5.3.4.2 Creating dataset 2006 with the year's GDP value

```
pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

#drop unwanted columns
df.drop(['Unnamed: 0', '2005', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2006' to 'Number of Crimes Reported'
df.rename(columns={'2006': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2006-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([6.3]*17))

#saving the new df to CSV file -> 2006_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2006_Crimes_and_GDP.csv')
```

Fig.32 Creating and Saving the new single data set with 2006 Crime records and the year's GDP

2006_Crimes_and_GDP.csv X				
				1 to 17 of 17 entries Filter
	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2006-01-01	Homicide	2,090	6.3
1	2006-01-01	Offences against morality	3,525	6.3
2	2006-01-01	Other offences against persons	18,723	6.3
3	2006-01-01	Robbery	5,234	6.3
4	2006-01-01	Breakings	7,420	6.3
5	2006-01-01	Theft of stock	2,209	6.3
6	2006-01-01	Stealing	10,874	6.3
7	2006-01-01	Theft by servant	2,700	6.3
8	2006-01-01	Vehicles and other thefts	1,660	6.3
9	2006-01-01	Dangerous drugs	5,821	6.3
10	2006-01-01	Traffic offences	62	6.3
11	2006-01-01	Criminal damage	3,518	6.3
12	2006-01-01	Economic crimes	1,873	6.3
13	2006-01-01	Corruption	252	6.3
14	2006-01-01	Offences involving police officers	76	6.3
15	2006-01-01	Offences involving tourists	84	6.3
16	2006-01-01	Other penal code offences	6,104	6.3

Fig.32.1 2006 data frame snap

5.3.4.3 Creating dataset 2007 with the year's GDP value

```
#Creating new CSV file -> 2007_Crimes_and_GDP.csv from Master Dataset
pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

#drop unwanted column
df.drop(['Unnamed: 0', '2005', '2006', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2007' to 'Number of Crimes Reported'
df.rename(columns={'2007': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2007-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([7.0]*17))

#saving the new df to CSV file -> 2007_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2007_Crimes_and_GDP.csv')
```

Fig.33 Creating and Saving the new single data set with 2007 Crime records and the year's GDP

2007_Crimes_and_GDP.csv X

1 to 17 of 17 entries Filter

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2007-01-01	Homicide	1,912	7.0
1	2007-01-01	Offences against morality	3,673	7.0
2	2007-01-01	Other offences against persons	17,831	7.0
3	2007-01-01	Robbery	3,492	7.0
4	2007-01-01	Breakings	6,337	7.0
5	2007-01-01	Theft of stock	1,568	7.0
6	2007-01-01	Stealing	10,749	7.0
7	2007-01-01	Theft by servant	2,169	7.0
8	2007-01-01	Vehicles and other thefts	1,221	7.0
9	2007-01-01	Dangerous drugs	5,401	7.0
10	2007-01-01	Traffic offences	46	7.0
11	2007-01-01	Criminal damage	2,770	7.0
12	2007-01-01	Economic crimes	1,908	7.0
13	2007-01-01	Corruption	177	7.0
14	2007-01-01	Offences involving police officers	32	7.0
15	2007-01-01	Offences involving tourists	10	7.0
16	2007-01-01	Other penal code offences	3,732	7.0

Fig.33.1 2007 data frame snap

5.3.4.4 Creating dataset 2008 with the year's GDP value

```
#Creating new CSV file -> 2008_Crimes_and_GDP.csv from Master Dataset

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2008' to 'Number of Crimes Reported'
df.rename(columns={'2008': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2008-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([1.5]*17))

#saving the new df to CSV file -> 2008_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2008_Crimes_and_GDP.csv')
```

Fig.34 Creating and Saving the new single data set with 2008 Crime records and the year's GDP

2008_Crimes_and_GDP.csv X				
	Year	Type of Crime	Number of Crimes Reported	1 to 17 of 17 entries Gross Domenstic Product (GDP)
0	2008-01-01	Homicide	2,037	1.5
1	2008-01-01	Offences against morality	3,116	1.5
2	2008-01-01	Other offences against persons	16,496	1.5
3	2008-01-01	Robbery	3,401	1.5
4	2008-01-01	Breakings	6,626	1.5
5	2008-01-01	Theft of stock	2,269	1.5
6	2008-01-01	Stealing	11,435	1.5
7	2008-01-01	Theft by servant	2,387	1.5
8	2008-01-01	Vehicles and other thefts	1,387	1.5
9	2008-01-01	Dangerous drugs	4,407	1.5
10	2008-01-01	Traffic offences	120	1.5
11	2008-01-01	Criminal damage	3,760	1.5
12	2008-01-01	Economic crimes	1,898	1.5
13	2008-01-01	Corruption	133	1.5
14	2008-01-01	Offences involving police officers	33	1.5
15	2008-01-01	Offences involving tourists	6	1.5
16	2008-01-01	Other penal code offences	3,994	1.5

Fig.32.1 2008 data frame snap

5.3.4.5 Creating dataset 2009 the year's with GDP value

```
#Creating new CSV file -> 2009_Crimes_and_GDP.csv from Master Dataset

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2009' to 'Number of Crimes Reported'
df.rename(columns={'2009': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2009-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([2.7]*17))

#saving the new df to CSV file -> 2009_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2009_Crimes_and_GDP.csv')
```

Fig.35 Creating and Saving the new single data set with 2009 Crime records and the year's GDP

2009_Crimes_and_GDP.csv X				
				1 to 17 of 17 entries
	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2009-01-01	Homicide	2,214	2.7
1	2009-01-01	Offences against morality	4,068	2.7
2	2009-01-01	Other offences against persons	20,539	2.7
3	2009-01-01	Robbery	2,939	2.7
4	2009-01-01	Breakings	7,053	2.7
5	2009-01-01	Theft of stock	2,876	2.7
6	2009-01-01	Stealing	11,972	2.7
7	2009-01-01	Theft by servant	2,732	2.7
8	2009-01-01	Vehicles and other thefts	1,439	2.7
9	2009-01-01	Dangerous drugs	5,541	2.7
10	2009-01-01	Traffic offences	59	2.7
11	2009-01-01	Criminal damage	3,417	2.7
12	2009-01-01	Economic crimes	2,324	2.7
13	2009-01-01	Corruption	158	2.7
14	2009-01-01	Offences involving police officers	56	2.7
15	2009-01-01	Offences involving tourists	5	2.7
16	2009-01-01	Other penal code offences	4,864	2.7

Fig.35.1 2009 data frame snap

5.3.4.6 Creating dataset 2010 with the year's GDP value

```
#Creating new CSV file -> 2010_Crimes_and_GDP.csv from Master Dataset

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2010' to 'Number of Crimes Reported'
df.rename(columns={'2010': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', ([["2010-01-01"]*17]))
df.insert(3, 'Gross Domenstic Product (GDP)', ([5.8]*17))

#saving the new df to CSV file -> 2010_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2010_Crimes_and_GDP.csv')
```

Fig.36 Creating and Saving the new single data set with 2010 Crime records and the year's GDP

2010_Crimes_and_GDP.csv X

	Year	Type of Crime	Number of Crimes Reported	1 to 17 of 17 entries Gross Domenstic Product (GDP)
0	2010-01-01	Homicide	2,239	5.8
1	2010-01-01	Offences against morality	4,817	5.8
2	2010-01-01	Other offences against persons	20,012	5.8
3	2010-01-01	Robbery	2,843	5.8
4	2010-01-01	Breakings	6,453	5.8
5	2010-01-01	Theft of stock	2,244	5.8
6	2010-01-01	Stealing	11,986	5.8
7	2010-01-01	Theft by servant	2,591	5.8
8	2010-01-01	Vehicles and other thefts	1,365	5.8
9	2010-01-01	Dangerous drugs	5,081	5.8
10	2010-01-01	Traffic offences	103	5.8
11	2010-01-01	Criminal damage	3,327	5.8
12	2010-01-01	Economic crimes	2,662	5.8
13	2010-01-01	Corruption	62	5.8
14	2010-01-01	Offences involving police officers	37	5.8
15	2010-01-01	Offences involving tourists	1	5.8
16	2010-01-01	Other penal code offences	4,956	5.8

Fig.36.1 2010 data frame snap

5.3.4.7 Creating dataset 2011 with the year's GDP value

```
#Creating new CSV file -> 2011_Crimes_and_GDP.csv from Master Dataset

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2010', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2011' to 'Number of Crimes Reported'
df.rename(columns={'2011': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2011-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([4.4]*17))

#saving the new df to CSV file -> 2011_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2011_Crimes_and_GDP.csv')
```

Fig.37 Creating and Saving the new single data set with 2011 Crime records and the year's GDP

2011_Crimes_and_GDP.csv X

1 to 17 of 17 entries Filter

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2011-01-01	Homicide	2,641	4.4
1	2011-01-01	Offences against morality	4,703	4.4
2	2011-01-01	Other offences against persons	20,144	4.4
3	2011-01-01	Robbery	3,262	4.4
4	2011-01-01	Breakings	7,325	4.4
5	2011-01-01	Theft of stock	2,269	4.4
6	2011-01-01	Stealing	13,797	4.4
7	2011-01-01	Theft by servant	2,889	4.4
8	2011-01-01	Vehicles and other thefts	1,768	4.4
9	2011-01-01	Dangerous drugs	4,649	4.4
10	2011-01-01	Traffic offences	100	4.4
11	2011-01-01	Criminal damage	3,345	4.4
12	2011-01-01	Economic crimes	3,036	4.4
13	2011-01-01	Corruption	52	4.4
14	2011-01-01	Offences involving police officers	27	4.4
15	2011-01-01	Offences involving tourists	0	4.4
16	2011-01-01	Other penal code offences	5,726	4.4

Fig.37.1 2011 data frame snap

5.3.4.8 Creating dataset 2012 with the year's GDP value

```
#Creating new CSV file -> 2012_Crimes_and_GDP.csv from Master Dataset

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2013', '2014', '2015', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2012' to 'Number of Crimes Reported'
df.rename(columns={'2012': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2012-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([4.6]*17))

#saving the new df to CSV file -> 2012_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2012_Crimes_and_GDP.csv')
```

Fig.38 Creating and Saving the new single data set with 2012 Crime records and the year's GDP

2012_Crimes_and_GDP.csv X

1 to 17 of 17 entries Filter

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2012-01-01	Homicide	2761	4.6
1	2012-01-01	Offences against morality	4806	4.6
2	2012-01-01	Other offences against persons	20698	4.6
3	2012-01-01	Robbery	3,262	4.6
4	2012-01-01	Breakings	7,578	4.6
5	2012-01-01	Theft of stock	2,377	4.6
6	2012-01-01	Stealing	14,111	4.6
7	2012-01-01	Theft by servant	2,984	4.6
8	2012-01-01	Vehicles and other thefts	1,663	4.6
9	2012-01-01	Dangerous drugs	4181	4.6
10	2012-01-01	Traffic offences	66	4.6
11	2012-01-01	Criminal damage	3769	4.6
12	2012-01-01	Economic crimes	3369	4.6
13	2012-01-01	Corruption	49	4.6
14	2012-01-01	Offences involving police officers	69	4.6
15	2012-01-01	Offences involving tourists	0	4.6
16	2012-01-01	Other penal code offences	6109	4.6

Fig.38.1 2012 data frame snap

5.3.4.9 Creating dataset 2013 with the year's GDP value

```
#Creating new CSV file -> 2013_Crimes_and_GDP.csv from Master Dataset

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2014', '2015', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2013' to 'Number of Crimes Reported'
df.rename(columns={'2013': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2013-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([5.9]*17))

#saving the new df to CSV file -> 2005_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2013 Crimes and GDP.csv')
```

Fig.39 Creating and Saving the new single data set with 2013 Crime records and the year's GDP

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2013-01-01	Homicide	2,878	5.9
1	2013-01-01	Offences against morality	4,779	5.9
2	2013-01-01	Other offences against persons	19,344	5.9
3	2013-01-01	Robbery	3,551	5.9
4	2013-01-01	Breakings	6,397	5.9
5	2013-01-01	Theft of stock	1,965	5.9
6	2013-01-01	Stealing	11,455	5.9
7	2013-01-01	Theft by servant	2,702	5.9
8	2013-01-01	Vehicles and other thefts	1,631	5.9
9	2013-01-01	Dangerous drugs	4,316	5.9
10	2013-01-01	Traffic offences	45	5.9
11	2013-01-01	Criminal damage	3,603	5.9
12	2013-01-01	Economic crimes	2,750	5.9
13	2013-01-01	Corruption	57	5.9
14	2013-01-01	Offences involving police officers	95	5.9
15	2013-01-01	Offences involving tourists	14	5.9
16	2013-01-01	Other penal code offences	6,250	5.9

Fig.39.1 2013 data frame snap

5.3.4.10 Creating dataset 2014 with the year's GDP value

```

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2015', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2014' to 'Number of Crimes Reported'
df.rename(columns={'2014': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2014-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([5.4]*17))

#saving the new df to CSV file -> 2005_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2014_Crimes_and_GDP.csv')

```

Fig.40 Creating and Saving the new single data set with 2014 Crime records and the year's GDP

2014_Crimes_and_GDP.csv X

1 to 17 of 17 entries Filter

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2014-01-01	Homicide	2,649	5.4
1	2014-01-01	Offences against morality	5,184	5.4
2	2014-01-01	Other offences against persons	19,911	5.4
3	2014-01-01	Robbery	3,011	5.4
4	2014-01-01	Breakings	5,656	5.4
5	2014-01-01	Theft of stock	1,848	5.4
6	2014-01-01	Stealing	10,042	5.4
7	2014-01-01	Theft by servant	2,279	5.4
8	2014-01-01	Vehicles and other thefts	1,239	5.4
9	2014-01-01	Dangerous drugs	4,850	5.4
10	2014-01-01	Traffic offences	100	5.4
11	2014-01-01	Criminal damage	3,709	5.4
12	2014-01-01	Economic crimes	3,038	5.4
13	2014-01-01	Corruption	138	5.4
14	2014-01-01	Offences involving police officers	53	5.4
15	2014-01-01	Offences involving tourists	21	5.4
16	2014-01-01	Other penal code offences	5,648	5.4

Fig.40.1 2014 data frame snap

5.3.4.11 Creating dataset 2015 with the year's GDP value

```

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2016', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2015' to 'Number of Crimes Reported'
df.rename(columns={'2015': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2015-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([5.7]*17))

#saving the new df to CSV file -> 2015_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2015_Crimes_and_GDP.csv')

```

Fig.41 Creating and Saving the new single data set with 2015 Crime records and the year's GDP

2015_Crimes_and_GDP.csv X				
	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2015-01-01	Homicide	2,648	5.7
1	2015-01-01	Offences against morality	6,164	5.7
2	2015-01-01	Other offences against persons	21,174	5.7
3	2015-01-01	Robbery	2,865	5.7
4	2015-01-01	Breakings	5,591	5.7
5	2015-01-01	Theft of stock	1,961	5.7
6	2015-01-01	Stealing	9,528	5.7
7	2015-01-01	Theft by servant	2,184	5.7
8	2015-01-01	Vehicles and other thefts	1,111	5.7
9	2015-01-01	Dangerous drugs	5,525	5.7
10	2015-01-01	Traffic offences	120	5.7
11	2015-01-01	Criminal damage	3,983	5.7
12	2015-01-01	Economic crimes	3,244	5.7
13	2015-01-01	Corruption	79	5.7
14	2015-01-01	Offences involving police officers	71	5.7
15	2015-01-01	Offences involving tourists	19	5.7
16	2015-01-01	Other penal code offences	6,223	5.7

Fig.42.1 2015 data frame snap

5.3.4.12 Creating dataset 2016 with the year's GDP value

```

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2017', '2018', '2019'], axis=1, inplace=True)

#Rename column '2016' to 'Number of Crimes Reported'
df.rename(columns={'2016': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2016-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([5.9]*17))

#saving the new df to CSV file -> 2016_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2016_Crimes_and_GDP.csv')

```

Fig.43 Creating and Saving the new single data set with 2016 Crime records and the year's GDP

2016_Crimes_and_GDP.csv X				
	Year	Type of Crime	Number of Crimes Reported	1 to 17 of 17 entries Gross Domenstic Product (GDP)
0	2016-01-01	Homicide	2,751	5.9
1	2016-01-01	Offences against morality	6,228	5.9
2	2016-01-01	Other offences against persons	22,295	5.9
3	2016-01-01	Robbery	2,697	5.9
4	2016-01-01	Breakings	5,621	5.9
5	2016-01-01	Theft of stock	1,918	5.9
6	2016-01-01	Stealing	10,361	5.9
7	2016-01-01	Theft by servant	2,440	5.9
8	2016-01-01	Vehicles and other thefts	1,355	5.9
9	2016-01-01	Dangerous drugs	6,160	5.9
10	2016-01-01	Traffic offences	139	5.9
11	2016-01-01	Criminal damage	4,307	5.9
12	2016-01-01	Economic crimes	3,503	5.9
13	2016-01-01	Corruption	92	5.9
14	2016-01-01	Offences involving police officers	57	5.9
15	2016-01-01	Offences involving tourists	15	5.9
16	2016-01-01	Other penal code offences	7,047	5.9

Fig.43.1 2016 data frame snap

5.3.4.13 Creating dataset 2017 with the year's GDP value

```

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2018', '2019'], axis=1, inplace=True)

#Rename column '2017' to 'Number of Crimes Reported'
df.rename(columns={'2017': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2017-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([4.8]*17))

#saving the new df to CSV file -> 2017_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2017_Crimes_and_GDP.csv')

```

Fig.44 Creating and Saving the new single data set with 2017 Crime records and the year's GDP

2017_Crimes_and_GDP.csv X

1 to 17 of 17 entries

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2017-01-01	Homicide	2,774	4.8
1	2017-01-01	Offences against morality	5,492	4.8
2	2017-01-01	Other offences against persons	22,515	4.8
3	2017-01-01	Robbery	2,713	4.8
4	2017-01-01	Breakings	6,131	4.8
5	2017-01-01	Theft of stock	2,136	4.8
6	2017-01-01	Stealing	11,656	4.8
7	2017-01-01	Theft by servant	2,632	4.8
8	2017-01-01	Vehicles and other thefts	1,404	4.8
9	2017-01-01	Dangerous drugs	5,565	4.8
10	2017-01-01	Traffic offences	69	4.8
11	2017-01-01	Criminal damage	4,262	4.8
12	2017-01-01	Economic crimes	3,695	4.8
13	2017-01-01	Corruption	75	4.8
14	2017-01-01	Offences involving police officers	86	4.8
15	2017-01-01	Offences involving tourists	15	4.8
16	2017-01-01	Other penal code offences	6,772	4.8

Fig.44.1 2017 data frame snap

5.3.4.14 Creating dataset 2018 with the year's GDP value

```
#Creating new CSV file -> 2018_Crimes_and_GDP.csv from Master Dataset

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2019'], axis=1, inplace=True)

#Rename column '2018' to 'Number of Crimes Reported'
df.rename(columns={'2018': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2018-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([6.3]*17))

#saving the new df to CSV file -> 2018_Crimes_and_GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2018_Crimes_and_GDP.csv')
```

Fig.45 Creating and Saving the new single data set with 2018 Crime records and the year's GDP

2018_Crimes_and_GDP.csv X				
	Year	Type of Crime	Number of Crimes Reported	1 to 17 of 17 entries Gross Domenstic Product (GDP)
0	2018-01-01	Homicide	2,856	6.3
1	2018-01-01	Offences against morality	7,233	6.3
2	2018-01-01	Other offences against persons	25,049	6.3
3	2018-01-01	Robbery	2,935	6.3
4	2018-01-01	Breakings	5,970	6.3
5	2018-01-01	Theft of stock	2,077	6.3
6	2018-01-01	Stealing	12,845	6.3
7	2018-01-01	Theft by servant	2,477	6.3
8	2018-01-01	Vehicles and other thefts	1,370	6.3
9	2018-01-01	Dangerous drugs	8,021	6.3
10	2018-01-01	Traffic offences	213	6.3
11	2018-01-01	Criminal damage	4,783	6.3
12	2018-01-01	Economic crimes	4,100	6.3
13	2018-01-01	Corruption	119	6.3
14	2018-01-01	Offences involving police officers	174	6.3
15	2018-01-01	Offences involving tourists	93	6.3
16	2018-01-01	Other penal code offences	7,953	6.3

Fig.45.1 2018 data frame snap

5.3.4.15 Creating dataset 2019 with the year's GDP value

```

pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Record_of_Crimes_from_2005_to_2019.csv'
df = pd.read_csv(pathdataset)

df.drop(['Unnamed: 0', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018'], axis=1, inplace=True)

#Rename column '2019' to 'Number of Crimes Reported'
df.rename(columns={'2019': 'Number of Crimes Reported'}, inplace=True)

# add new column 'Year' before 'Type of Crime'
df.insert(0, 'Year', (["2019-01-01"]*17))
df.insert(3, 'Gross Domenstic Product (GDP)', ([5.4]*17))

#saving the new df to CSV file -> 2019 Crimes and GDP.csv
df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2019_Crimes_and_GDP.csv')

```

Fig.46 Creating and Saving the new single data set with 2019 Crime records and the year's GDP

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2019-01-01	Homicide	2,971	5.4
1	2019-01-01	Offences against morality	8,051	5.4
2	2019-01-01	Other offences against persons	27,196	5.4
3	2019-01-01	Robbery	2,858	5.4
4	2019-01-01	Breakings	5,976	5.4
5	2019-01-01	Theft of stock	1,962	5.4
6	2019-01-01	Stealing	13,954	5.4
7	2019-01-01	Theft by servant	2,226	5.4
8	2019-01-01	Vehicles and other thefts	1,298	5.4
9	2019-01-01	Dangerous drugs	8,011	5.4
10	2019-01-01	Traffic offences	341	5.4
11	2019-01-01	Criminal damage	4,852	5.4
12	2019-01-01	Economic crimes	4,786	5.4
13	2019-01-01	Corruption	130	5.4
14	2019-01-01	Offences involving police officers	77	5.4
15	2019-01-01	Offences involving tourists	48	5.4
16	2019-01-01	Other penal code offences	8,674	5.4

Fig.47.1 2019 data frame snap

5.3.5 Merging the single yearly datasets to create a new dataset with yearly GDPs

I first read all the single yearly datasets as data frames before proceeding to join them

```
#Reading the csv datasets to dataframes
df2005 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2005_Crimes_and_GDP.csv')
df2006 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2006_Crimes_and_GDP.csv')
df2007 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2007_Crimes_and_GDP.csv')
df2008 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2008_Crimes_and_GDP.csv')
df2009 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2009_Crimes_and_GDP.csv')
df2010 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2010_Crimes_and_GDP.csv')
df2011 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2011_Crimes_and_GDP.csv')
df2012 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2012_Crimes_and_GDP.csv')
df2013 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2013_Crimes_and_GDP.csv')
df2014 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2014_Crimes_and_GDP.csv')
df2015 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2015_Crimes_and_GDP.csv')
df2016 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2016_Crimes_and_GDP.csv')
df2017 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2017_Crimes_and_GDP.csv')
df2018 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2018_Crimes_and_GDP.csv')
df2019 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/2019_Crimes_and_GDP.csv')
```

Fig.48 Reading the single data frames

I then joined the data frames along the rows and printed out the columns of the newly created data frame.

```
#Joining the dfs read along the rows
df = df2005.append([df2006, df2007, df2008, df2009, df2010, df2011, df2012, df2013, df2014, df2015, df2016, df2017, df2018], ignore_index=True, sort=False)

df.columns

Index(['Unnamed: 0', 'Year', 'Type of Crime', 'Number of Crimes Reported',
      'Gross Domenstic Product (GDP)'],
      dtype='object')
```

Fig.49 Joining the single data frames across rows and printing the column of the newly created data frame from this join

The next step is updating the data frame's, in which I did a few modifications to the data frame like:

5.3.5.1 Dropping unwanted columns on the new dataset with yearly GDPs

```
#Updating the final dataset - dropping unwanted columns
df.drop(['Unnamed: 0'], axis=1, inplace=True)

df.head()
```

	Year	Type of Crime	Number of Crimes Reported	Gross Domenstic Product (GDP)
0	2005-01-01	Homicide	2,313	5.9
1	2005-01-01	Offences against morality	3,153	5.9
2	2005-01-01	Other offences against persons	17,304	5.9
3	2005-01-01	Robbery	6,936	5.9
4	2005-01-01	Breakings	8,454	5.9

Fig.50 Dropping unwanted column

5.3.5.2 Removing all the special characters from the 'Number of Crimes Reported' column.

This was to ensure that the entries in this column which were integer types, had no other features like commas whatsoever but only integers.

Removing all the special features from the "Number of Crimes Reported" column

```
[ ] cols = ['Number of Crimes Reported']
df[cols] = df[cols].astype(str) # cast to string

# Removing special characters
df[cols] = df[cols].replace({'\': ' ', ',': ' ', '-': ' '}, regex=True)

df.head()
```

	Year	Type of Crime	Number of Crimes Reported	Gross Domestic Product (GDP)
0	2005-01-01	Homicide	2313	5.9
1	2005-01-01	Offences against morality	3153	5.9
2	2005-01-01	Other offences against persons	17304	5.9
3	2005-01-01	Robbery	6936	5.9
4	2005-01-01	Breakings	8454	5.9

Fig.50.1 Removing special characters

5.3.5.3 Renaming the column

The columns which were named as strings had to be modified. The strings had white space in between and it was important to do away with the white spaces. In Python, white spaces represent ‘empty’ and this would lead to errors ahead. I achieved this by adding underscores in between the words.

For example, I renamed the column

‘Type of Crime’ to ‘Type_of_Crime’

and

‘Number of Crimes Reported’ to ‘Number_of_Crimes_Reported’

```
Renaming the columns
```

```
[ ] df.rename(columns={'Type of Crime': 'Type_of_Crime'}, inplace=True)
    df.rename(columns={'Number of Crimes Reported': 'Number_of_Crimes_Reported'}, inplace=True)
    df.rename(columns={'Gross Domestic Product (GDP)': 'Gross_Domestic_Product_GDP'}, inplace=True)
```

```
[ ] df.head()
```

	Year	Type_of_Crime	Number_of_Crimes_Reported	Gross_Domestic_Product_GDP
0	2005-01-01	Homicide	2313	5.9
1	2005-01-01	Offences against morality	3153	5.9
2	2005-01-01	Other offences against persons	17304	5.9
3	2005-01-01	Robbery	6936	5.9
4	2005-01-01	Breakings	8454	5.9

Fig.51.2 Renaming column names to acceptable Python string formats

5.3.5.4 Saved the updated data frame as a CSV file.

```
Saving the new updated data frame
```

```
[ ] # Saving the updated data frame to csv.
    # This data set now has crime recods and
    # yearly GDP value from 2005 to 2019

    df.to_csv('/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Crimes_Reported_from_2005_to_2019_with_Corresponding_annual_GDP.csv')
```

Fig.51.3 Saving the new data frame as a CSV file containing

	Year	Type_of_Crime	Number_of_Crimes_Reported	Gross_Domestic_Product_GDP
0	2005-01-01	Homicide	2313	5.9
1	2005-01-01	Offences against morality	3153	5.9
2	2005-01-01	Other offences against persons	17304	5.9
3	2005-01-01	Robbery	6936	5.9
4	2005-01-01	Breakings	8454	5.9
5	2005-01-01	Theft of stock	2219	5.9
6	2005-01-01	Stealing	12589	5.9
7	2005-01-01	Theft by servant	2874	5.9
8	2005-01-01	Vehicles and other thefts	1718	5.9
9	2005-01-01	Dangerous drugs	6356	5.9
10	2005-01-01	Traffic offences	38	5.9
11	2005-01-01	Criminal damage	3236	5.9
12	2005-01-01	Economic crimes	1390	5.9
13	2005-01-01	Corruption	107	5.9
14	2005-01-01	Offences involving police officers	29	5.9
15	2005-01-01	Offences involving tourists	32	5.9
16	2005-01-01	Other penal code offences	6652	5.9
17	2006-01-01	Homicide	2090	6.3
18	2006-01-01	Offences against morality	3525	6.3
19	2006-01-01	Other offences against persons	18723	6.3
20	2006-01-01	Robbery	5234	6.3
21	2006-01-01	Breakings	7420	6.3
22	2006-01-01	Theft of stock	2209	6.3
23	2006-01-01	Stealing	10874	6.3

Fig.51.4 Part of the data set containing crimes recorded from 2005 to 2019 with dates on the 'Year' column and GDP values of the different years

5.4 Exploratory Data Analysis

After creating and updating the new data set I proceeded to explore the features in the data set. I used graphs to discover the trends from 2005 to 2019 in the Number_of_Crimes_Reported and 'Gross_Domestic_Product_GDP'. I then checked the correlation between the two main features in the data set which were the Number_of_Crimes_Reported and 'Gross_Domestic_Product_GDP'.

5.4.1 Loading the data set, reading it as data fame, and indexing the 'Year' column.

I loaded the data set through its path where I had saved it then read it as a data frame. I then proceeded to make the 'Year' column be the index column.

```
#loading the dataset
pathdataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Crimes_Reported_from_2005_to_2019_with_Corresponding_annual_GDP.csv'

df = pd.read_csv(pathdataset, parse_dates=['Year'],
                 index_col=['Year'],)
```

Fig.52 Loading the data set to begin data exploration

5.4.2 Studying the trend in the Gross_Domestic_Product_GDP

Plotting the 'Gross_Domestic_Product_GDP' on the y-axis, I studied the trend in this metric, GDP, and discovered that, it was increasing then took a huge drop in 2008. The GDP would then steadily increase up to 2010, then maintain constant value changes between 4 and 6 from 2011 up to 2018 onwards. This would show that the country's GDP has been constantly on a low mark with very little improvement.

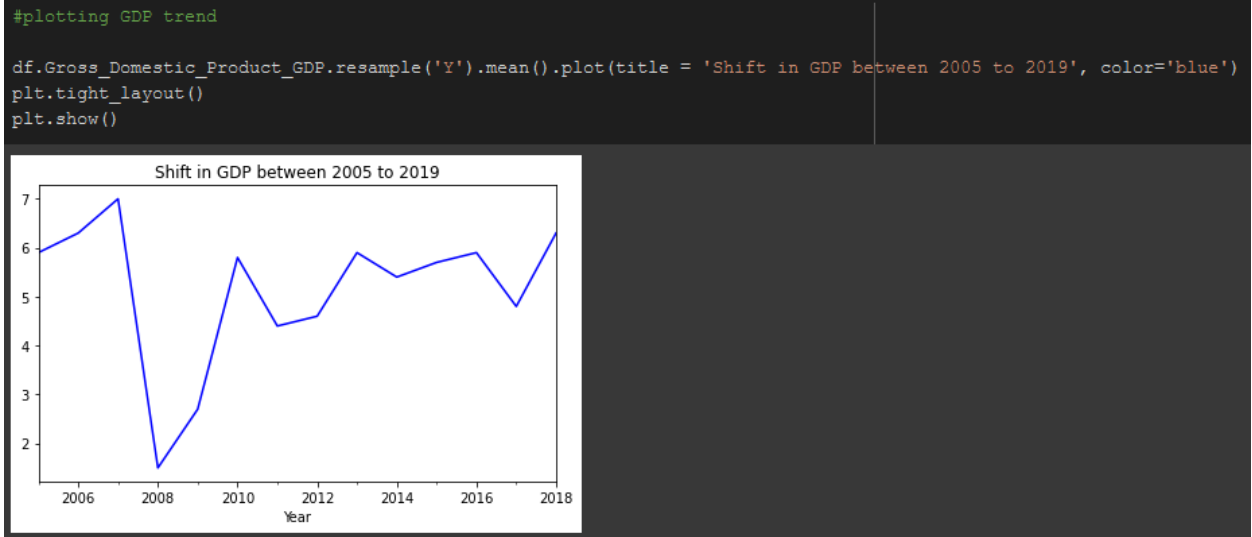


Fig.53 Trend in GDP between 2005 and 2019

5.4.3 Studying the trend in the Number_of_Crimes_Reported

After analyzing the pattern in GDP over years, I proceeded to discover the pattern in Number_of_Crimes_Reported on an average over years. Averagely, crime has been on an increase over the decade.

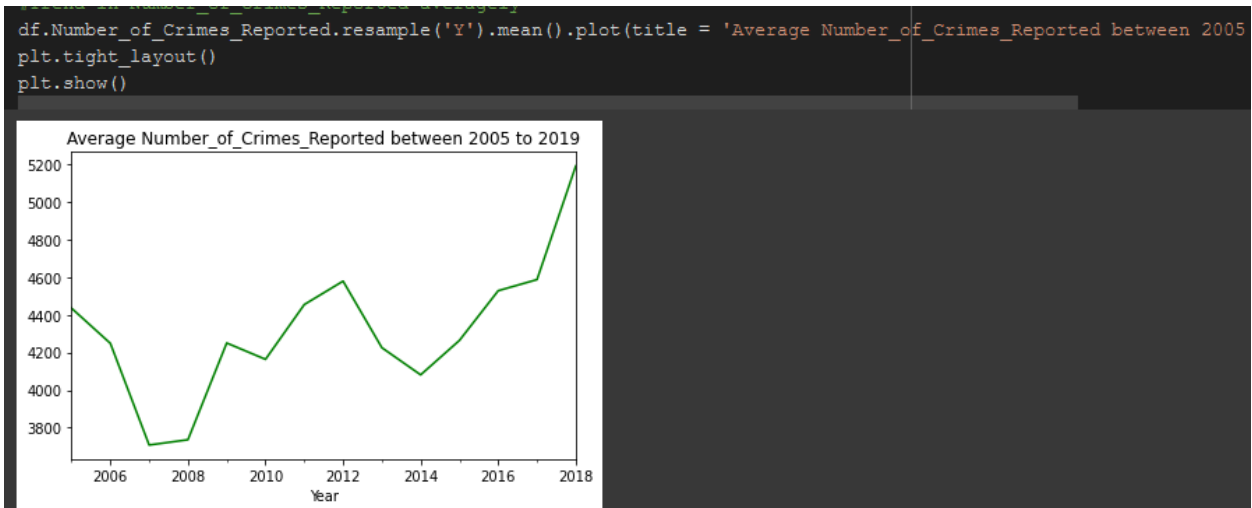


Fig.54 Trend in Number of crimes reported averagely between 2005 and 2019

5.5 Modeling

The fifth phase was developing a Machine learning model that would predict the number of crimes that will be reported for different types of crime. The use of Recurrent Neural Networks, which form the foundation of deep learning, was employed in this phase. The algorithm I used was the Long Short-Term Memory (LSTM) which is an artificial RNN. I chose to use the LSTM algorithm since this algorithm is convenient for time-series predictions or forecasting like in our case where the aim was to predict the number of crimes that should be expected soon.

The steps which I took in this phase were:

1. Data preparation for LSTM
2. One Hot Encoding
3. Normalizing the attributes using MinMax Scalar
4. Framing the problem into a Supervised Learning problem
5. Removing the columns not to be predicted
6. Splitting the data set into Train and Test set
7. Design and fitting the neural network
8. Plotting the loss chart

5.5.1 Data Preparation for LSTM

I first imported the required packages from Python which included:

- `matplotlib.pyplot` – To enable the use of Python plot functions that would allow the plotting of lines and labels to create figures too.
- `read_csv` and `get_dummies` – To read CSV files and create dummy data frames respectively.
- `MinMaxScaler` – To change the values of features by scaling the values into a specific range.
- `mean_squared_error` – To calculate the average squared difference that would be between the actual values and the estimated values.
- `Sequential` – API that would let us build our model layer-by-layer.

Data Preperation for LSTM

```
[ ] #importing required packages
    from math import sqrt
    from numpy import concatenate
    import matplotlib.pyplot as plt
    from pandas import read_csv, get_dummies
    from pandas import DataFrame
    from pandas import concat
    from sklearn.preprocessing import MinMaxScaler
    from sklearn.metrics import mean_squared_error
    from keras.models import Sequential
    from keras.layers import Dense
    from keras.layers import LSTM

    # convert series to supervised learning
    def series_to_supervised(data, n_in=1, n_out=1, dropnan=True):
        n_vars = 1 if type(data) is list else data.shape[1]
        df = DataFrame(data)
        cols, names = list(), list()

        # input sequence (t-n, ... t-1)
        for i in range(n_in, 0, -1):
            cols.append(df.shift(i))
            names += [('var%d(t-%d)' % (j+1, i)) for j in range(n_vars)]

        # forecast sequence (t, t+1, ... t+n)
        for i in range(0, n_out):
            cols.append(df.shift(-i))
            if i == 0:
                names += [('var%d(t)' % (j+1)) for j in range(n_vars)]
            else:
                names += [('var%d(t+%d)' % (j+1, i)) for j in range(n_vars)]

        # put it all together
        agg = concat(cols, axis=1)
        agg.columns = names

        # drop rows with NaN values
        if dropnan:
            agg.dropna(inplace=True)
        return agg
```

Fig. 55 LSTM data preparation

The aim is to build a model that predicts the Number of Crimes Reported.

1. I first load the dataset that is intended to be used

```
[ ] import pandas as pd

# load dataset
pathDataset = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Crimes_Reported_from_2005_to_2019_with_Corresponding_annual_GDP.csv'
dataset = pd.read_csv(pathDataset, header=0, index_col=0)
dataset.head()
```

	Year	Type_of_Crime	Number_of_Crimes_Reported	Gross_Domestic_Product_GDP
0	2005-01-01	Homicide	2313	5.9
1	2005-01-01	Offences against morality	3153	5.9
2	2005-01-01	Other offences against persons	17304	5.9
3	2005-01-01	Robbery	6936	5.9
4	2005-01-01	Breakings	8454	5.9

Fig. 55.1 Load and read the data set to be used

2. Make 'Year' column to be index of the data set

```
[ ] df_index = pd.read_csv(pathDataset, parse_dates=['Year'], index_col=['Year'])
df_index.to_csv('/content/drive/MyDrive/colab_notebooks/indexed_dataset.csv')

[ ] df_index = pd.read_csv('/content/drive/MyDrive/colab_notebooks/indexed_dataset.csv', header=0, index_col=0)
df_index.columns

Index(['Unnamed: 0', 'Type_of_Crime', 'Number_of_Crimes_Reported',
      'Gross_Domestic_Product_GDP'],
      dtype='object')

[ ] df_index.shape

(238, 4)
```

Fig. 55.2 Make the 'Year' column the index of the data set to be used, naming and saving this new data set created as indexed_dataset.csv

Reorder column - Number of Crimes Reported - made as dependent

- I will then reorder the columns in the dataset to make the "Number of Crimes Reported" column to be a dependent variable ... It will be dependent of the GDP

```
[ ] cols = ['Number_of_Crimes_Reported', 'Gross_Domestic_Product_GDP', 'Type_of_Crime']
dataset = dataset.reindex(columns=cols)
dataset.columns

Index(['Number_of_Crimes_Reported', 'Gross_Domestic_Product_GDP',
      'Type_of_Crime'],
      dtype='object')

[ ] dataset.shape

(238, 3)
```

Fig. 55.3 Reorder the dataset columns to have the Number_of_Crimes_Reported be the first column, dependent on GDP

5.5.2 One-Hot Encoding

After reordering the data sets columns, I then did a One-Hot Encoding on the 'Type_of_Crime' column. This led to the creation of a dummy data set in which I created columns named after each type of crime that was recorded, merging this with the string 'Type_of_Crime'.

For example, the types of crime recorded were Homicide and Breakings. The new columns created in the dummy data set would therefore be 'Type_of_Crime_Homicide' and 'Type_of_Crime_Breakings', and so on for all the other types of crime.

Having created the new columns, this technique would then add values into the different columns as 1s and 0s. 1s value in the newly created columns were entries in which the type of crime appeared, and 0s where they did not appear. This was to enable numeric inputs to be entered into the model as 1s and 0s, also acknowledged by the fact that neural networks take in numeric values as inputs.

```
One Hot encoding Type_of_Crime column

One hot encoding is the process of transforming a column of string d_type to columns of 1s and 0s; since most predictive models take as input numerical data

[ ] # generate binary values using get_dummies
    dum_df = get_dummies(dataset, columns=["Type_of_Crime"])# merge with main df bridge_df on key values
    #dataset = dataset.merge(dum_df)

    # dataset
    dum_df
```

Fig. 56 One Hot Encoding

	Year	Number_of_Crimes_Reported	Gross_Domestic_Product_GDP	Type_of_Crime_Breakings	Type_of_Crime_Corruption	Type_of_Crime_Criminal damage	Type_of_Crii damage
0	2005-01-01	2313	5.9	0	0	0	
1	2005-01-01	3153	5.9	0	0	0	
2	2005-01-01	17304	5.9	0	0	0	
3	2005-01-01	6936	5.9	0	0	0	
4	2005-01-01	8454	5.9	1	0	0	
...
233	2018-01-01	4100	6.3	0	0	0	
234	2018-01-01	119	6.3	0	1	0	
235	2018-01-01	174	6.3	0	0	0	
236	2018-01-01	93	6.3	0	0	0	
237	2018-01-01	7953	6.3	0	0	0	

Fig. 56.1 Dummy data frame's snap after One Hot Encoding

```

#Renaming columns
dataset.rename(columns = {
    'Type_of_Crime_Criminal damage':'Type_of_Crime_Criminal_damage',
    'Type_of_Crime_Dangerous drugs':'Type_of_Crime_Dangerous_drugs',
    'Type_of_Crime_Economic crimes':'Type_of_Crime_Economic_crimes' ,
    'Type_of_Crime_Offences against morality':'Type_of_Crime_Offences_against_morality',
    'Type_of_Crime_Offences involving police officers':'Type_of_Crime_Offences_involving_police_officers',
    'Type_of_Crime_Offences involving tourists':'Type_of_Crime_Offences_involving_tourists',
    'Type_of_Crime_Other offences against persons':'Type_of_Crime_Other_offences_against_persons',
    'Type_of_Crime_Other penal code offences':'Type_of_Crime_Other_penal_code_offences',
    'Type_of_Crime_Theft by servant':'Type_of_Crime_Theft_by_servant',
    'Type_of_Crime_Theft of stock':'Type_of_Crime_Theft_of_stock',
    'Type_of_Crime_Traffic offences':'Type_of_Crime_Traffic_offences',
    'Type_of_Crime_Vehicles and other thefts':'Type_of_Crime_Vehicles_and_other_thefts'
}, inplace =True)

dataset.columns

Index(['Year', 'Number_of_Crimes_Reported', 'Gross_Domestic_Product_GDP',
      'Type_of_Crime_Breakings', 'Type_of_Crime_Corruption',
      'Type_of_Crime_Criminal_damage', 'Type_of_Crime_Dangerous_drugs',
      'Type_of_Crime_Economic_crimes', 'Type_of_Crime_Homicide',
      'Type_of_Crime_Offences_against_morality',
      'Type_of_Crime_Offences_involving_police_officers',
      'Type_of_Crime_Offences_involving_tourists',
      'Type_of_Crime_Other_offences_against_persons',
      'Type_of_Crime_Other_penal_code_offences', 'Type_of_Crime_Robbery',
      'Type_of_Crime_Stealing', 'Type_of_Crime_Theft_by_servant',
      'Type_of_Crime_Theft_of_stock', 'Type_of_Crime_Traffic_offences',
      'Type_of_Crime_Vehicles_and_other_thefts'])

```

Fig. 56.2 Renaming the dummy data frame's columns into Python's acceptable string format by adding underscores to cover white spaces after One Hot Encoding

5.5.2.1 Saving the data set as a new CSV file

I then saved the dummy dataset as a Final data set in CSV format. This was going to be the data set that I used in model training and testing

Notice that the different types of crimes that were created were now of integer types, which would be acceptable by the neural networks since they accept data inputs as numerical.

```

pathFinal = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/'
dataset.to_csv(pathFinal + 'Final_1_Crimes_Recorded_from_2005_to_2019_with_corresponding_Yearly_GDP_values.csv')

import pandas as pd
pathFinal = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/'
df = pd.read_csv(pathFinal + 'Final_1_Crimes_Recorded_from_2005_to_2019_with_corresponding_Yearly_GDP_values.csv',
                 parse_dates = ['Year'], index_col = ['Year'])

df.dtypes

Unnamed: 0                                int64
Number_of_Crimes_Reported                 int64
Gross_Domestic_Product_GDP               float64
Type_of_Crime_Breakings                  int64
Type_of_Crime_Corruption                 int64
Type_of_Crime_Criminal_damage            int64
Type_of_Crime_Dangerous_drugs            int64
Type_of_Crime_Economic_crimes            int64
Type_of_Crime_Homicide                  int64
Type_of_Crime_Offences_against_morality  int64
Type_of_Crime_Offences_involving_police_officers int64
Type_of_Crime_Offences_involving_tourists int64
Type_of_Crime_Other_offences_against_persons int64
Type_of_Crime_Other_penal_code_offences  int64
Type_of_Crime_Robbery                   int64
Type_of_Crime_Stealing                   int64
Type_of_Crime_Theft_by_servant           int64
Type_of_Crime_Theft_of_stock             int64
Type_of_Crime_Traffic_offences           int64
Type_of_Crime_Vehicles_and_other_thefts  int64

```

Fig. 56.2 Saving the dummy data set – the data set that would be used for training and testing our model

5.5.3 Normalizing the attributes

Normalizing is the process of transforming the features' values which are numeric in a given data set so that the values be in a specific range. Normalization is mostly achieved using a scalar. For example, in this case, the values in the 'Number_of_Crimes_Reported' column, which is a feature in our dataset, were normalized. Its values were transformed to be in a range between 0 and 1.

5.5.3.1 Normalizing the attributes using the MinMax Scaler

Before normalizing the features' values in the data frame, I first converted all the values in the data frame into float data types.

```

[ ] values = df.values
    # ensure all data is float
    values = values.astype('float32')
    # normalize features
    scaler = MinMaxScaler(feature_range=(0, 1))
    scaled = scaler.fit_transform(values)

```

Fig. 57 Normalizing the data frame's features' values to a range between 0 and 1

5.5.3.2 Saving the Scaler

Having done the normalization, I then saved the scaler.

Saving the Scaler

```
[ ] import joblib
    pathScaler = '/content/drive/MyDrive/colab_notebooks/Crimes_Recorded.pkl'

    joblib.dump(scaler, pathScaler)

['/content/drive/MyDrive/colab_notebooks/Crimes_Recorded.pkl']
```

Fig. 57.1 Saving the Scaler

5.5.4 Framing the problem into a Supervised Learning problem

I then went ahead to frame the problem into a supervised learning problem with the big question being:

Given the measure of Economic growth as a GDP value at a given time, what will be the predicted number of crimes for different types of crime at a time step (t), following the last time step (t-1) which depicts the previous records?

```
reframed = series_to_supervised(scaled, 1, 1)
reframed.columns

Index(['var1(t-1)', 'var2(t-1)', 'var3(t-1)', 'var4(t-1)', 'var5(t-1)',
      'var6(t-1)', 'var7(t-1)', 'var8(t-1)', 'var9(t-1)', 'var10(t-1)',
      'var11(t-1)', 'var12(t-1)', 'var13(t-1)', 'var14(t-1)', 'var15(t-1)',
      'var16(t-1)', 'var17(t-1)', 'var18(t-1)', 'var19(t-1)', 'var1(t)',
      'var2(t)', 'var3(t)', 'var4(t)', 'var5(t)', 'var6(t)', 'var7(t)',
      'var8(t)', 'var9(t)', 'var10(t)', 'var11(t)', 'var12(t)', 'var13(t)',
      'var14(t)', 'var15(t)', 'var16(t)', 'var17(t)', 'var18(t)', 'var19(t)'],
      dtype='object')
```

Fig. 58 Framing the problem into supervised learning

5.5.5 Removing the columns not to be predicted

After removing the columns not to be predicted, the resulting output will be 19 input variables and 1 output variable which will be Number_of_Crimes. The output variable will var1(t), the dependent variable which we want to predict.

```
#Columns to be dropped
reframed.columns[20:]

Index(['var2(t)', 'var3(t)', 'var4(t)', 'var5(t)', 'var6(t)', 'var7(t)',
      'var8(t)', 'var9(t)', 'var10(t)', 'var11(t)', 'var12(t)', 'var13(t)',
      'var14(t)', 'var15(t)', 'var16(t)', 'var17(t)', 'var18(t)', 'var19(t)'],
      dtype='object')

#dropping the displayed columns -- columns not to be predicted
reframed.drop(reframed.columns[20:], axis=1, inplace=True)

#display the remainig columns -- columns to predict
print(reframed.head())
```

	var1(t-1)	var2(t-1)	var3(t-1)	...	var18(t-1)	var19(t-1)	var1(t)
1	0.092339	0.8	0.0	...	0.0	0.0	0.125873
2	0.125873	0.8	0.0	...	0.0	0.0	0.690806
3	0.690806	0.8	0.0	...	0.0	0.0	0.276897
4	0.276897	0.8	0.0	...	0.0	0.0	0.337499
5	0.337499	0.8	1.0	...	0.0	0.0	0.088586

```
[5 rows x 20 columns]
```

Fig. 59 Dropping columns not to be predicted

5.5.6 Splitting the data set into Train and Test set

Our data set had 14 years' worth of data. I decided to split this data set into 13 years for training the model and the remaining one year for testing the model

```

values = reframed.values

# Setting the train dataset to be the first 13 years of data
n_train_years = 13*17

# split into train and test sets
train = values[:n_train_years, :]
test = values[n_train_years:, :]

# split into input and outputs
train_X, train_y = train[:, :-1], train[:, -1]
test_X, test_y = test[:, :-1], test[:, -1]

# reshape input to be 3D [samples, timesteps, features]
train_X = train_X.reshape((train_X.shape[0], 1, train_X.shape[1]))
test_X = test_X.reshape((test_X.shape[0], 1, test_X.shape[1]))

print(train_X.shape, train_y.shape, test_X.shape, test_y.shape)

(221, 1, 19) (221,) (16, 1, 19) (16,)

```

Fig. 59 Split dataset to train and test

5.5.7 Design and Fitting the Neural Network Model

I trained the model with 200 epochs each with a batch size of 7 and set an early stopping when the model's loss value did not improve. Training would continue for a further 20 steps if the model's loss did not improve to be certain that the best-trained model has been obtained.

In this case, training at *Epoch 00056* recoded the best value loss of *0.03991*. Before saving at this point, training proceeded twenty more times to get any improvement in the value loss but there was none. Early stopping was thus invoked and training completed.

Fitting was done to ensure the model does not overfit or underfit, thereby preventing erroneous output when input data will be presented to the model for testing or prediction.

```

from keras.layers import Dropout

# Initialising the RNN
model = Sequential()
model.add(LSTM(units = 50, input_shape=(train_X.shape[1], train_X.shape[2])))

# Adding a dropout of 20% to minimize overfitting
model.add(Dropout(0.2))

# Adding the output layer
# For Full connection layer I use dense, with unit=1 since output is 1D
# I use softplus activation, since I want the output to be only positive values (0 to +ve inf)
model.add(Dense(1, activation='softplus'))

#compiling the model
model.compile(loss='mae', optimizer='adam')

from keras.callbacks import EarlyStopping, ModelCheckpoint

pathModel = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Predicting_Number_of_Crimes_model'

# Create callbacks -- EarlyStopping, ModelCheckpoint

# EarlyStopping callback with patience
es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=20)

# ModelCheckpoint callback, for saving best model
mc = ModelCheckpoint(pathModel, monitor='val_loss', mode='min', verbose=1, save_best_only=True)

# fitting the network model
history = model.fit(train_X, train_y, epochs= 200, batch_size=7,
                    validation_data=(test_X, test_y), verbose=2, shuffle=False,
                    callbacks=[es, mc])

```

```

Epoch 1/200

Epoch 00001: val_loss improved from inf to 0.41936, saving model to /content/drive/MyDrive/Colab Notebooks/School P
32/32 - 3s - loss: 0.4880 - val_loss: 0.4194 - 3s/epoch - 108ms/step
Epoch 2/200

Epoch 00002: val_loss improved from 0.41936 to 0.32103, saving model to /content/drive/MyDrive/Colab Notebooks/Scho
32/32 - 0s - loss: 0.3981 - val_loss: 0.3210 - 138ms/epoch - 4ms/step
Epoch 3/200

Epoch 00003: val_loss improved from 0.32103 to 0.21452, saving model to /content/drive/MyDrive/Colab Notebooks/Scho
32/32 - 0s - loss: 0.2959 - val_loss: 0.2145 - 165ms/epoch - 5ms/step
Epoch 4/200

Epoch 00004: val_loss improved from 0.21452 to 0.15547, saving model to /content/drive/MyDrive/Colab Notebooks/Scho
32/32 - 0s - loss: 0.1979 - val_loss: 0.1555 - 184ms/epoch - 6ms/step

```

Fig. 60 Display of designing and fitting our model, and the start of model training

```
Epoch 00069: val_loss did not improve from 0.03899
32/32 - 0s - loss: 0.0225 - val_loss: 0.0395 - 127ms/epoch - 4ms/step
Epoch 70/200

Epoch 00070: val_loss did not improve from 0.03899
32/32 - 0s - loss: 0.0241 - val_loss: 0.0400 - 127ms/epoch - 4ms/step
Epoch 71/200

Epoch 00071: val_loss did not improve from 0.03899
32/32 - 0s - loss: 0.0242 - val_loss: 0.0403 - 131ms/epoch - 4ms/step
Epoch 72/200

Epoch 00072: val_loss did not improve from 0.03899
32/32 - 0s - loss: 0.0231 - val_loss: 0.0449 - 108ms/epoch - 3ms/step
Epoch 73/200

Epoch 00073: val_loss did not improve from 0.03899
32/32 - 0s - loss: 0.0230 - val_loss: 0.0445 - 109ms/epoch - 3ms/step
Epoch 74/200

Epoch 00074: val_loss did not improve from 0.03899
32/32 - 0s - loss: 0.0236 - val_loss: 0.0430 - 117ms/epoch - 4ms/step
Epoch 75/200

Epoch 00075: val_loss did not improve from 0.03899
32/32 - 0s - loss: 0.0213 - val_loss: 0.0439 - 126ms/epoch - 4ms/step
Epoch 76/200

Epoch 00076: val_loss did not improve from 0.03899
32/32 - 0s - loss: 0.0228 - val_loss: 0.0433 - 126ms/epoch - 4ms/step
Epoch 00076: early stopping
```

Fig. 61 Display of part of the 20 more steps in model training before reaching the early stopping

5.5.8 Plotting the Loss Chart

We can notice that our model was trained successfully and plotting the graph below, to validate the model, I can conclude that the model's performance is good since after training and testing was done and the training and testing matched using a line graph, the two outcomes were very close.


```

] import matplotlib.pyplot as plt

# plotting the history
plt.plot(history.history['loss'], label='train')
plt.plot(history.history['val_loss'], label='test')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.grid(True)
plt.legend()
plt.show()

```

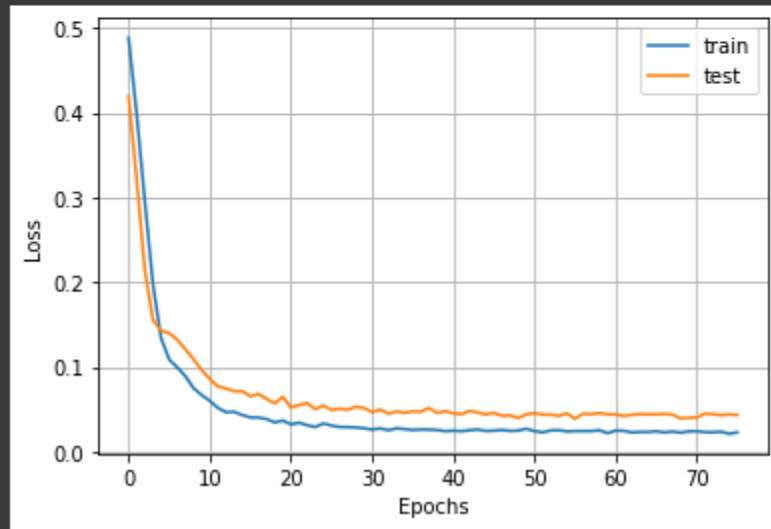


Fig. 62 Display of model's value loss after training and testing was done

5.6 Model Evaluation

To evaluate the model, I first loaded the saved model then tested it to see if it could predict values.

```

#load model
from tensorflow.keras.models import load_model

pathModel = '/content/drive/MyDrive/Colab Notebooks/School Project docs and files/Predicting_Number_of_Crimes_model.h5'

model = load_model(pathModel)

```

Fig. 63.1 Loading the model

```

#making a prediction
yhat = model.predict(test_X)

#reshaping test_X
test_x = test_X.reshape((test_X.shape[0], test_X.shape[2]))

#invert scaling for forecast
inv_yhat = concatenate((yhat, test_x[:, 1:]), axis=1)
inv_yhat = scaler.inverse_transform(inv_yhat)
inv_yhat = inv_yhat[:,0]

#invert scaling for actual
test_y = test_y.reshape((len(test_y), 1))
inv_y = concatenate((test_y, test_x[:, 1:]), axis=1)
inv_y = scaler.inverse_transform(inv_y)
inv_y = inv_y[:,0]

```

Fig. 63.2 Making a prediction

```

#Type casting the predictions to int type
inv_yhat = inv_yhat.astype(int)

#Displaying Predicted values
print('The values that are predicted are: ')
print(inv_yhat)

The values that are predicted are:
[ 4786 20988  3034  6666  2109 11805  2529  1404  5470   121  3653  2850
    137   126   125  5906]

```

Fig. 63.3 Display of predicted values

5.6.1 Evaluating the model using Root Mean Squared Error (RMSE)

I then evaluated the model using Root Mean Square Error which is used as a heuristic to evaluate trained models for their accuracy and functionality too. I managed to tune the model's hyper-parameters through which I obtained a low RMSE of 1533.166. This was achieved by using the vanilla LSTM with a batch size of 7, after training for 200 epochs.

Normalizing the RMSE in the range 0 to 1 yielded an NRMSE of 0.062, which is low, depicting that the model's performance was excellent.

To get the model's accuracy, I obtained the difference between 1 and the normalized RMSE obtained. The difference was 0.938 translating to an accuracy of 93.8 % which can be concluded as accuracy of 94 %.

```
#calculating the RMSE
rmse = sqrt(mean_squared_error(inv_y, inv_yhat))
print('Root Mean Square Error: %.3f' % rmse)

#Normalizing the RMSE using range
nrmse = rmse/ (inv_y.max() - inv_yhat.min())
print('Normalized Root Mean Square Error: %.3f' %nrmse)

#Accuracy of model
accuracy = 1-nrmse
print("Accuracy of model is: %.3f " % accuracy)

Root Mean Square Error: 1533.166
Normalized Root Mean Square Error: 0.062
Accuracy of model is: 0.938
```

Fig. 63.4 Model's accuracy using RMSE

5.6.2 Plotting Actual Values against Forecast values

```
import matplotlib.pyplot as plt
plt.plot(inv_y, color = 'blue', label = 'Actual')
plt.plot(inv_yhat, color = 'red', label = 'Predicted')
plt.title('Number of Crimes Reported')
plt.xlabel('Type_of_Crime Columns')
plt.ylabel('Number_of_Crimes_Reported')
plt.legend()
plt.show()
```

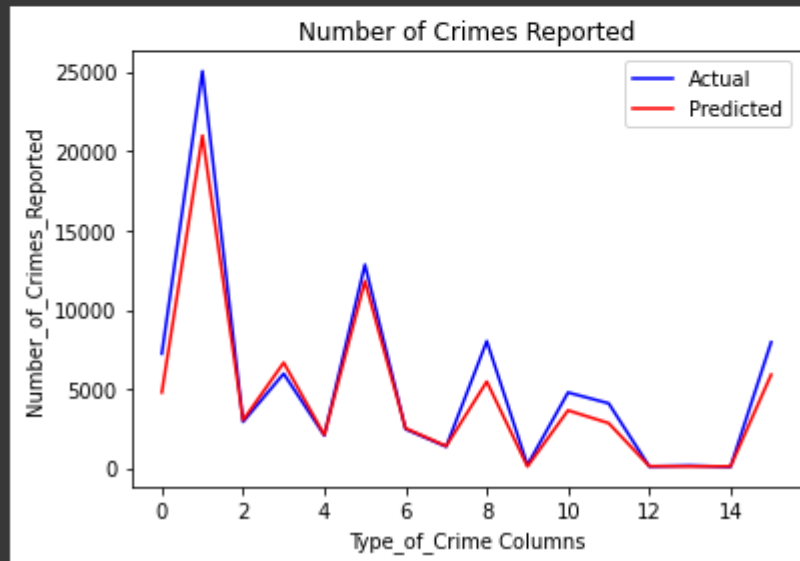


Fig. 63.5 Actual against Predicted values

5.7 Model Deployment

I first installed the dependencies I need to use like flask and gunicorn. I then created a web app interface that accepts a data set with criminal records and the year's GDP value.

For the interface's frontend, I used HTML and CSS then used Python for the backend.

```

C:\WINDOWS\System32\WindowsPowerShell\v1.0\powershell.exe
(deployment) PS C:\Users\Okite Eric> cd E:\deployment
(deployment) PS E:\deployment> flask run
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
2022-01-14 08:30:09.221893: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with
  oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations:
  AVX AVX2
  To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [14/Jan/2022 08:31:09] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [14/Jan/2022 08:31:10] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [14/Jan/2022 08:31:54] "POST / HTTP/1.1" 302 -
127.0.0.1 - - [14/Jan/2022 08:31:54] "GET /prediction/static/files%5CCrimes_and_GDP_2018_dataset.csv HTTP/1.1" 200 -
127.0.0.1 - - [14/Jan/2022 09:17:29] "GET /prediction/static/files%5CCrimes_and_GDP_2018_dataset.csv HTTP/1.1" 200 -
127.0.0.1 - - [14/Jan/2022 09:17:37] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [14/Jan/2022 09:17:47] "POST / HTTP/1.1" 302 -
127.0.0.1 - - [14/Jan/2022 09:17:47] "GET /prediction/static/files%5CCrimes_and_GDP_2018_dataset.csv HTTP/1.1" 200 -

```

Fig. 63.6 Anaconda's PowerShell- prompt used to install the flask for model deployment, then start the model

```

Terminal  Help  index.html - deployment - Visual Studio Code
1  index.html  prediction.html
s > index.html > html > head > style > .navbar
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Predictive Analysis of Number of Crimes Based on Economic Growth in Kenya</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
  <style>
    /* Remove the navbar's default margin-bottom and rounded borders */
    .navbar {
      margin-bottom: 0;
      border-radius: 0;
    }

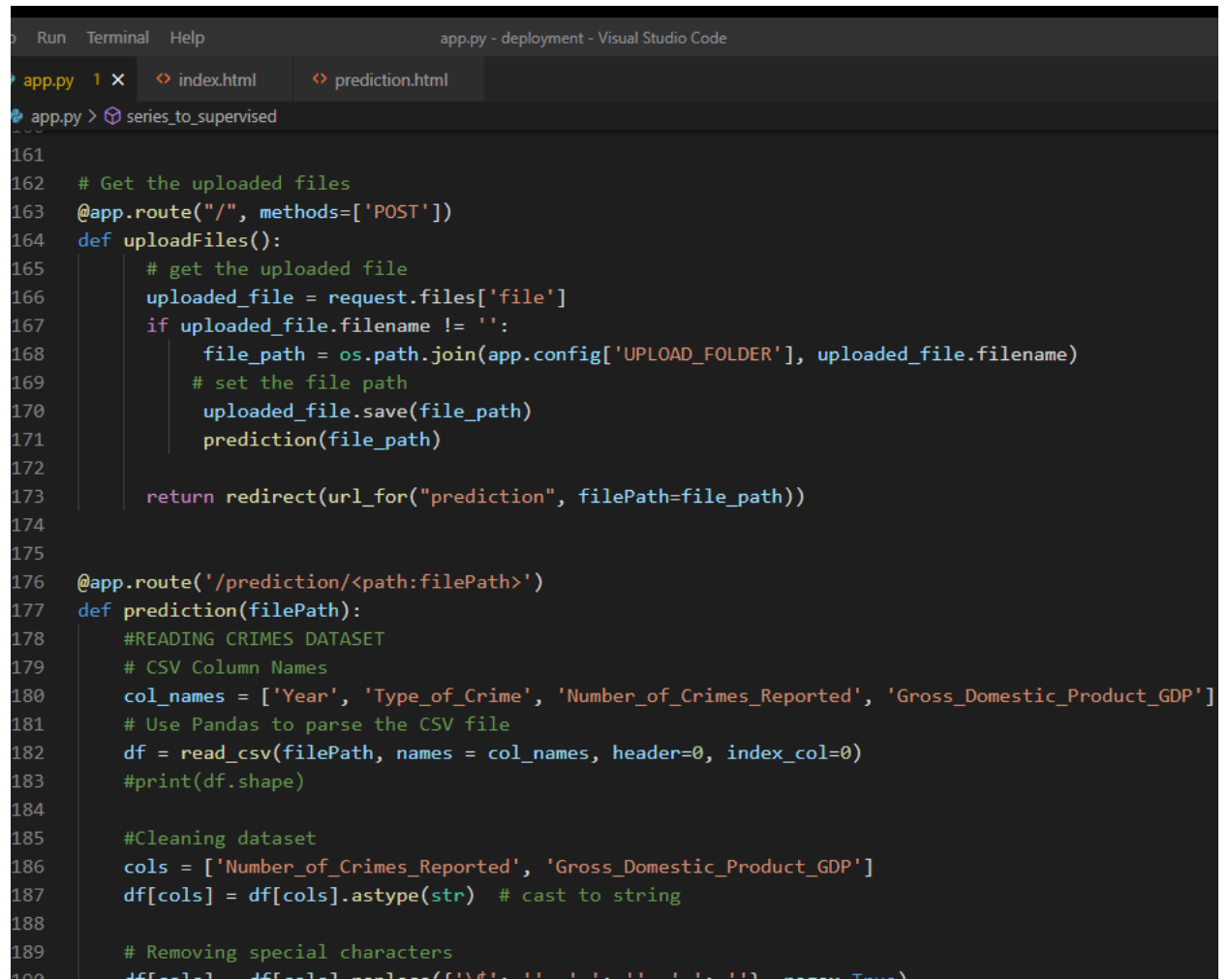
    /* Set height of the grid so .sidenav can be 100% (adjust as needed) */
    .row.content {height: 450px}

    /* Set gray background color and 100% height */
    .sidenav {
      padding-top: 20px;
      background-color: #f1f1f1;
      height: 100%;
    }

    /* Set black background color, white text and some padding */
    footer {
      background-color: #555;
      color: white;

```

Fig. 63.7 Frontend - Index code snippet of the model's web app



```

161
162 # Get the uploaded files
163 @app.route("/", methods=['POST'])
164 def uploadFiles():
165     # get the uploaded file
166     uploaded_file = request.files['file']
167     if uploaded_file.filename != '':
168         file_path = os.path.join(app.config['UPLOAD_FOLDER'], uploaded_file.filename)
169         # set the file path
170         uploaded_file.save(file_path)
171         prediction(file_path)
172
173     return redirect(url_for("prediction", filePath=file_path))
174
175
176 @app.route('/prediction/<path:filePath>')
177 def prediction(filePath):
178     #READING CRIMES DATASET
179     # CSV Column Names
180     col_names = ['Year', 'Type_of_Crime', 'Number_of_Crimes_Reported', 'Gross_Domestic_Product_GDP']
181     # Use Pandas to parse the CSV file
182     df = read_csv(filePath, names = col_names, header=0, index_col=0)
183     #print(df.shape)
184
185     #Cleaning dataset
186     cols = ['Number_of_Crimes_Reported', 'Gross_Domestic_Product_GDP']
187     df[cols] = df[cols].astype(str) # cast to string
188
189     # Removing special characters
190     df[cols] = df[cols].replace({'/': '\\', ' ': ' ', ' ': ' '}, regex=True)

```

Fig. 63.8 Backend code snippet of the model's web app

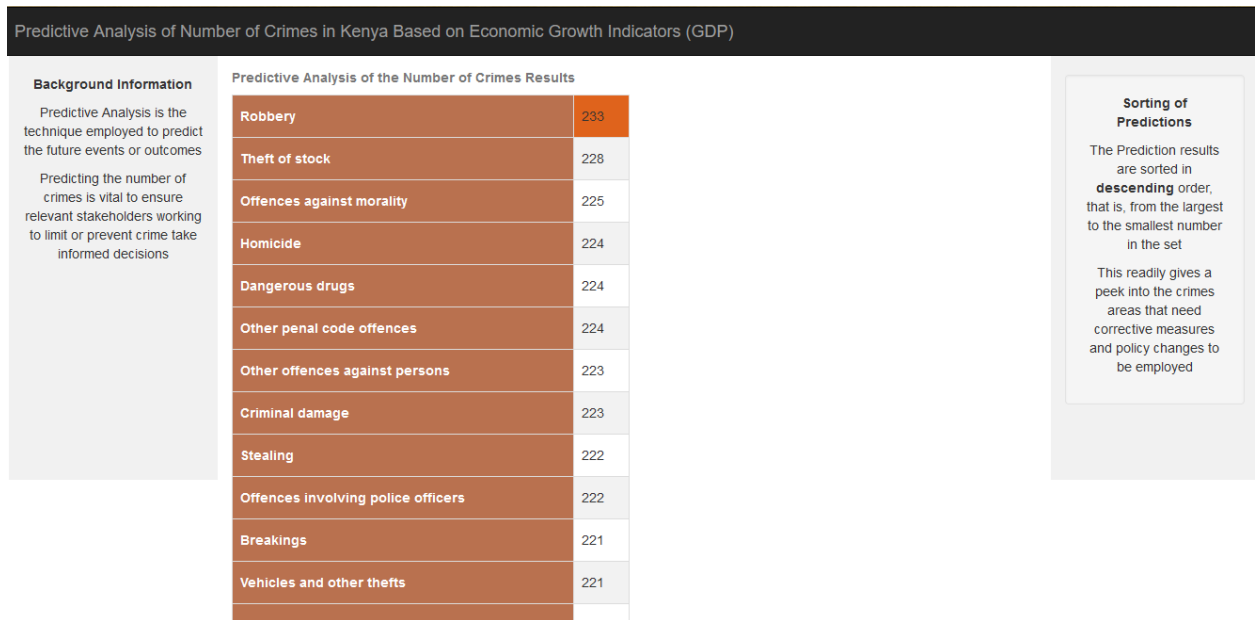


Fig. 63.9 Different types of Crimes Predictions made on the Web app interface of the model

CHAPTER 6: CONCLUSION, CHALLENGES, AND RECOMMENDATIONS

6.1 Conclusion

The model works as expected. The predictions on the different types of crime through numbers are made. These predictions made are what is expected in 2019 according to the assessment. According to the model, the number of crimes for the different types of crime appears to be averagely low. This is because the metric that was used to achieve the predictions was only one, Economic growth, through its indicator Gross Domestic Product. GDP.

6.2 Challenges faced during the Project

- Getting a data set with all crime metrics like age, gender, education, and employment was a challenge through this project. I, therefore, had to prepare data sets to achieve the task.
- Extracting these data sets from the Statistical Abstracts for their preparation was a challenge too which led to utilizing more time that would have been used in developing the model being spent in the data preparation part.

6.3 Project Recommendations

I recommend that datasets with more crime metrics should be made available to enable the exploration of various aspects of crime. This would enable the model to give more accurate predictions once an input data set is uploaded since the model will be built based on many crime variables.

It is my hope that this project will be considered to help tackle crime in different places in our country Kenya and the nation at large to ensure peaceful environment exist throughout, for safer living alongside scaling the heights to acquiring better living standards through by providing conducive environments for productivity and economic growth.

CHAPTER 7: REFERENCES

- [1] Biermann F, Kanie N, Kim RE. Global governance by goal-setting: the novel approach of the UN Sustainable Development Goals. *Curr Opin Environ Sustain* 2017;26–27:26–31. <https://doi.org/10.1016/J.COSUST.2017.01.010>.
- [2] Barbier EB, Burgess JC. The sustainable development goals and the systems approach to sustainability. *Economics* 2017;11:1–22. https://doi.org/10.5018/ECONOMICS-EJOURNAL.JA.2017-28/DOWNLOADASSET/SUPPL/JOURNALARTICLES_2017-28-SM5.PDF.
- [3] Crime Victims: Theory, Policy and Practice - Basia Spalek - Google Books n.d. https://books.google.co.ke/books?hl=en&lr=&id=1csEDgAAQBAJ&oi=fnd&pg=PP1&dq=what+is+crime+according+to+victim+support&ots=08UljjgR96&sig=j5PHhXsdrVbD59T5H4-E-5L0hwx&redir_esc=y#v=onepage&q=what is crime according to victim support&f=false (accessed January 20, 2022).
- [4] Kithusya PM. Factors influencing organized crime in urban centres; the case of the City of Nairobi, in Kenya 2012.
- [5] Thurania N, and FM-IJ of SS, 2013 undefined. Collaboration between public and private security in Kenya. *Citeseer* 2013;1:1.
- [6] Cameron S. Killing for Money and the Economic Theory of Crime. <https://doi.org/10.1080/003467642013845336> 2014;72:28–41. <https://doi.org/10.1080/00346764.2013.845336>.
- [7] Fourie J. The Data Revolution in African Economic History. *J Interdiscip Hist* 2016;47:193–212. https://doi.org/10.1162/JINH_A_00977.
- [8] Mose NG. Determinants of regional economic growth in Kenya. *African J Bus Manag* 2021;15:1–12. <https://doi.org/10.5897/AJBM2020.9118>.
- [9] Stec A, Klabjan D. Forecasting Crime with Deep Learning 2018.
- [10] predictive analysis of crime using RNNs - Google Scholar n.d. https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=predictive+analysis+of+crime+using+RNNs&btnG= (accessed January 20, 2022).
- [11] Krishnan A, Sarguru A, and AS-IJ of P, 2018 undefined. Predictive analysis of crime data using deep learning. *AcadpublEu* n.d.
- [12] Patil AP, Nawal DJ, Jain D. Crime Prediction Application Using Artificial Intelligence. *Lect Notes Electr Eng* 2020;605:238–45. https://doi.org/10.1007/978-3-030-30577-2_20.