

DAT7015_2_2312602

by Oluwatosin Ojo

Submission date: 15-Jan-2024 11:52AM (UTC+0000)

Submission ID: 222042511

File name: 150319_Oluwatosin_Ojo_DAT7015_2_2312602_2208030_2035717712.docx (10.04M)

Word count: 5490

Character count: 29831



M.Sc. Data Analytics & Technologies

2023-2024

Big Data Technology in Crime Analysis: A Comprehensive Review of Criminal Datasets, Addressing Challenges and Proposing Solutions.

DAT7015 BIG DATA TECHNOLOGIES

Assignment 2

Student Id: 2312602

Modul Tutor: Anchal Argar | Pradeep



Abstract

The introduction of machine learning has resulted in a meteoric launch in the developments occurring in various sectors. There is a sudden technological shift in the use of artificial intelligence in the development of applications and solutions being used. There is an increase in the volume of data being generated in different formats (unstructured or unstructured), and the speed of generating and processing the data which is known as Big Data. Big data is primarily composed of 3 characteristics which are volume, variety, and velocity.

Big data just like the name says refers to a large set of data that cannot be processed using conventional methods. As a result, non-conventional methods had to be introduced to handle big data. This report highlights the two datasets given named criminal dataset and offense code dataset. Both datasets are regarded as big data.

The report studies the influence of big data in crime analysis, identifies issues before the introduction of big data, and the solutions proposed to solve the issues while referring to the datasets provided.

Keywords:

- Dataset
- Big Data,
- Machine Learning
- Crime Analysis
- Technology.

Table of Contents

<i>Abstract</i>	<i>i</i>
<i>Table of Contents</i>	<i>ii</i>
<i>List of Figures</i>	<i>iv</i>
<i>List of Tables</i>	<i>vi</i>
<i>Executive Summary</i>	<i>1</i>
<i>Research Questions</i>	<i>2</i>
<i>Research Methodology</i>	<i>3</i>
PRISMA Diagram.....	<i>4</i>
<i>Section A: Data Analysis</i>	<i>5</i>
A1. My Setup	<i>5</i>
1.1 Installing Spark and its session.....	<i>5</i>
1.2 Java and Hadoop.....	<i>5</i>
1.3 Install Map Reduce	<i>5</i>
1.4 Importing necessary libraries.....	<i>6</i>
1.5 Set Environment Variables.....	<i>6</i>
A2. Using Pandas.....	<i>7</i>
2.1 Read the ODS file.....	<i>7</i>
2.2 Load Dataset 1:	<i>7</i>
2.3 Create a copy of dataset.....	<i>8</i>
2.4 Remove worksheets from the workbook.....	<i>8</i>
2.5 Merging the worksheets	<i>9</i>
2.6 Pre-Processing Activity.....	<i>10</i>
A3. Using Spark SQL	<i>14</i>
3.1 Create a Spark data frame.....	<i>14</i>
3.2 Create a Temporary Table	<i>14</i>
3.3 Spark Queries.....	<i>15</i>
A4. Using Hive	<i>19</i>
4.1 Convert Spark dataframe to Hive.	<i>19</i>
4.2 Perform Hive Queries.....	<i>19</i>
A5. Using Spark	<i>21</i>
5.1 Create a Spark Dataframe	<i>21</i>
5.2 Splitting on columns.	<i>21</i>
5.3 Rearrange Columns.....	<i>22</i>
5.4 Drop Columns	<i>22</i>
5.5 Correlation	<i>23</i>

5.6 Outliers.....	25
A6. Map Reduce27
A7. Machine Learning.....	.28
7.1 Extract the dataset.....	.28
7.2 Encoding the columns.....	.28
7.3 Split the dataset.....	.29
7.4 Features and Target.....	.29
7.5 Initiate Models.....	.30
7.6 Mean Square Error.....	.31
7.4 Plot Graph31
A8. Visualization33
8.1 Visualization with Power BI.....	.33
8.2 Visualization with Tableau34
Section B: Crime Analysis Report36
B1. Analysis Answering Research Questions36
B2. Discussions and Research Findings39
B3. Issues In the Use of Big Data Analytics in Crime Analysis.	.43
B4. Solution Using Various Big Data Technologies.45
Conclusion47
References.....	.48
Abbreviations	<i>i</i>

List of Figures

Figure 1: PRISMA Diagram.....	4
Figure 2: Installing Dependencies.....	5
Figure 3: Install Java and Hadoop	5
Figure 4: Install Map Reduce Libraries	5
Figure 5: Import Pandas, Findspark	6
Figure 6: Set Environment Variable and Enable Spark session.....	6
Figure 7: Load the dataset.....	7
Figure 8: Create Dataset Copy.....	8
Figure 9: Removing worksheets.	8
Figure 10: Merge Worksheets.....	9
Figure 11: Rename Column	10
Figure 12: Drop Rows	11
Figure 13: Show Duplicates	11
Figure 14: Handling duplicates in Dataset 1.....	12
Figure 15: Handling Duplicates in Dataset2.....	12
Figure 16: Detect Missing Values	13
Figure 17: Handle Missing Values.....	13
Figure 18: Create a Spark Dataframe	14
Figure 19: Create a temporary Table.....	14
Figure 20: SQL Left Join	15
Figure 21: SQL Select.....	15
Figure 22: SQL Count.....	16
Figure 23: SQL Describe.....	16
Figure 24: SQL Columns.....	16
Figure 25: SQL Distinct	17
Figure 26: SQL Null Values.....	17
Figure 27: SQL Where.....	18
Figure 28: SQL Order By	18
Figure 29: Create a Hive Table.....	19
Figure 30: Hive Query 1	19
Figure 31: Hive Query 2	20

Figure 32: Create a New Spark Dataframe	21
Figure 33: Split Column	21
Figure 34: Reorder Column	22
Figure 35: Drop Column	22
Figure 36: Correlations in the dataset	23
Figure 37: Numerical Correlation	23
Figure 38: Bivariate Correlation.....	24
Figure 39: Installing Libraries for Outliers.....	25
Figure 40: Detect Outliers	25
Figure 41: Handle Outliers.....	26
Figure 42: Reduce Functools	27
Figure 43: Importing Libraries	28
Figure 44: Extract Dataset	28
Figure 45: Encode Categorical Variables	29
Figure 46: Train_Test_Split.....	29
Figure 47: Features and Target	29
Figure 48: Highlight Features and Target	29
Figure 49: Initiate Models	30
Figure 50: Linear Regression	30
Figure 51: Decision Tree	30
Figure 52: Random Forest	31
Figure 53: Mean Square Error (MSE).	31
Figure 54: Least Error Graph	32
Figure 55: Crime analysis with Power Bi	33
Figure 56: Offense Analysis.....	34
Figure 57: Force Analysis	35
Figure 58: Research Question 1 Solution	39
Figure 59: Research Question 2 Solution	40
Figure 60: Regression Model Graph	41

List of Tables

Table 1: Analysis answering research questions.....	36
---	----

Executive Summary

This report critically examines provided datasets which include criminal analysis across various financial years. The primary objective of the documentation is to explain the relationship within the datasets and also point out the relationship between big data and crime analysis. It aims to draft research questions after carefully reviewing the given dataset, perform some calculations on the dataset, and provide solutions to research questions.

In addition to addressing the research questions, the report focuses on the challenges associated with the use of big data solutions and the solutions provided in crime analysis. Upon highlighting the challenges, it offers strategic solutions adopted to mitigate against the known challenges. The approach used in the report ensures a thorough examination of the subject field, infusing both theoretical insights with practical knowledge.

Research Questions

The following research questions below have been carefully drafted after an extensive review of the datasets provided. The aim is to address the positive and negative effects of big data analytics in crime analysis.

1. What are the most common types of offenses recorded in the database? What are the top five offenses in the dataset and what relevance does that cover in the criminal analysis?
2. What force made the most arrests across all the financial years and in what financial year did they make the most arrest?
3. Are there any inconsistencies or anomalies in the data that need to be addressed to improve the data integrity? What technologies were adopted to ensure data reliability and robustness to provide meaningful contributions?
4. What analytical approach proves most effective in predictive modelling for anticipating the frequency and nature of offenses, and how can this method be optimally applied to enhance proactive law enforcement strategies?

Research Methodology

The methodology being adopted is the systematic review approach and the qualitative research method. The systematic review approach entails extracting already published journals, books, and articles. After extracting the necessary documents, I then select the documents that support my focus topic “Crime Analysis” and review the selections to answer the research questions provided (Laura, 2021).

Upon completion of the review process, we adopt the Triangulation approach, specifically the qualitative research process which includes working with multiple datasets and theories or technologies to address the research questions (Bhandari, 2022).

Some of the technologies adopted to answer the research questions include:

- the use of python,
- Pandas,
- Spark,
- Spark SQL,
- Hive
- Correlations,
- Map Reduce
- Machine Learning for predictive analytics,
- Tableau for Visualizations.

The technologies adopted and findings are further explained in SECTION A: Data Analysis.

PRISMA Diagram

The PRISMA flow diagram is a representation of the studies carried out, the phases involved, the selection process, and the findings.

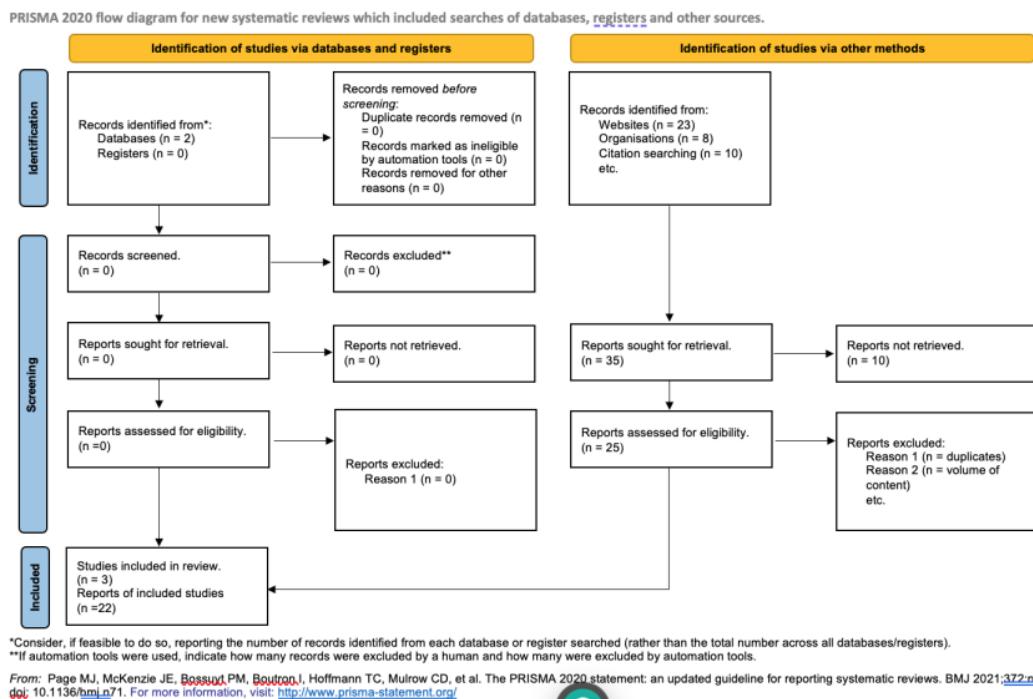


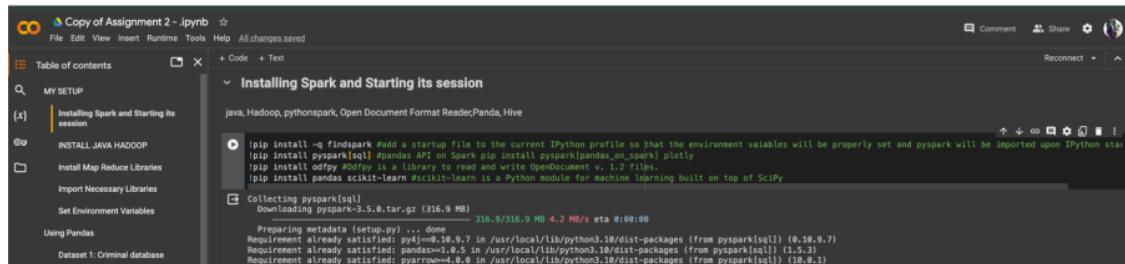
Figure 1: PRISMA Diagram

Source: (Page *et al.*, 2021)

Section A: Data Analysis

A1. My Setup

1.1 Installing Spark and its session.



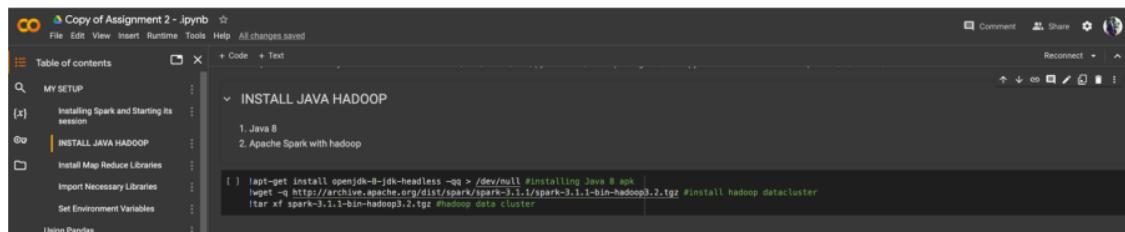
The screenshot shows a Jupyter Notebook interface titled "Copy of Assignment 2 - ipynb". The left sidebar has a "Table of contents" section with a tree view. The "MY SETUP" section is expanded, showing "Installing Spark and Starting its session", "INSTALL JAVA HADOOP", "Install Map Reduce Libraries", "Import Necessary Libraries", "Set Environment Variables", "Using Pandas", and "Dataset 1: Criminal database". The main notebook area shows a code cell under "Installing Spark and Starting its session" with the following command:

```
!pip install -q findspark #Add a startup file to the current IPython profile so that the environment variables will be properly set and pypark will be imported upon IPython start
!apt-get install openjdk-8-jdk-headless -qq > /dev/null #installing Java 8 apk
wget -q http://archive.apache.org/dist/spark/spark-3.1.1/spark-3.1.1-bin-hadoop3.2.tgz #install hadoop datacluster
tar xf spark-3.1.1-bin-hadoop3.2.tgz #hadoop data cluster
```

Output from the command shows the download and extraction of PySpark and Hadoop packages.

Figure 2: Installing Dependencies

1.2 Java and Hadoop



The screenshot shows a Jupyter Notebook interface titled "Copy of Assignment 2 - ipynb". The left sidebar has a "Table of contents" section with a tree view. The "MY SETUP" section is expanded, showing "Installing Spark and Starting its session", "INSTALL JAVA HADOOP", "Install Map Reduce Libraries", "Import Necessary Libraries", "Set Environment Variables", and "Using Pandas". The main notebook area shows a code cell under "INSTALL JAVA HADOOP" with the following command:

```
!apt-get install openjdk-8-jdk-headless -qq > /dev/null #installing Java 8 apk
wget -q http://archive.apache.org/dist/spark/spark-3.1.1/spark-3.1.1-bin-hadoop3.2.tgz #install hadoop datacluster
tar xf spark-3.1.1-bin-hadoop3.2.tgz #hadoop data cluster
```

Output from the command shows the download and extraction of Java and Hadoop packages.

Figure 3: Install Java and Hadoop

1.3 Install Map Reduce



The screenshot shows a Jupyter Notebook interface titled "Copy of Assignment 2 - ipynb". The left sidebar has a "Table of contents" section with a tree view. The "MY SETUP" section is expanded, showing "Set Environment Variables", "Using Pandas", "Dataset 1: Criminal database", "Load the dataset", "Highlight the sheets in the dataset", and "Make a copy". The main notebook area shows a code cell under "Install Map Reduce Libraries" with the following command:

```
!pip install mrjob #install the library for Map Reduce
!pip install pathlib
!pip install PyYAML
```

Output from the command shows the download and extraction of MrJob library.

Figure 4: Install Map Reduce Libraries

1.4 Importing necessary libraries.

The screenshot shows a Jupyter Notebook interface with a sidebar containing a table of contents for a 'MY SETUP' section. The main area displays Python code for importing Pandas and setting up a Spark session.

```
import pandas as pd
import numpy as np

from pyspark.sql import SparkSession
spark = SparkSession.builder.master("local[*]").appName("Assignment2").getOrCreate()
spark.conf.set("spark.sql.repair.schema.enabled", True) # Property used to format output tables better
```

Below the code, it shows the configuration of the SparkSession:

```
sparkSession - In-memory
SparkContext
SparkUI
Version: v3.5.8
Master: [local]
AppName: Assignment2
```

Figure 5: Import Pandas, Findspark

1.5 Set Environment Variables

The screenshot shows a Jupyter Notebook interface with a sidebar containing a table of contents for a 'MY SETUP' section. The main area displays Python code for setting environment variables and creating a Spark session.

```
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.1.1-bin-hadoop3.2"

!ls
sample_data spark-3.1.1-bin-hadoop3.2 spark-3.1.1-bin-hadoop3.2.tgz

# Enabling Hive to use in Spark
spark = SparkSession.builder \
    .master("local[*]") \
    .appName("Assignment2") \
    .config("spark.sql.warehouse.dir", "<path>/sparkhivewarehouse") \
    .enableNativeSupport() \
    .getOrCreate()
```

Figure 6: Set Environment Variable and Enable Spark session.

A2. Using Pandas

2.1 Read the ODS file.

Given that the datasets given are both in ODS formats, I must first read the ODS file by using Pandas. Afterward, I will convert the panda file into a PySpark and create a database for the dataset.

2.2 Load Dataset 1:

Dataset 1 is the criminal workbook containing multiple worksheets. In this section, I endeavor to highlight the worksheets I have decided to work with, remove the worksheets I don't need, and merge all the worksheets highlighted into a single worksheet for easy access.

The screenshot shows a Jupyter Notebook interface with a sidebar containing a 'Table of contents' for 'Dataset 1: Criminal database for different years'. The main area displays Python code for reading an ODS file and processing its data. The code includes:

- Reading the ODS file: `dataset = pd.read_excel("/content/drive/MyDrive/Colab Notebooks/Assignment 2/Dataset.ods", sheet_name = None)`
- Highlighting the worksheets in the dataset: `dataset.keys()`
- Checking the first 5 rows of the '2012-13' worksheet: `dataset['2012-13'].head(5)`
- Checking the details of columns: `dataset['2014-15'].shape`, `dataset['2015-16'].shape`, `dataset['2021-22'].shape`, `dataset['2022-23'].shape`

The output of the code shows the first 5 rows of the '2012-13' worksheet:

Financial Year	Financial Quarter	Force Name	Offence Description	Offence Group	Offence Subgroup	Offence Code	Number of Offences
0	2012/13	1	Action Fraud	Fraud offences	Fraud Action Fraud	AF	20323
1	2012/13	1	Avon and Somerset	Abandoning child under two years (incomes only)	Violence against the person	Violence without injury	12
2	2012/13	1	Avon and Somerset	Abandoning from lawful custody	Miscellaneous crimes against society	Miscellaneous offences against society	80
3	2012/13	1	Avon and Somerset	Abuse of children through prostitution and por...	Sexual offences	Other sexual offences	71
4	2012/13	1	Avon and Somerset	Abuse of position of trust of a sexual nature	Sexual offences	Other sexual offences	73

Figure 7: Load the dataset.

- To Read the dataset:

29

```
Dataset = pd.read_excel("/content/drive/MyDrive/Colab Notebooks/Assignment 2/Dataset.ods",
sheet_name = None )
```

- To know how many worksheets are in the workbook: `dataset.keys()`
- To check the first 5 rows of a particular worksheet: `dataset ['2012-13'].head(5) #code to check first 5 rows in 2012-13 worksheet`

- To know the details of a particular worksheet: dataset ['2014-15'].shape #code to check details of the column

2.3 Create a copy of dataset.

Make a copy of the dataset before performing operations so you have an original file to fall back to.

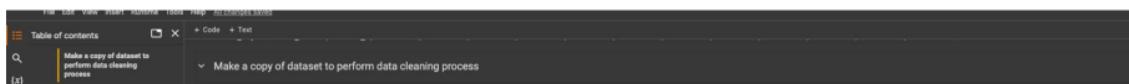


Figure 8: Create Dataset Copy

2.4 Remove worksheets from the workbook.

 A screenshot of a Jupyter Notebook interface. The top menu bar includes File, Edit, View, Insert, Notebook, Tools, Help, and Help. Below the menu is a 'Table of contents' sidebar with several items: 'Cleaning up the dataset' (selected), 'Make a copy of dataset to perform data cleaning process', 'Merge the sheets in the dataset', 'Pre-processing Techniques', 'Rename the Column', 'Checking for duplicates', 'Handling Null Values', 'Creating a SparkDataFrame', 'Creating a Temporary Table', 'Dataset 2 - Offence Codes', 'Pre-processing Techniques', and 'Load the dataset'. The main area shows a code cell starting with '+ Code + Text' followed by Python code for removing worksheets. The code includes:


```
[ ] del dataset1['Cover_sheet'] #deleting the worksheet "Coversheet"
[ ] dataset1.keys() #Code to check no of worksheets in dataset
dict_keys(['Notes_','2012-13','2013-14','2014-15','2015-16','2016-17','2017-18','2018-19','2019-20','2020-21','2021-22','2022-23','2023-24'])
SheetD represents worksheets we are not using
[ ] sheetD = ['Notes_','2012-13','2013-14','2016-17','2017-18','2018-19','2019-20','2020-21','2023-24'] #deleting worksheet 'Notes'
[ ] for sheet_name in sheetD:
    if sheet_name in dataset1:
        del dataset1[sheet_name]
dataset1.keys()
dict_keys(['2014-15','2015-16','2021-22','2022-23'])
```

Figure 9: Removing worksheets.

Upon making a copy, I can now highlight worksheets I won't be using and remove them from the workbook.

In the first line of code, I attempt to delete the worksheet cover sheet. However, due to the volume of worksheets identified, I attempted to write a for-loop construct to delete the worksheets highlighted once.

Upon deletion, I therefore use .keys() to show the worksheets left in the workbook.

2.5 Merging the worksheets.

In this section, after removing unwanted worksheets, I proceed to merge the worksheets left into one worksheet for easy access so I can easily clean up and perform our pre-processing activities conveniently.

The screenshot shows a Jupyter Notebook interface with a sidebar containing a table of contents and a main code editor area. The table of contents lists various sections such as 'Merging the sheets in the dataset', 'Pre-processing Techniques', 'Handling Null Values', etc. The main code editor contains Python code for concatenating datasets:

```
[ ] Dataset1 = pd.concat(dataset1, ignore_index=True) #new dataset without Notes and Coversheet  
[ ] Dataset1.keys()  
Index(['Financial Year', 'Financial Quarter', 'Force Name',  
       'Offence Description', 'Offence Group', 'Offence Subgroup',  
       'Offence Code', 'Number of Offences'],  
      dtype='object')  
  
Dataset1.info()  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 89984 entries, 0 to 89983  
Data columns (total 8 columns):  
 #   Column           Non-Null Count  Dtype     
 0   Financial Year    89984 non-null   object    
 1   Financial Quarter 89984 non-null   int64    
 2   Force Name         89984 non-null   object    
 3   Offence Description 89984 non-null   object    
 4   Offence Group      89984 non-null   object    
 5   Offence Subgroup   89984 non-null   object    
 6   Offence Code        89984 non-null   object    
 7   Number of Offences 89984 non-null   int64    
dtypes: int64(2), object(6)  
memory usage: 5.5+ MB  
  
[ ] Dataset1.shape  
(89984, 8)
```

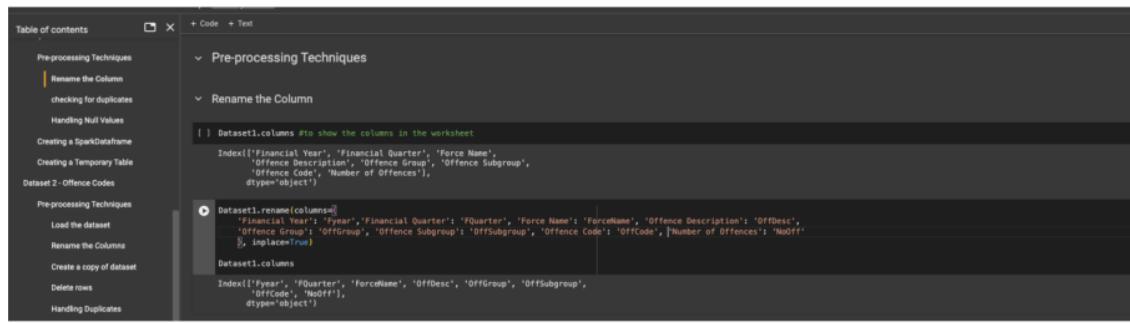
Figure 10: Merge Worksheets

2.6 Pre-Processing Activity

The pre-processing activities done in this section are to ensure the dataset is being worked on and prepared for final use. This is to remove redundancies and correct the errors identified.

2.6.1 Rename Columns

The first task is to rename the columns. This is done because having column names with spaces is difficult to read and work on.



A screenshot of a Jupyter Notebook interface. The left sidebar contains a 'Table of contents' with several sections: 'Pre-processing Techniques' (selected), 'Rename the Column', 'checking for duplicates', 'Handling Null Values', 'Creating a SparkDataFrame', 'Creating a Temporary Table', 'Dataset 2: Offence Codes', 'Pre-processing Techniques' (selected again), 'Load the dataset', 'Rename the Column', 'Create a copy of dataset', 'Delete rows', and 'Handling Duplicates'. The main area shows two snippets of Python code. The top snippet is a comment: '# show the columns in the worksheet'. The bottom snippet is the actual code: 'Dataset1.rename(columns={\'Financial Year\': \'Fyear\', \'Financial Quarter\': \'Quarter\', \'Force Name\': \'ForceName\', \'Offence Description\': \'OffDesc\', \'Offence Group\': \'OffGroup\', \'Offence Subgroup\': \'OffSubgroup\', \'Offence Code\': \'OffCode\', \'Number of Offences\': \'NoOff\'}, inplace=True)'. Below this, another comment 'Dataset1.columns' is followed by 'Index([\'Fyear\', \'Quarter\', \'ForceName\', \'OffDesc\', \'OffGroup\', \'OffSubgroup\', \'OffCode\', \'NoOff\'], inplace=True)'.

Figure 11: Rename Column

2.6.2 Drop Rows

I identified some unwanted rows at the end of our dataset and use pandas to remove them.

- `dataset2.tail(5)`
- `dataset2.drop(dataset2.index[-2:], inplace=True) #delete last two rows of the dataset`

Figure 12: Drop Rows

2.6.3 Handling duplicates.

In this dataset, I attempt to show a list of the duplicate rows.

Figure 13: Show Duplicates

Some files could have been entered multiple times, therefore I use the .drop_duplicates() to remove duplicate entries.

The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with options like 'Using Pandas', 'Dataset 1: Criminal database', 'Load the dataset', 'Highlight the sheets in the dataset', and 'Make a copy'. The main area has a title 'Handling duplicates' and a code cell containing:
```python  
Dataset1 = [dataset1.drop\_duplicates(Dataset1) #deleting the duplicates found in the worksheet  
Dataset1.shape  
(8994, 8)  
```

Figure 14: Handling duplicates in Dataset 1

There are no duplicates in dataset 1 as it still returns the initial dataset information.

The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with options like 'Make a copy', 'Remove worksheets from the workbook', 'Merging the worksheets', 'Rename the Column', and 'Handling duplicates'. The main area has a code cell containing:
```python  
[ ] distinct\_values = dataset2['Offencecode'].unique() #code to check for distinct rows in offcode  
[ ] Dataset2 = dataset2.drop\_duplicates(Dataset2) # Drop duplicates deletes the duplicates after the first occurrence.  
Dataset2.shape  
(186, 6)  
```

Figure 15: Handling Duplicates in Dataset2

2.6.4 Missing Values

11 Missing data is defined as the values or data that is not stored (or not present) for some variable/s in the given data. Missing values in datasets are common in datasets and can sometimes be needed to ensure the accuracy of data. In any dataset, you must first

- Detect missing values.
- Handle missing values.

2.6.5 Detect Missing Values

The next step is to work with missing Null Values. To do this, I must first identify the null values in the dataset, upon identifying this, if any, I can then perform some activities on it.

As shown in the figure below,

- Isnull() will check the dataset and show true or false for where it finds null values.
- Isnull().sum() will check for sum of null values in each column.

The screenshot shows a Jupyter Notebook interface. On the left is a sidebar with various options like 'Rename or copy', 'Merge worksheets', 'Handling duplicates', 'Detect Missing Values', etc. The main area has a section titled 'Detecting Missing values' with the following code:

```
[ ] Dataset2.isnull() #code to check for null values
```

Below this is a table showing the results of the isnull() operation:

	Offencecode	Offdescription	oldOffGroup	oldOffsubGroup	newOffGroup	newSGroup
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
...
191	False	False	False	True	False	False
192	False	False	True	True	False	False
193	False	False	False	True	False	False
194	False	False	False	False	False	False
195	False	False	False	False	False	False

186 rows x 6 columns

```
[ ] Dataset2.isnull().sum() #code to give a sum of null values in each column
```

Below this is another table showing the sum of null values for each column:

	Offencecode	Offdescription	oldOffGroup	oldOffsubGroup	newOffGroup	newSGroup
Offencecode	0	0	0	0	0	0
Offdescription	0	0	0	0	0	0
oldOffGroup	0	0	0	0	0	0
oldOffsubGroup	0	0	0	0	0	0
newOffGroup	0	0	0	0	0	0
newSGroup	0	0	0	0	0	0

dtype: int64

Figure 16: Detect Missing Values

2.6.6 Handle Missing Values

There are different methods to handle missing values some of which include using:

- Replace with mean, median, mode,
- Replace with constant values (0),
- Imputation (using nearby values),
- Delete rows/columns,
- Impute the data by yourself.

The screenshot shows a Jupyter Notebook interface. On the left is a sidebar with various options like 'Rename or copy', 'Merge worksheets', 'Handling duplicates', 'Handle Missing Values', etc. The main area has a section titled 'Handling Missing Values' with the following code:

```
[ ] #off.fillna(0, inplace=True) #code to change every null value to a constant (0).
```

```
[ ] Dataset2['oldOffGroup'].fillna(0, inplace = True) #code to change null values in oldOffgroup with a constant (0).
```

Below this is a warning message:

```
<ipython-input-77-bc49cbff15b>:1: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame  
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy  
Dataset2['oldOffGroup'].fillna(0, inplace = True) #code to change null values in oldOffgroup with a constant (0).
```

```
[ ] Dataset2['oldOffsubGroup'].fillna('Unknown', inplace=True) #code to change null values in oldsubgroup with 'Unknown'.
```

Below this is another warning message:

```
<ipython-input-79-6137cc5a92d1>:1: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame  
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy  
Dataset2['oldOffsubGroup'].fillna('Unknown', inplace=True) #code to change null values in oldsubgroup with 'Unknown'.
```

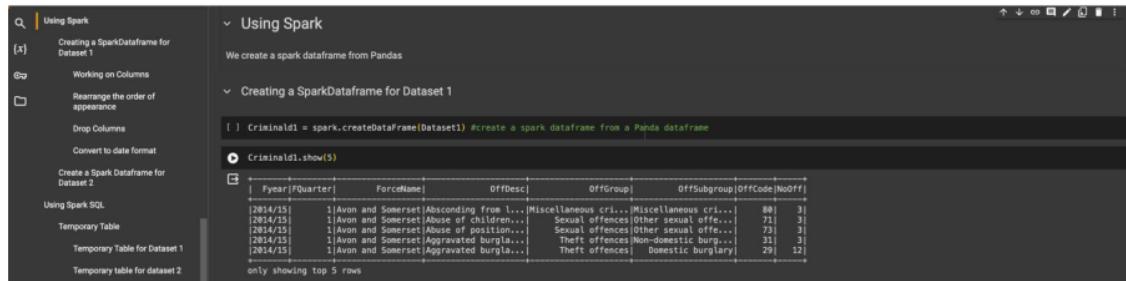
```
[ ] Dataset2.isnull().sum()
```

Figure 17: Handle Missing Values

A3. Using Spark SQL

3.1 Create a Spark data frame.

Upon completing the initial cleaning with pandas, I convert the padas dataset into a spark data frame.



The screenshot shows a Jupyter Notebook interface with a sidebar containing navigation links like 'Using Spark', 'Creating a SparkDataFrame for Dataset 1', 'Working on Columns', 'Rearrange the order of appearance', 'Drop Columns', 'Convert to date format', 'Create a Spark DataFrame for Dataset 2', 'Using Spark SQL', 'Temporary Table', 'Temporary Table for Dataset 1', and 'Temporary table for dataset 2'. The main area displays code and its output. The code is:

```
Criminaldf = spark.createDataFrame(Dataset1) #create a spark dataframe from a Pandas DataFrame
Criminaldf.show(5)
```

The output shows the first five rows of the DataFrame:

Fyear	FQuarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	
2014/15	1	Avon and Somerset	Abandoning from 1...	Miscellaneous cri...	Sexual offences	Other sexual offe...	88	3
2014/15	1	Avon and Somerset	Abuse of children...	Sexual offences	Other sexual offe...	71	3	
2014/15	1	Avon and Somerset	Abuse of position...	Sexual offences	Other sexual offe...	73	3	
2014/15	1	Avon and Somerset	Aggravated burlgari...	Theft offences	Non-domestic burg...	31	3	
2014/15	1	Avon and Somerset	Aggravated burlgari...	Theft offences	Domestic burglary	29	12	

only showing top 5 rows

Figure 18: Create a Spark Dataframe

3.2 Create a Temporary Table

For us to work with SQL, I have to save the spark dataframe on a SQL database by creating a temporary table to accept the values in datasets 1 and 2.



The screenshot shows a Jupyter Notebook interface with a sidebar containing navigation links like 'Using Spark SQL', 'Temporary Table', 'Temporary Table for Dataset 1', 'Perform SQL Queries using Spark SQL', 'SQL Syntax 1: to Merge Datasets', 'Handle Columns', and 'SQL Syntax 2: Sqld Select, SQl Count'. The main area displays code and its output. The code is:

```
Criminaldf.createOrReplaceTempView("Crime") # Register Temporary Table
Offencecode.createOrReplaceTempView("Offence") #creating a temporary table name offence
```

Figure 19: Create a temporary Table.

To perform SQL operations, I create a temporary table using `createOrReplaceTempView()`. This is done to accept the details of the dataset in the temporary table format in SQL.

3.3 Spark Queries

SQL Queries used in this code are:

3.3.1 SQL Join

SQL Join is used to merge different tables using similar values between both tables. It combines rows on different tables using the related columns. (W3Schools, no date)

The SQL join query can be done in different ways, some of which are:

- 26
 - Inner Join
 - Left Join
 - Right Join
 - Full join
 - Union



```
Using Spark SQL
Temporary Table
Temporary Table for Dataset 1
Temporary table for dataset 2
Perform SQL Queries using
Spark SQL
SQL Syntax 1: to Merge
Datasets
```

SQL Syntax 1: to Merge Datasets

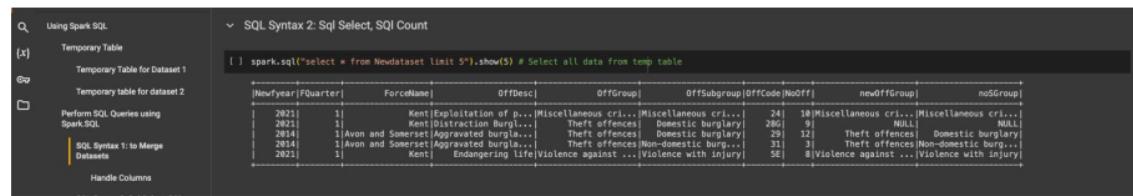
```
[ ] #oc.createOrReplaceTempView("Mergedata") #creating a temporary table name offence
[ ] Mergedata = spark.sql("SELECT * FROM Crime LEFT JOIN Offence ON Crime.OffCode = offence.Offencecode") #code to merge the datasets into one
[ ] Mergedata.count()
[ ] 89984
```

Figure 20: SQL Left Join

3.3.2 SQL Select

SQL Select is used to specify the columns to view in the table.

- SQL Select Column_name is used to select a specific column in the table named column_name.



```
Using Spark SQL
Temporary Table
Temporary Table for Dataset 1
Temporary table for dataset 2
Perform SQL Queries using
Spark SQL
SQL Syntax 1: to Merge
Datasets
```

Handle Columns

SQL Syntax 2: Sq1 Select, Sq1 Count

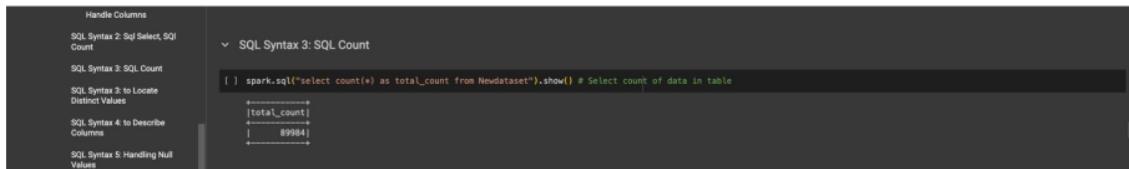
```
[ ] spark.sql("select * from Newdataset limit 5").show(5) # Select all data from temp table
```

NewYear	FQuarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	newOffGroup	noSGroup
2021	1	Kent	Exploitation of p... Miscellaneous cri...	Miscellaneous cri...	241	18	Miscellaneous cri...		
2021	1	Kent	Distraction Burgl...	Theft offences	Domestic burglary	286	9	NULL	NULL
2021	1	Avon and Somerset	Aggravated burgla...	Theft offences	Domestic burglary	29	12	Theft offences	Domestic burglary
2021	1	Avon and Somerset	Aggravated burgla...	Theft offences	Non-domestic burg...	31	1	Theft offences	Non-domestic burg...
2021	1	Kent	Endangering life/Violence against ... Violence with injury			56	8	Violence against ... Violence with injury	

Figure 21: SQL Select

3.3.3 SQL Count

SQL Count is used to count the number of rows in a table



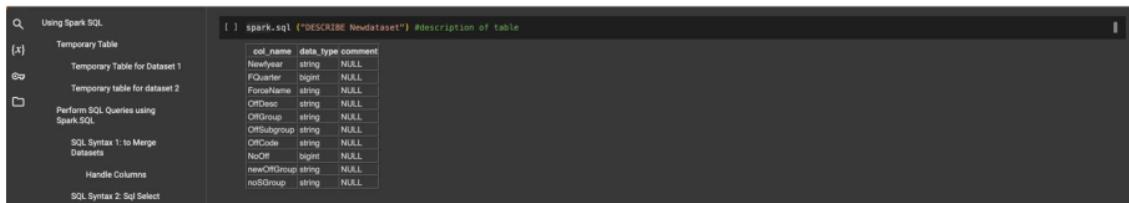
```
Handle Columns
SQL Syntax 2: Sql Select, SQL Count
SQL Syntax 3: SQL Count
SQL Syntax 3: to Locate Distinct Values
SQL Syntax 4: to Describe Columns
SQL Syntax 5: Handling Null Values

SQL Syntax 3: SQL Count
[ ] spark.sql("select count(*) as total_count from Newdataset").show() # Select count of data in table
+-----+
|total_count|
+-----+
| 89984 |
+-----+
```

Figure 22: SQL Count

3.3.4 SQL Describe

SQL Describe gives a breakdown of the structure of a database.



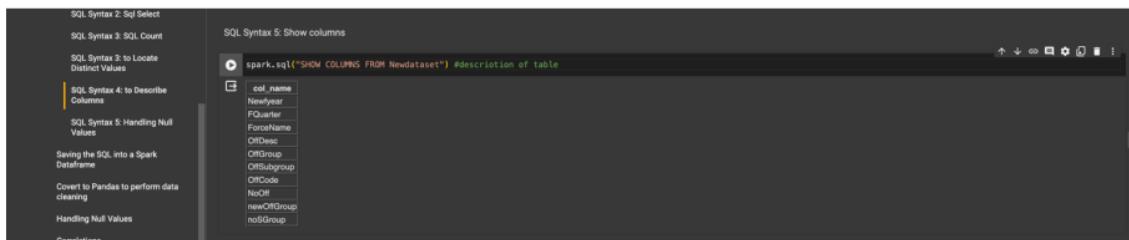
```
Using Spark SQL
Temporary Table
Temporary Table for Dataset 1
Temporary table for dataset 2
Perform SQL Queries using Spark SQL
SQL Syntax 1: to Merge Datasets
Handle Columns
SQL Syntax 2: Sql Select

spark.sql ("DESCRIBE Newdataset") #description of table
+-----+
|col_name|data_type|comment|
+-----+
|Newyear |string |NULL |
|FQuarter|bigint|NULL |
|ForName |string |NULL |
|OffDesc |string |NULL |
|OffGroup|string |NULL |
|OffSubgroup|string |NULL |
|OffCode |string |NULL |
|NoOff |bigint|NULL |
|newOffGroup|string |NULL |
|noSGroup |string |NULL |
+-----+
```

Figure 23: SQL Describe

3.3.5 SQL Show Columns

Show Columns state is used to show the details of the columns in a database.



```
SQL Syntax 2: Sql Select
SQL Syntax 3: SQL Count
SQL Syntax 3: to Locate Distinct Values
SQL Syntax 4: to Describe Columns
SQL Syntax 5: Handling Null Values
Saving the SQL into a Spark Dataframe
Convert to Pandas to perform data cleaning
Handling Null Values
Correlations

SQL Syntax 5: Show columns
spark.sql("SHOW COLUMNS FROM Newdataset") #description of table
+-----+
|col_name|
+-----+
|Newyear |
|FQuarter|
|ForName |
|OffDesc |
|OffGroup|
|OffSubgroup|
|OffCode |
|NoOff |
|newOffGroup|
|noSGroup |
+-----+
```

Figure 24: SQL Columns

3.3.6 SQL Distinct

SELECT DISTINCT statement is used to show distinct values in a column.

```

spark.sql("SELECT DISTINCT OffGroup FROM Newdataset").show() highlighting the different OffGroup in the table
spark.sql("select COUNT(DISTINCT OffGroup) from Newdataset").show()
+-----+
|OffGroup|
+-----+
|Public order offe...|
|Fraud offences|
|Sexual offences|
|Criminal damage a...|
|Possession of arms...|
|Drug offences|
|Theft offences|
|Miscellaneous cri...|
|Violence against ...|
+-----+
|count(DISTINCT OffGroup)|
+-----+
|          18|
+-----+

```

Figure 25: SQL Distinct

3.3.7 SQL Null Values

In SQL we first detect the null values using IS NULL.

After detecting Null Values, we can either handle them by:

- Deletion
- Imputation using Coalesce Function as shown below.

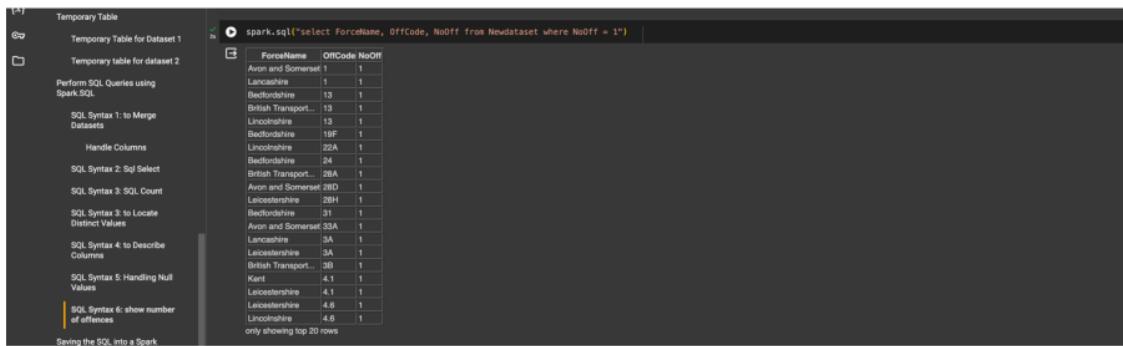
Newyear FQuarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	newOffGroup	noSGroup
2021	1	Avon and Somerset	Assaults on emerg...	Violence against...	Violence with injury	8	NULL	NULL
2021	1	Bedfordshire	Assaults on emerg...	Violence against...	Violence with injury	5	NULL	NULL
2021	1	British Transport	Assaults on emerg...	Violence against...	Violence with injury	0	NULL	NULL
2021	1	Cambridgeshire	Assaults on emerg...	Violence against...	Violence with injury	7	NULL	NULL
2021	1	Chester	Assaults on emerg...	Violence against...	Violence with injury	13	NULL	NULL
2021	1	Cleveland	Assaults on emerg...	Violence against...	Violence with injury	8	NULL	NULL
2021	1	Cumbria	Assaults on emerg...	Violence against...	Violence with injury	8	NULL	NULL
2021	1	Derbyshire	Assaults on emerg...	Violence against...	Violence with injury	1	NULL	NULL
2021	1	Devon and Cornwall	Assaults on emerg...	Violence against...	Violence with injury	14	NULL	NULL
2021	1	Dorset	Assaults on emerg...	Violence against...	Violence with injury	0	NULL	NULL
2021	1	Dyfed-Powys	Assaults on emerg...	Violence against...	Violence with injury	5	NULL	NULL
2021	1	Essex	Assaults on emerg...	Violence against...	Violence with injury	7	NULL	NULL
2021	1	Gloucestershire	Assaults on emerg...	Violence against...	Violence with injury	0	NULL	NULL
2021	1	Greater Manchester	Assaults on emerg...	Violence against...	Violence with injury	21	NULL	NULL
2021	1	Gwent	Assaults on emerg...	Violence against...	Violence with injury	17	NULL	NULL
2021	1	Hampshire	Assaults on emerg...	Violence against...	Violence with injury	23	NULL	NULL
2021	1	Hertfordshire	Assaults on emerg...	Violence against...	Violence with injury	15	NULL	NULL
2021	1	Humber	Assaults on emerg...	Violence against...	Violence with injury	8	NULL	NULL
2021	1	Kent	Assaults on emerg...	Violence against...	Violence with injury	36	NULL	NULL

Newyear FQuarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	newOffGroup	noSGroup	Nosp	Nosp
2021	1	Kent	Murder	Violence against...	Homicide	1	3	Violence against...	Violence against...	Homicide
2021	1	Avon and Somerset	Assault without i...	Violence without...	Violence without...	104	125	Violence against...	Violence against...	Violence without...
2014	1	Avon and Somerset	Assault without i...	Violence against...	Violence without...	108A	2008	Violence against...	Violence against...	Violence without...
2021	1	Kent	Modest slavery	Violence against...	Violence without...	106	88	NULL	unknown	unknown
2021	1	Kent	Interfering with...	Vehicle interference	Vehicle interference	126	308	Theft offences	Vehicle offences	Theft offences
2014	1	Avon and Somerset	Attempted murder	Violence against...	Violence with injury	2	4	Violence against...	Violence with injury	Violence against...

Figure 26: SQL Null Values

3.3.8 SQL Where

SQL where is used to specify a particular argument or position. It helps for filtering purposes.



The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with a tree view of notebooks and a search bar. The main area contains a code cell with the following content:

```
spark.sql("select ForceName, OffCode, NoOff from Newdataset where NoOff = 1")
```

Below the code cell is a table output showing the results of the query:

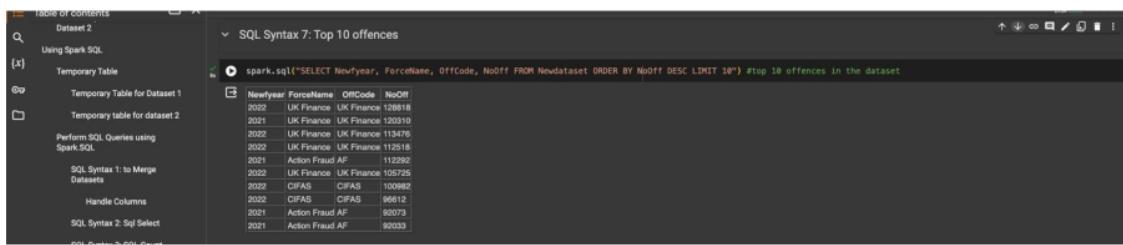
ForceName	OffCode	NoOff
Avon and Somerset	1	1
Bedfordshire	13	1
British Transport	13	1
Lincolnshire	13	1
Bedfordshire	19F	1
Lincolnshire	22A	1
Bedfordshire	24	1
British Transport	28A	1
Avon and Somerset	28D	1
Leicestershire	28H	1
Bedfordshire	31	1
Avon and Somerset	3A	1
Leicestershire	3A	1
Lincolnshire	3A	1
British Transport	3B	1
Kent	4.1	1
Leicestershire	4.1	1
Leicestershire	4.6	1
Lincolnshire	4.6	1

only showing top 20 rows

Figure 27: SQL Where

3.3.9 SQL Order By

SQL Order by is used when trying to show a result in either ascending or descending order.



The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with a tree view of notebooks and a search bar. The main area contains a code cell with the following content:

```
spark.sql("SELECT Newyear, ForceName, OffCode, NoOff FROM Newdataset ORDER BY NoOff DESC LIMIT 10") #top 10 offences in the dataset
```

Below the code cell is a table output showing the results of the query:

Newyear	ForceName	OffCode	NoOff
2022	UK Finance	UK Finance	128918
2021	UK Finance	UK Finance	120310
2022	UK Finance	UK Finance	113476
2022	UK Finance	UK Finance	112518
2021	Action Fraud AF		112929
2022	UK Finance	UK Finance	105725
2022	CIFAS	CIFAS	100860
2022	CIFAS	CIFAS	96612
2021	Action Fraud AF		92373
2021	Action Fraud AF		92033

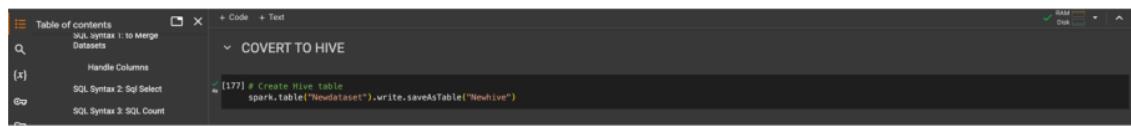
Figure 28: SQL Order By

A4. Using Hive

Upon completing the SQL queries and clean-up, I then attempt to write the SQL table into a Hive table to perform Hive Queries using Google Collab.

4.1 Convert Spark dataframe to Hive.

In this section, I convert the spark dataframe named Newdataset to Hive named Newhive.



The screenshot shows a Google Colab interface. On the left, there's a sidebar with various notebooks and sections like 'Table of contents', 'Handle Columns', and 'SQL Syntax'. The main area has a title 'COVERT TO HIVE' and contains the following code:

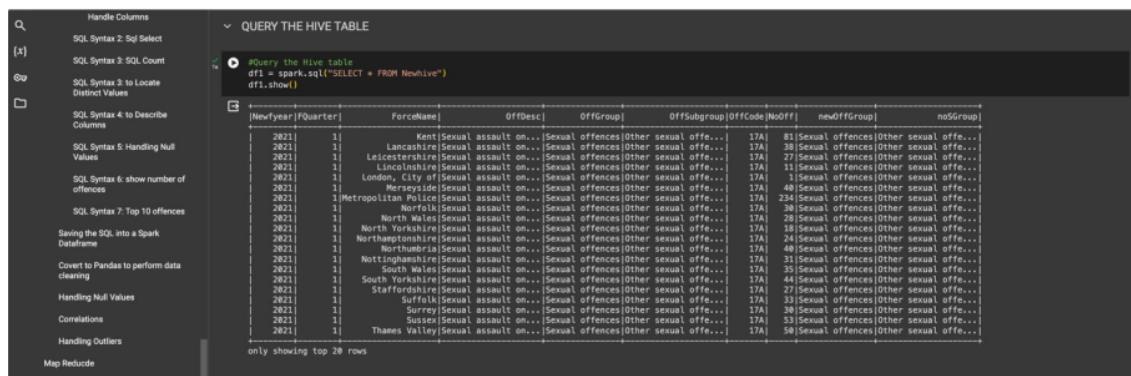
```
[177] # Create Hive table
spark.table("Newdataset").write.saveAsTable("Newhive")
```

Figure 29: Create a Hive Table

4.2 Perform Hive Queries

4.2.1 Hive Query 1

Hive Query to show the hive table



The screenshot shows a Google Colab interface with a sidebar containing various notebooks and sections. The main area displays a query titled 'QUERY THE HIVE TABLE' with the following code:

```
#Query the Hive table
df = spark.sql("SELECT * FROM Newhive")
df.show()
```

Below the code, the results are displayed in a table with the following columns: [NewYear], [Quarter], [ForceName], [OffDesc], [OffGroup], [OffSubgroup], [OffCode], [NoOff], [newOffGroup], and [no5Group]. The table shows data for 2021, including entries for Kent, Lancashire, Leicestershire, Lincolnshire, London City of, Merseyside, Metropolitan Police, North Wales, North Yorkshire, Northamptonshire, Nottinghamshire, South Wales, Staffordshire, Suffolk, Surrey, Sussex, and Thames Valley, detailing sexual assault offenses.

NewYear	Quarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	newOffGroup	no5Group
2021	1	Kent	Sexual assault on...	Sexual offences	Other sexual off...	17A	81	Sexual offences	Other sexual off...
2021	1	Lancashire	Sexual assault on...	Sexual offences	Other sexual off...	17A	38	Sexual offences	Other sexual off...
2021	1	Leicestershire	Sexual assault on...	Sexual offences	Other sexual off...	17A	27	Sexual offences	Other sexual off...
2021	1	Lincolnshire	Sexual assault on...	Sexual offences	Other sexual off...	17A	11	Sexual offences	Other sexual off...
2021	1	London City of	Sexual assault on...	Sexual offences	Other sexual off...	17A	1	Sexual offences	Other sexual off...
2021	1	Merseyside	Sexual assault on...	Sexual offences	Other sexual off...	17A	40	Sexual offences	Other sexual off...
2021	1	Metropolitan Police	Sexual assault on...	Sexual offences	Other sexual off...	17A	234	Sexual offences	Other sexual off...
2021	1	North Wales	Sexual assault on...	Sexual offences	Other sexual off...	17A	30	Sexual offences	Other sexual off...
2021	1	North Yorkshire	Sexual assault on...	Sexual offences	Other sexual off...	17A	28	Sexual offences	Other sexual off...
2021	1	Northamptonshire	Sexual assault on...	Sexual offences	Other sexual off...	17A	18	Sexual offences	Other sexual off...
2021	1	Nottinghamshire	Sexual assault on...	Sexual offences	Other sexual off...	17A	24	Sexual offences	Other sexual off...
2021	1	South Wales	Sexual assault on...	Sexual offences	Other sexual off...	17A	41	Sexual offences	Other sexual off...
2021	1	Nottinghamshire	Sexual assault on...	Sexual offences	Other sexual off...	17A	31	Sexual offences	Other sexual off...
2021	1	South Wales	Sexual assault on...	Sexual offences	Other sexual off...	17A	35	Sexual offences	Other sexual off...
2021	1	South Yorkshire	Sexual assault on...	Sexual offences	Other sexual off...	17A	30	Sexual offences	Other sexual off...
2021	1	Staffordshire	Sexual assault on...	Sexual offences	Other sexual off...	17A	27	Sexual offences	Other sexual off...
2021	1	Suffolk	Sexual assault on...	Sexual offences	Other sexual off...	17A	33	Sexual offences	Other sexual off...
2021	1	Surrey	Sexual assault on...	Sexual offences	Other sexual off...	17A	30	Sexual offences	Other sexual off...
2021	1	Sussex	Sexual assault on...	Sexual offences	Other sexual off...	17A	33	Sexual offences	Other sexual off...
2021	1	Thames Valley	Sexual assault on...	Sexual offences	Other sexual off...	17A	36	Sexual offences	Other sexual off...

Figure 30: Hive Query 1

4.2.2 Hive Query 2

Hive query to the data of offenses in the year 2021

The screenshot shows a Jupyter Notebook interface with a sidebar containing various SQL-related functions like Handle Columns, SQL Syntax 2: Sql Select, etc. The main area displays a query titled 'QUERY HIVE TABLE 2':

```
[182]: df2 = spark.sql("Select * from hivetable where Newyear = 2021")
df2.show()
```

The resulting DataFrame 'df2' has the following schema and data:

Newyear	Forcequarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	newOffGroup	no5Group
2021	1	Kent	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	31	Sexual offences	Other sexual offe...
2021	1	Lancashire	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	38	Sexual offences	Other sexual offe...
2021	1	Leicestershire	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	27	Sexual offences	Other sexual offe...
2021	1	Lincolnshire	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	11	Sexual offences	Other sexual offe...
2021	1	London, City of	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	1	Sexual offences	Other sexual offe...
2021	1	Merseyside	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	48	Sexual offences	Other sexual offe...
2021	1	Metropolitan	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	23	Sexual offences	Other sexual offe...
2021	1	Norfolk	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	38	Sexual offences	Other sexual offe...
2021	1	North Wales	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	28	Sexual offences	Other sexual offe...
2021	1	North Yorkshire	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	18	Sexual offences	Other sexual offe...
2021	1	Northumbria	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	20	Sexual offences	Other sexual offe...
2021	1	Northumbria	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	48	Sexual offences	Other sexual offe...
2021	1	Nottinghamshire	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	31	Sexual offences	Other sexual offe...
2021	1	South Wales	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	35	Sexual offences	Other sexual offe...
2021	1	South Yorkshire	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	17	Sexual offences	Other sexual offe...
2021	1	Staffordshire	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	27	Sexual offences	Other sexual offe...
2021	1	Suffolk	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	33	Sexual offences	Other sexual offe...
2021	1	Surrey	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	30	Sexual offences	Other sexual offe...
2021	1	Sussex, East	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	33	Sexual offences	Other sexual offe...
2021	1	Thames Valley	Sexual assault on...	Sexual offences	Other sexual offe...	17A)	50	Sexual offences	Other sexual offe...

Figure 31: Hive Query 2

A5. Using Spark

5.1 Create a Spark Dataframe

Save the Hive table into a Spark dataframe to perform further analysis.

```
| Using Spark
| Saving the SQL into a Spark Dataframe
|
| [ ] Criminaldataset = spark.sql("select * from NeighbourTable")
| [ ] Criminaldataset.show(5)
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| FyYearQuarter | ForceName | OffDesc | OffGroup | OffSubgroup | OffCode | NoOff | OffFenceCode | OffDescription |
|-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 2021/22 | 1 | Kent | Sexual assault on... | Sexual offences | Other sexual offe... | 17A | 81 | 17A Sexual assault on... | Sexual offences | Most serious sexu... | Sexual offences | Other sexual offe... |
| 2021/22 | 1 | Lancashire | Sexual assault on... | Sexual offences | Other sexual offe... | 17A | 38 | 17A Sexual assault on... | Sexual offences | Most serious sexu... | Sexual offences | Other sexual offe... |
| 2021/22 | 1 | Leicestershire | Sexual assault on... | Sexual offences | Other sexual offe... | 17A | 27 | 17A Sexual assault on... | Sexual offences | Most serious sexu... | Sexual offences | Other sexual offe... |
| 2021/22 | 1 | Lincolnshire | Sexual assault on... | Sexual offences | Other sexual offe... | 17A | 11 | 17A Sexual assault on... | Sexual offences | Most serious sexu... | Sexual offences | Other sexual offe... |
| 2021/22 | 1 | London, City of | Sexual assault on... | Sexual offences | Other sexual offe... | 17A | 1 | 17A Sexual assault on... | Sexual offences | Most serious sexu... | Sexual offences | Other sexual offe... |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

only showing top 5 rows

Figure 32: Create a New Spark Dataframe

5.2 Splitting on columns.

There are some inconsistencies with the data, and I attempt to clean it further using spark. In the figure below, I am splitting the Financial Year column by extracting values before and after \ into a separate column.

```
| Using Spark
| Creating a SparkDataframe for Dataset 1
| Working on Columns
| Working on Columns
| Rearrange the order of appearance
| Drop Columns
| Convert to date format
| Create a Spark Dataframe for Dataset 2
| Using Spark SQL
| Temporary Table
| Temporary Table for Dataset 1
| Temporary table for dataset 2
| Perform SQL Queries using Spark SQL
| SQL Syntax 1: Merge Datasets
| Handling Columns
| SQL Syntax 2: SqfSelect, SQL Count
| SQL Syntax 3: Locate Distinct Values
| SQL Syntax 4: Drop Column
|
| [ ] from pyspark.sql.functions import split, when
|
| # Split 'FyYear' into two columns
| CriminalId1 = CriminalId1.withColumn("Newfyear", split(CriminalId1['FyYear'], '/')[0])
| CriminalId1 = CriminalId1.withColumn("Endyear", split(CriminalId1['FyYear'], '/')[1])
|
| # Update 'ShortYear' column to replace '15' with '2015'
| #Cr = Cr.withColumn('Endyear', when(Cr['Endyear'] == '15', '2015').otherwise(Cr['Endyear']))
|
| # Show the result
| CriminalId1.show(5)
+-----+-----+-----+-----+-----+-----+-----+-----+
| FyYear|Quarter |ForceName |OffDesc |OffGroup |OffSubgroup |OffCode |NoOff |Newfyear|Endyear|
|-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 2021/15 | 1 |Avon and Somerset|Abandoning from ...|Miscellaneous cri...|Miscellaneous cri...| 88 | 3 | 2021 | 15 |
| 2021/15 | 1 |Avon and Somerset|Abuse of children...|Sexual offences|Other sexual offe...| 71 | 3 | 2021 | 15 |
| 2021/15 | 1 |Avon and Somerset|Abuse of position...|Sexual offences|Other sexual offe...| 73 | 3 | 2021 | 15 |
| 2021/15 | 1 |Avon and Somerset|Aggravated burgla...|Theft offences|Non-domestic burgl...| 51 | 3 | 2021 | 15 |
| 2021/15 | 1 |Avon and Somerset|Aggravated burgla...|Theft offences|Domestic burglary| 20 | 12 | 2021 | 15 |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

only showing top 5 rows

```
| [ ] CriminalId1.columns
| ['FyYear',
|  'Quarter',
|  'ForceName',
|  'OffDesc',
|  'OffGroup',
|  'OffSubgroup',
|  'OffCode',
|  'NoOff',
|  'Newfyear',
|  'Endyear']
```

Figure 33: Split Column

5.3 Rearrange Columns

After splitting the column and creating new columns, I rearrange the order of appearance of the columns on the table.

The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with navigation links like 'Using Spark', 'Creating a SparkDataframe for Dataset 1', 'Working on Columns', 'Rearrange the order of appearance' (which is highlighted), 'Drop Columns', 'Convert to date format', 'Using Spark SQL', 'Temporary Table', and 'Temporary Table for Dataset 1'. The main area contains Python code and its output:

```
from pyspark.sql.functions import col #importing the col function
order = ['Year','NewYear','EndYear','FQuarter','ForceName','OffDesc','OffGroup','OffSubgroup','OffCode','NoOff']
Criminald1 = Criminald1.select(*order, *(col(column) for column in Criminald1.columns if column not in order)) #rearrange the order of appearance
Criminald1.show(5)
```

Year	NewYear	EndYear	FQuarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	
2014/15	2014	15	1	Avon and Somerset	Abandoning from home	Miscellaneous crime	Miscellaneous crime		80	3
2014/15	2014	15	1	Avon and Somerset	Abuse of children...	Sexual offences	Other sexual offence		71	3
2014/15	2014	15	1	Avon and Somerset	Abuse of position...	Sexual offences	Other sexual offence		73	3
2014/15	2014	15	1	Avon and Somerset	Aggravated burlary...	Theft offences	Non-domestic burglary		31	3
2014/15	2014	15	1	Avon and Somerset	Aggravated burlary...	Theft offences	Domestic burglary		29	12

only showing top 5 rows

Figure 34: Reorder Column

5.4 Drop Columns

Now I attempt to drop the columns I don't need in the dataset. The columns not needed are dropped using .drop[] syntax.

The screenshot shows a Jupyter Notebook interface. The sidebar is identical to Figure 34. The main area contains Python code and its output:

```
Criminald1 = Criminald1.drop('Year')
Criminald1.show(5)
```

NewYear	EndYear	FQuarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	
2014	15	1	Avon and Somerset	Abandoning from home	Miscellaneous crime	Miscellaneous crime		80	3
2014	15	1	Avon and Somerset	Abuse of children...	Sexual offences	Other sexual offence		71	3
2014	15	1	Avon and Somerset	Abuse of position...	Sexual offences	Other sexual offence		73	3
2014	15	1	Avon and Somerset	Aggravated burlary...	Theft offences	Non-domestic burglary		31	3
2014	15	1	Avon and Somerset	Aggravated burlary...	Theft offences	Domestic burglary		29	12

only showing top 5 rows


```
Criminald1 = Criminald1.drop('EndYear')
Criminald1.show(5)
```

NewYear	FQuarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	
2014	1	Avon and Somerset	Abandoning from home	Miscellaneous crime	Miscellaneous crime		3	
2014	1	Avon and Somerset	Abuse of children...	Sexual offences	Other sexual offence		71	3
2014	1	Avon and Somerset	Abuse of position...	Sexual offences	Other sexual offence		73	3
2014	1	Avon and Somerset	Aggravated burlary...	Theft offences	Non-domestic burglary		31	3
2014	1	Avon and Somerset	Aggravated burlary...	Theft offences	Domestic burglary		29	12

only showing top 5 rows

Figure 35: Drop Column

5.5 Correlation

Correlations is being used to find the similarities between different variables in a dataset.

I use cor() in the pyspark data frame to check the correlation between different columns with numerical values. The heatmap is being used to identify the relationship between different numerical variables while the pair plot is used to identify the relationship between different pairs of variables.

```
↳ Correlations
↳ [100] Criminaldataset.corr()
<ipython-input-188-e76f73ed12>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
Criminaldataset.corr()
    Newyear  FQuarter  NoOff
Newyear  1.000000e+00 -7.198551e-14  0.05414
FQuarter -7.198551e-14  1.000000e+00 -0.001130
NoOff   3.481367e-02 -1.130277e-03  1.000000

↳ [109] # Assuming Criminaldataset is a PySpark DataFrame
correlation_matrix = Criminaldataset.corr()
<ipython-input-119-9d5982658>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
correlation_matrix = Criminaldataset.corr()
```

Figure 36: Correlations in the dataset

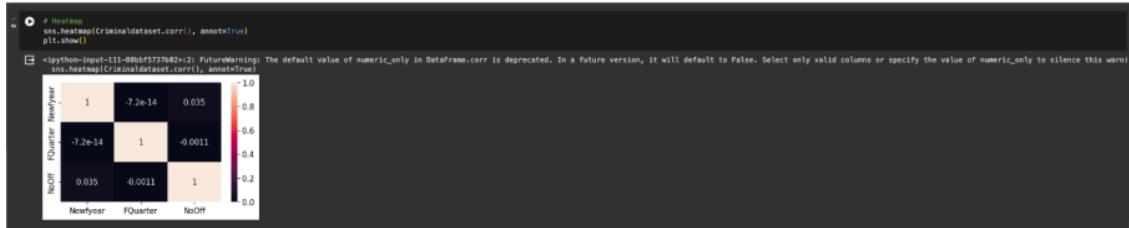


Figure 37: Numerical Correlation

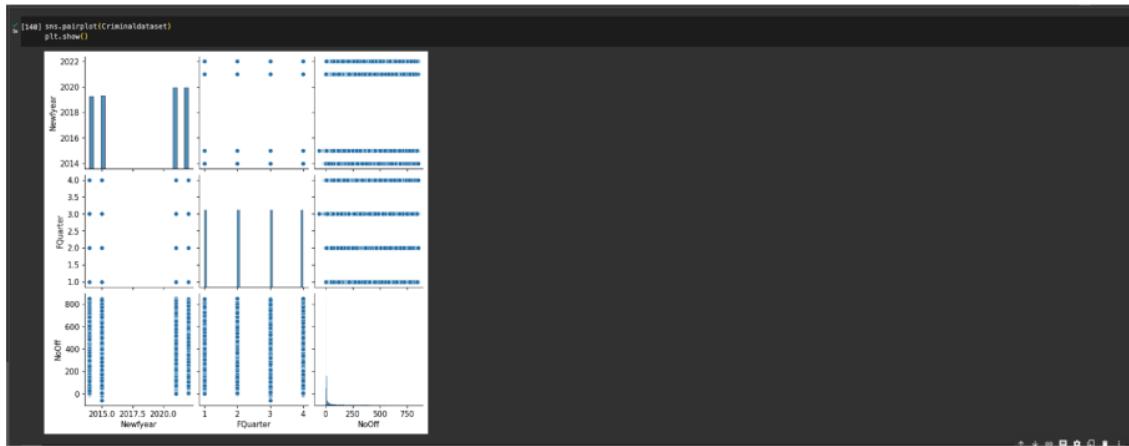


Figure 38: Bivariate Correlation

5.6 Outliers

Outliers are referred to as data points that significantly differ from other results.

```
Outliers
[ ] pip install matplotlib
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.7.2)
Requirement already satisfied: contours>=1.8.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.2.0)
Requirement already satisfied: keyring>=16.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: numpy<1.28 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.23.5)
Requirement already satisfied: packaging<22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (22.0)
Requirement already satisfied: pyparsing<3.0.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.4.0)
Requirement already satisfied: pytz<2022.2.8 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2022.1)
Requirement already satisfied: pygments>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.3)
Requirement already satisfied: python-dateutil<2.7.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six<2.12 in /usr/local/lib/python3.10/dist-packages (from python-dateutil<2.7.0>+matplotlib) (1.16.6)

[ ] import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import pandas as pd
from scipy import stats

[ ] plt.rcParams.update({'font.size': 18.5, 'figure.figsize': (5, 3)})
```

Figure 39: Installing Libraries for Outliers

5.6.1 Detect Outliers

To detect outliers, I plot a scatter plot diagram to plot a graph showing the outliers.

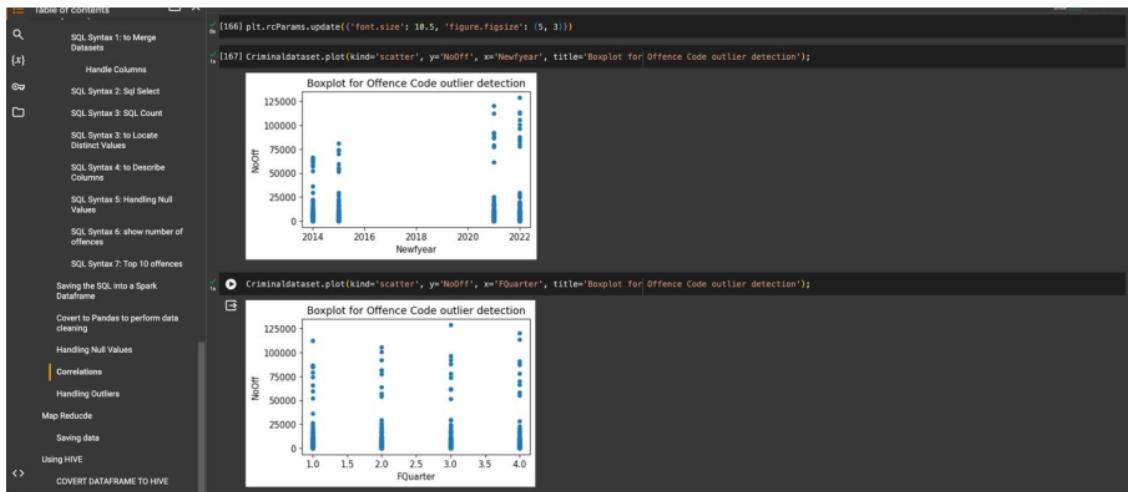


Figure 40: Detect Outliers

5.6.2 Handle Outliers

To handle outliers, I draw a graph showing the no of offenses against the financial years.

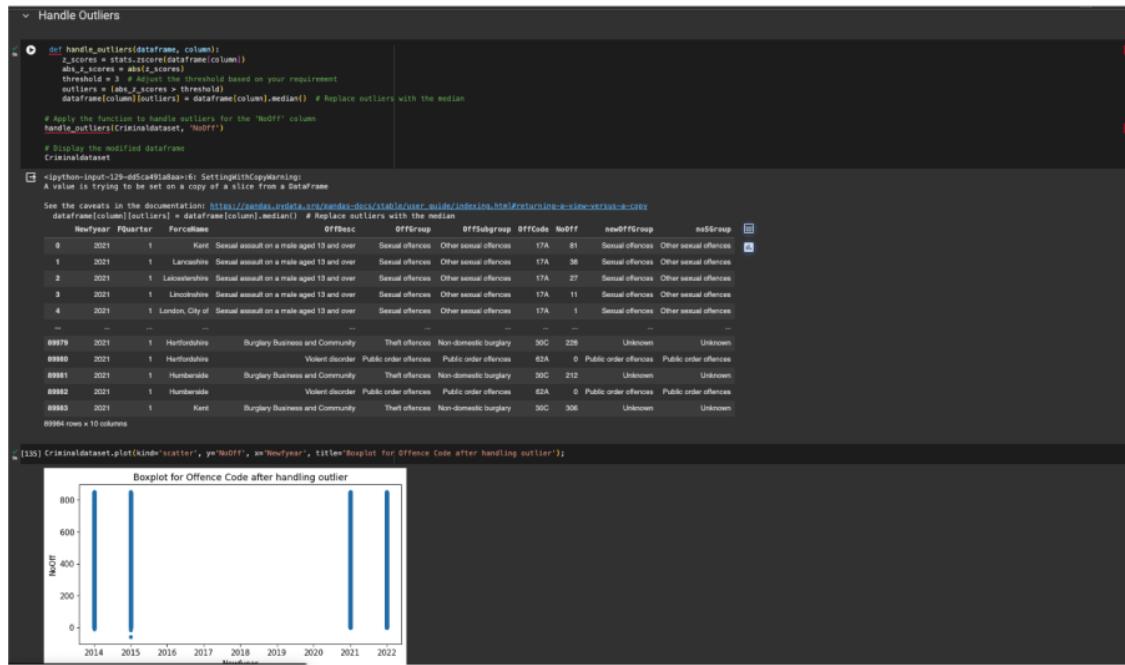
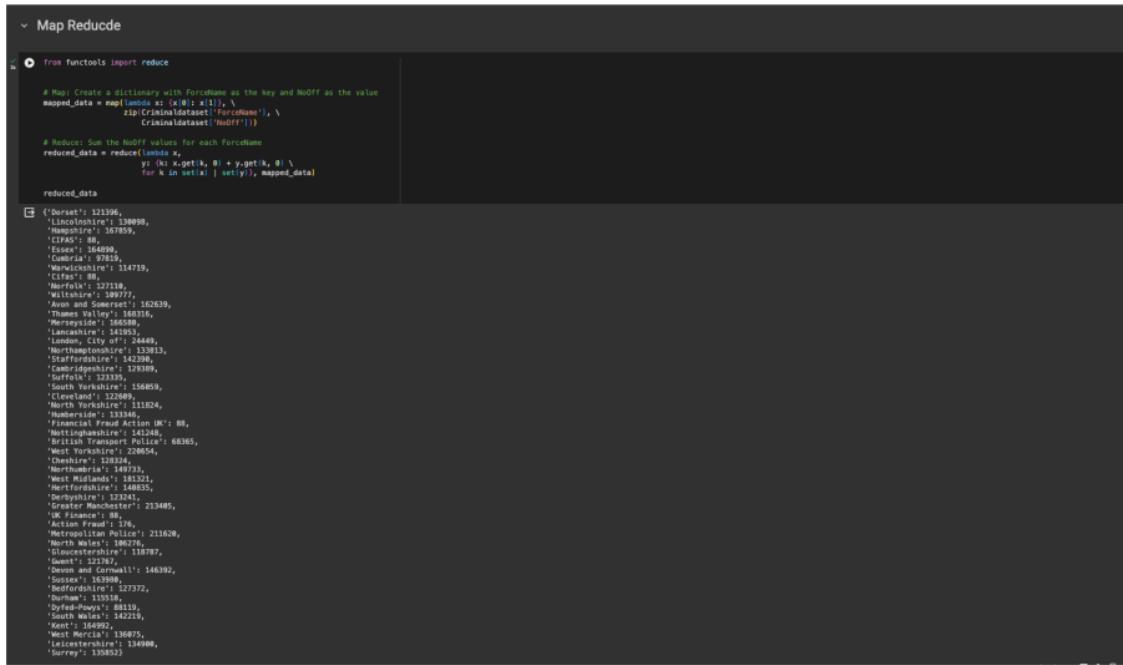


Figure 41: Handle Outliers

A6. Map Reduce

10
A MapReduce job usually splits the input data set into independent chunks which are processed by the map tasks in a completely parallel manner. Map reduce splits data into key-value pairs and reduces the data by counting the amount of time those key-value pairs appear.

To perform Map-Reduce for the dataset, I use the `functools` function and import the `reduce` functionality.



The screenshot shows a Jupyter Notebook cell with the title "Map Reduce". The code imports `functools` and `reduce`, defines a mapping function to create a dictionary of force names and their `NoOff` values, and then reduces the data to sum the `NoOff` values for each force name. The output is a large list of force names and their corresponding `NoOff` counts.

```
# Map: Create a dictionary with ForceName as the key and NoOff as the value
mapped_data = map(lambda x: (x[0], \n    zip(CrimeNoData['ForceName'], \n        CrimeNoData['NoOff'])), \n    CrimeNoData)

# Reduces Sum the NoOff values for each ForceName
reduced_data = reduced(mapped_data, \n    lambda x: (x[0], \n        x[1]), \n    lambda x: (x[0], \n        sum(x[1])), \n    lambda x: (x[0], \n        x[1]), \n    mapped_data)

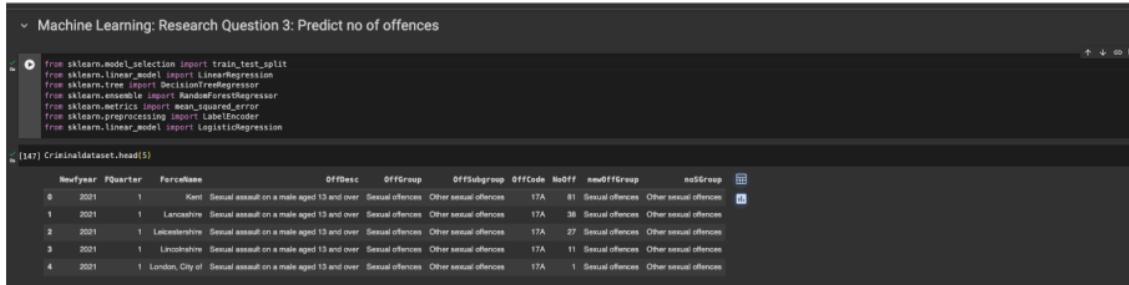
reduced_data
```

Force Name	NoOff
Berkshire	12196
Buckinghamshire	130098
Hampshire	167859
CIPAS	88
Isle of Wight	9098
Cumbria	97819
Merseyside	114719
Greater Manchester	121000
Norfolk	127110
Wiltshire	18077
Avon and Somerset	162639
Thames Valley	168316
Highways Agency	168346
London, City of	2440
Northumbria	188383
Staffordshire	142398
Cambridgeshire	128109
Teeside	113333
South Yorkshire	156459
Cleveland	32269
North Yorkshire	111824
Humberside	133346
Financial Fraud Action UK	88
Healthcare Commission	88
British Transport Police	68365
West Yorkshire	226554
Gloucestershire	138787
Greater Manchester	149733
West Midlands	188321
Humberside PCC	140493
Derbyshire	123241
Shropshire	233485
UK Finance	88
Action Fraud	178
Metropolitan Police	213628
North Wales	186276
Gloucestershire	138787
Leicester	113333
Devon and Cornwall	146392
Sussex	163988
Bedfordshire	127372
Durham	115518
Dyfed-Powys	88119
Isle of Wight	142119
Kent	164992
West Mercia	136975
Essex	134988
Surrey	130852

Figure 42: Reduce Functools

A7. Machine Learning

Installing all the necessary libraries needed to perform the predictions and all.



The screenshot shows a Jupyter Notebook cell with the following code:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVC, LinearSVC, NuSVC
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
```

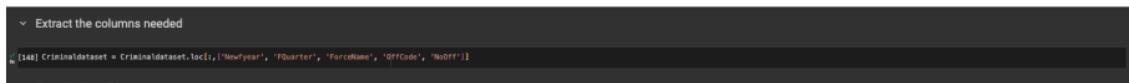
Below the code, the output shows the first 5 rows of the 'Criminaldataset' DataFrame:

	NewYear	FQuarter	ForceName	OffDesc	OffGroup	OffSubgroup	OffCode	NoOff	nsOffGroup	nsGroup
0	2021	1	Kent	Sexual assault on a male aged 13 and over	Sexual offences	Other sexual offences	17A	81	Sexual offences	Other sexual offences
1	2021	1	Lancashire	Sexual assault on a male aged 13 and over	Sexual offences	Other sexual offences	17A	38	Sexual offences	Other sexual offences
2	2021	1	Leicestershire	Sexual assault on a male aged 13 and over	Sexual offences	Other sexual offences	17A	27	Sexual offences	Other sexual offences
3	2021	1	Lincolnshire	Sexual assault on a male aged 13 and over	Sexual offences	Other sexual offences	17A	11	Sexual offences	Other sexual offences
4	2021	1	London, City of	Sexual assault on a male aged 13 and over	Sexual offences	Other sexual offences	17A	1	Sexual offences	Other sexual offences

Figure 43: Importing Libraries

7.1 Extract the dataset.

Extract the data needed using .loc[:,[]]



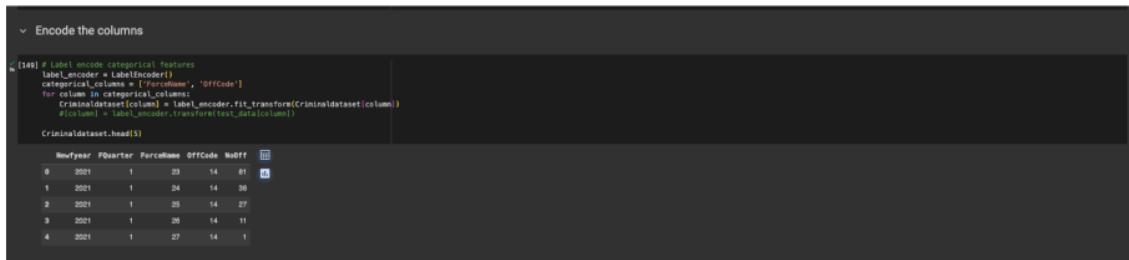
The screenshot shows a Jupyter Notebook cell with the following code:

```
Criminaldataset = Criminaldataset.loc[:, ['NewYear', 'FQuarter', 'ForceName', 'OffCode', 'NoOff']]
```

Figure 44: Extract Dataset

7.2 Encoding the columns.

Encoding is done in ML for string columns. This is done by providing each column with a unique variable so I can perform ML queries on the specified columns.



The screenshot shows a Jupyter Notebook cell with the following code:

```
label_encoder = LabelEncoder()
categorical_columns = ['ForceName', 'OffCode']
for column in categorical_columns:
    Criminaldataset[column] = label_encoder.fit_transform(Criminaldataset[column])
    #column = label_encoder.transform(test_data[column])
```

Below the code, the output shows the first 5 rows of the 'Criminaldataset' DataFrame after encoding:

	NewYear	FQuarter	ForceName	OffCode	NoOff
0	2021	1	23	14	81
1	2021	1	24	14	38
2	2021	1	25	14	27
3	2021	1	26	14	11
4	2021	1	27	14	1

Figure 45: Encode Categorical Variables

7.3 Split the dataset.

For us to perform ML queries I need to have a train and test dataset. Given the dataset provided is one large dataset, I have to write a code to split the dataset into a training dataset and a testing dataset.
21

To split the dataset, I use the train_test_split () as shown in the figure below.

```
Splitting dataset
In [158]: # Split the dataset into train and test sets (70/30)
train_data, test_data = train_test_split(criminaldataset, test_size=0.3, random_state=2)
```

Figure 46: Train_Test_Split

7.4 Features and Target

Here I set the features and target.

```
Separate feature from Target
In [159]: x = ['Neighbour', 'Quarter', 'Forgiveness', 'OffCode'] #feature
y = ['Murder'] #target
```

Figure 47: Features and Target

7.4.1 Highlight Features and targets for Train and testing data

The next step is to highlight the features and target for the test and training datasets.

```
Highlight train and test data for x and y
In [162]: X_train = train_data[x]
X_test = test_data[x]
y_train = train_data[y]
y_test = test_data[y]
```

Figure 48: Highlight Features and Target

7.5 Initiate Models

In the given dataset, I am trying to predict the number of offenses. Hence, the regression model is being used because I am working on getting continuous values.

```
✓ Initiate Regression Models to use

[153]: # Initialize regression models
linear_reg = LinearRegression()
decision_tree_reg = DecisionTreeRegressor()
random_forest_reg = RandomForestRegressor()
```

Figure 49: Initiate Models

7.5.1 Linear Regression

```
✓ Linear Regression

[154]: # Fit the models on the training data
linear_reg.fit(X_train, y_train)
LinearRegression()
[LinearRegression()]

[155]: # Make predictions on the test set
y_pred_linear = linear_reg.predict(X_test)
y_pred_linear
array([58.88724369,
       48.55427137,
       53.55789367,
       ...,
       92.84798481,
       101.88888889,
       76.7582398])
```

Figure 50: Linear Regression

7.5.2 Decision Tree

```
✓ Decision Tree

[156]: #Fit the model on the training data
decision_tree_reg.fit(X_train, y_train)
DecisionTreeRegressor()
[DecisionTreeRegressor()]

[157]: #Make predictions on the test set
y_pred_tree = decision_tree_reg.predict(X_test)
y_pred_tree
array([39.,  4.,  0., ...,  6.,  1., 33.])
```

Figure 51: Decision Tree

7.5.3 Random Forest

```
Random Forest
[1]: #Fit the model on the training dataset
random_forest_regr.fit(X_train, y_train)
<ipython-input-58-11b9c95c52d8>:2: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  random_forest_regr.fit(X_train, y_train)
  - RandomForestRegressor()
  RandomForestRegressor()

[159]: Make predictions on the test data
y_pred_forest = random_forest_regr.predict(X_test)
y_pred_forest
array([44.22, 3.15, 3.16, ..., 8.97, 1.81, 35.1])
```

Figure 52: Random Forest

31

7.6 Mean Square Error

The Mean Square Error (MSE) calculation is being done to check for the preferred regression model to use with the least error.

```
Calculate the Mean Square Error
[1]: # Calculate mean squared errors
from sklearn import metrics
mse_linear = metrics.mean_squared_error(y_test, y_pred_linear)
mse_tree = metrics.mean_squared_error(y_test, y_pred_tree)
mse_forest = metrics.mean_squared_error(y_test, y_pred_forest)

# Display mean squared errors
print("Mean Squared Error (Linear Regression):", mse_linear)
print("Mean Squared Error (Decision Tree):", mse_tree)
print("Mean Squared Error (Random Forest):", mse_forest)

[1]: Mean Squared Error (Linear Regression): 19456.7259734888
Mean Squared Error (Decision Tree): 5612.9394559946
Mean Squared Error (Random Forest): 3981.845873717575
```

Figure 53: Mean Square Error (MSE).

7.4 Plot Graph

I attempt to plot a graph to capture the MSE calculation done in the figure above.

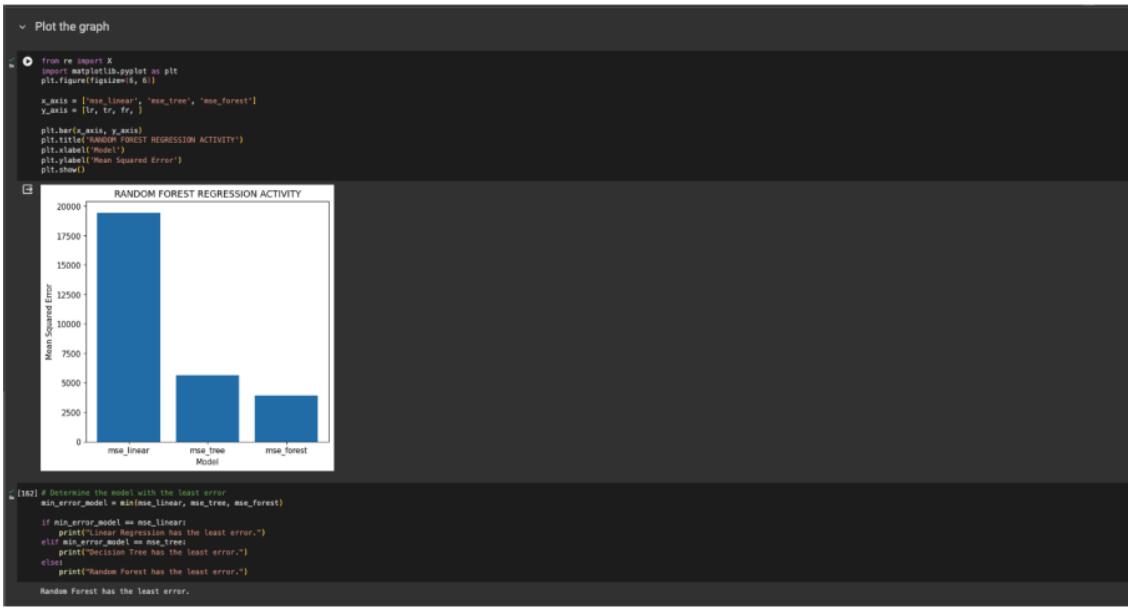


Figure 54: Least Error Graph

A8. Visualization

Data Visualization refers to the graphical representation of information and data with charts, graphs, maps, etc. In big data, data visualization refers to the graphical representation of data for an informed decision-making process (Kirk, 2019). The figure below shows a graphical representation of the datasets provided over some time.

8.1 Visualization with Power BI

In the figure below, Power BI is used to illustrate the research questions.

From the figure below, we can determine the total number of arrests to the nearest whole number, the total number of distinct forces involved, identify the financial year with the highest crime rate, and also identify the force with the highest number of arrests. Further analysis shows a detailed breakdown of the research questions, thereby enhancing comprehension and facilitating decision-making processes.

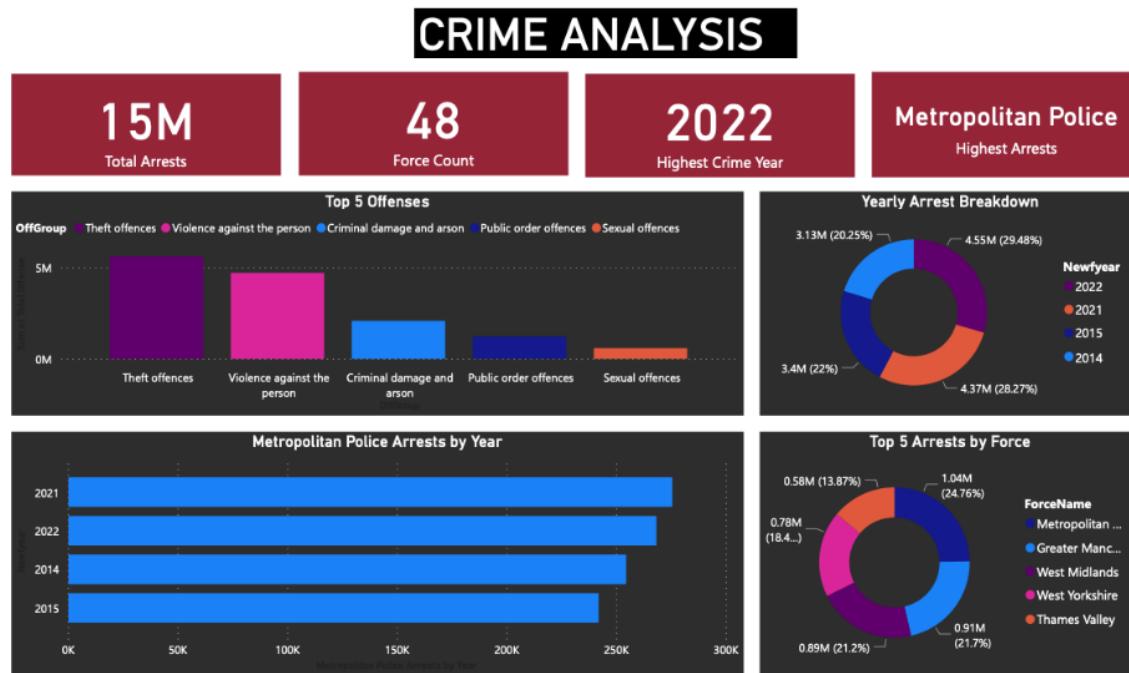


Figure 55: Crime analysis with Power Bi

8.2 Visualization with Tableau

Tableau is used for visualizing the dataset below using the story format of Tableau. Story 1 named Offense Analysis shows a breakdown of the total number of offenses against the financial year, offense group, and subgroup.

Crime Analysis_Assignment 2

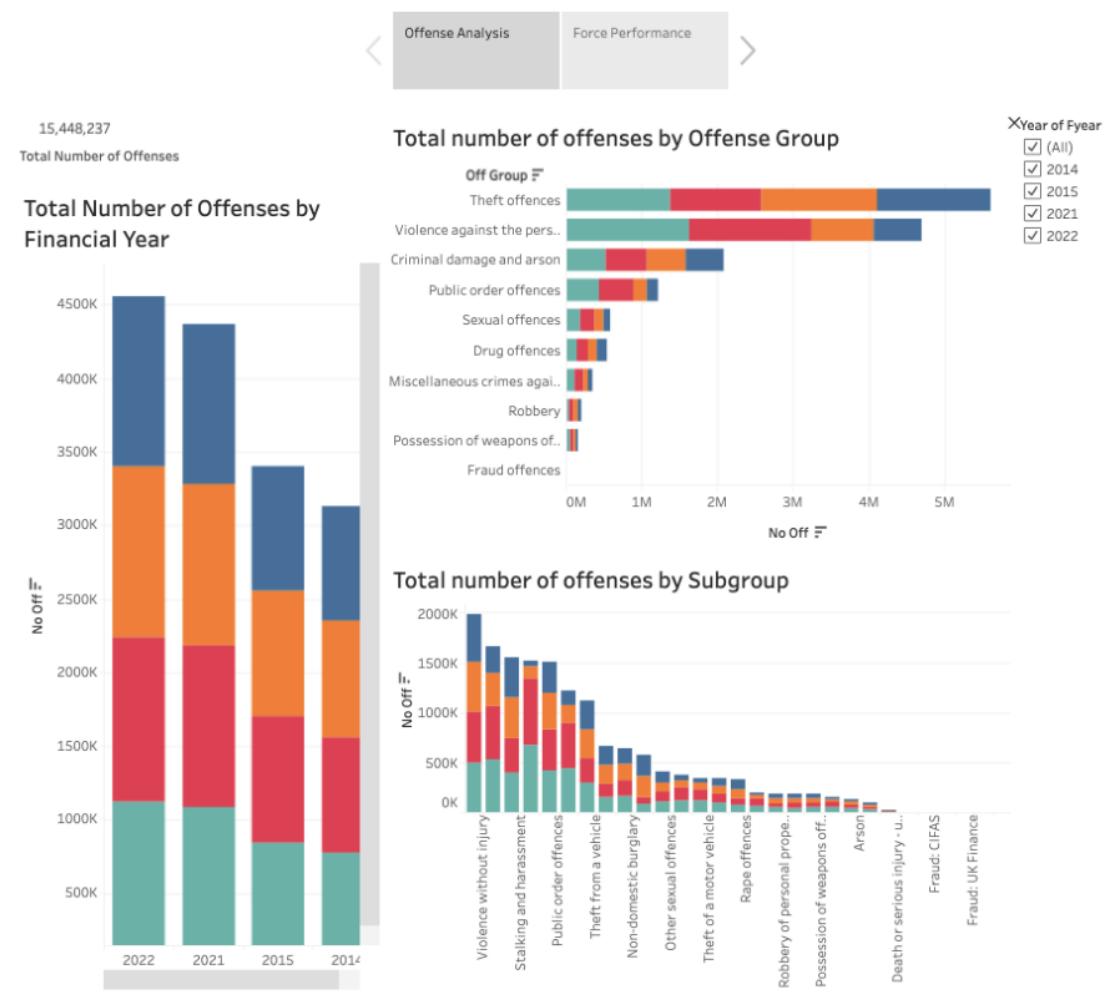


Figure 56: Offense Analysis

Story 2 named Force Performance shows a breakdown of how the forces performed before the changes made in 2017 and after the changes made in 2017.

Crime Analysis_Assignment 2

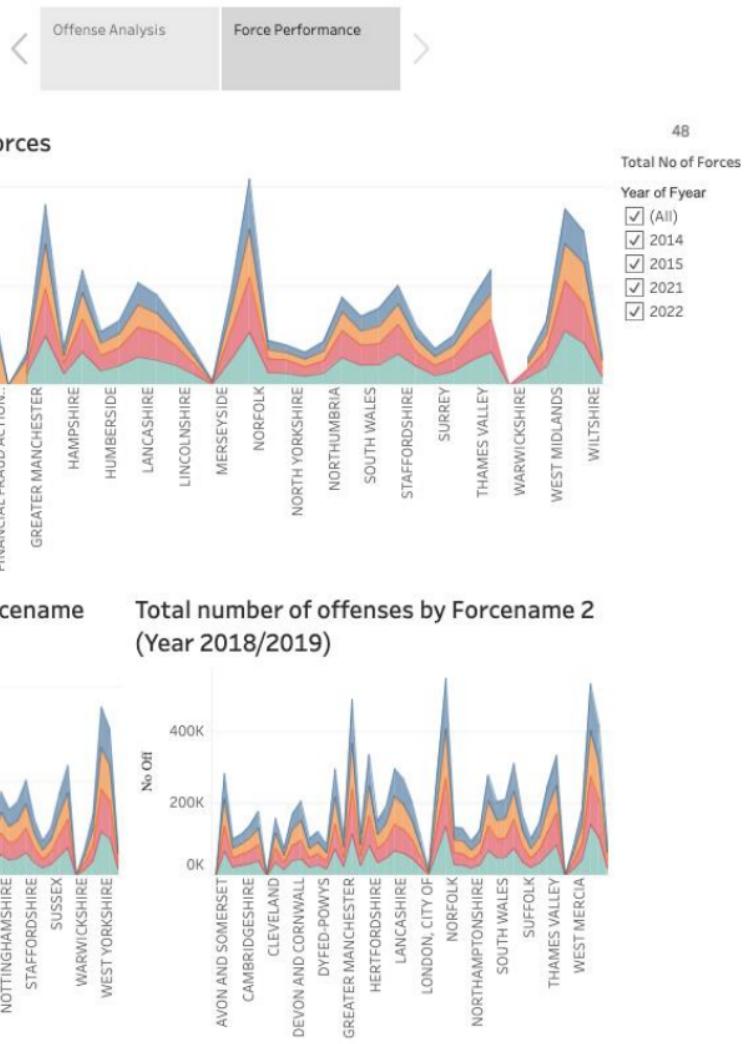


Figure 57: Force Analysis

Section B: Crime Analysis Report

B1. Analysis Answering Research Questions

Table 1: Analysis answering research questions.

Research Question	CHALLENGES	TECHNOLOGY USED	IMPACT
1. What are the most common types of offenses recorded in the database? What are the top five offenses in the dataset and what relevance does that cover in the criminal analysis?			
What are the most common types of offenses recorded in the database?	The extensive volume of data poses a challenge, as the substantial amount collected makes it time-intensive to meticulously review each entry and precisely identify the various offense types as well as the top ten offenses within the dataset.	<ul style="list-style-type: none">• Spark• SQL Distinct• SQL Count• SQL Select• SQL Order By• DESC Limit	<ul style="list-style-type: none">• Violence against the person• Theft offences• Sexual offences• Robbery• Public order offences <ul style="list-style-type: none">• Theft Offence• Violence against the person• Fraud Offences• Criminal damage offences• Public order offences
What are the top five offenses in the dataset and what relevance does that cover in the criminal analysis?			
2. What force made the most arrests across all the financial years and in what financial year did they make the most arrest?			

What force made the most arrest?	Given the substantial volume of collected data, specialized tools are essential to efficiently identify all distinct law enforcement agencies and calculate the total number of offenses committed each year. This process is crucial for promptly discerning the law enforcement agency with the highest arrest rates.	<ul style="list-style-type: none"> • Hive SQL Syntax • SQL Sum • SQL Order By 	<ul style="list-style-type: none"> • Metropolitan Police
What financial year did they make the most arrest?			<ul style="list-style-type: none"> • 2022/23
3. Are there any inconsistencies or anomalies in the data that need to be addressed to improve the data integrity? What technologies were adopted to ensure data reliability and robustness to provide meaningful contributions?			

<p>Are there any inconsistencies or anomalies in the data that need to be addressed to improve the data integrity?</p> <p>What technologies were adopted to ensure data reliability and robustness to provide meaningful contributions?</p>	<p>After scrutinizing the provided dataset, discrepancies were identified, introducing the potential for negative or inaccurate results.</p> <p>Additionally, the execution of calculations on the data may have led to erroneous decision-making.</p>	<ul style="list-style-type: none"> • Pandas • Spark • SQL • Scatter plot • Heatmaps • Map Reduce 	<ul style="list-style-type: none"> • Splitting columns • Drop columns/rows. • Renaming columns • Change data types. • Handle Missing Values • Handling Outlier • Reduce Functools <p>1.</p>
<p>4. What analytical approach proves most effective in predictive modelling for anticipating the frequency and nature of offenses, and how can this method be optimally applied to enhance proactive law enforcement strategies?</p>			
<p>What analytical approach proves most effective in predictive modelling for anticipating the frequency and nature of offenses, and how can this method be optimally applied to enhance proactive law enforcement strategies?</p>	<p>There was no way to accurately predict the frequency and nature of offenses.</p>	<ul style="list-style-type: none"> • Regression Model/ Classification Model • Linear Regression • Decision Tree • Random Forest 	<p>Regression model is being used in this calculation because we are working on Number of offenses and are thus predicting a continuous variable.</p>

B2. Discussions and Research Findings

In this section, the report addresses the syntax run to provide answers to the research questions.

2. What are the most common types of offenses recorded in the database? What are the top five offenses in the dataset and what relevance does that cover in the criminal analysis?

```
[128]: spark.sql("""SELECT OffGroup, SUM(NoOff) AS TotalOffences \
    FROM Newdataset GROUP BY OffGroup ORDER BY TotalOffences Desc """).show()
```

OffGroup[TotalOffences]	
Theft offences	6732679
Violence against ...	5978442
Fraud offences	3379836
Criminal damage a...	2189582
Public order offe...	1555195
Robbery	145711
Sexual offences	584369
Miscellaneous cril...	348252
Possession of wea...	232711
Possession of wrea...	154268


```
[129]: spark.sql("""SELECT Distinct OffGroup, SUM(NoOff) AS TotalOffences \
    FROM Newdataset GROUP BY OffGroup ORDER BY TotalOffences Desc Limit 5 """).show()
```

OffGroup[TotalOffences]	
Theft offences	6732679
Violence against ...	5978442
Fraud offences	3379836
Criminal damage a...	2189582
Public order offe...	1555195

Figure 58: Research Question 1 Solution

The figure above provides a solution to the research question 1.

The most common types of offenses are:

- 13 • Violence against the person
- Theft offenses
- Sexual offenses
- Robbery
- Public order offense

The top five offenses are:

- Theft Offence 16
- Violence against the person

- Fraud Offenses
- Criminal damage offenses
- Public order offenses

Using this in crime analysis can help while performing predictive analysis to find the prevalent type of offense. It can also help government officials plan how to allocate resources and better tackle criminal organizations within an area.

3. What force made the most arrests across all the financial years and in what financial year did they make the most arrest?

```

↳ Research Question 2:
What force made the most arrest across all the financial years and in what financial year did they make the most arrest?

What force made the most arrest ?

(spark.sql("""SELECT ForceName, SUM(NdOff) AS TotalOffences
    FROM NeighbourTable GROUP BY ForceName ORDER BY TotalOffences DESC LIMIT 2""").show())
+-----+-----+
|ForceName|TotalOffences|
+-----+-----+
|Metropolitan Police| 2253113 |
|Action Fraud| 2257798 |
+-----+-----+

What Year did they make the most arrest ?

(spark.sql("""SELECT ForceName, Fyear, SUM(NdOff) AS TotalOffences
    FROM NeighbourTable GROUP BY ForceName, Fyear ORDER BY TotalOffences DESC Limit 2""").show())
+-----+-----+-----+
|ForceName|Fyear|TotalOffences|
+-----+-----+-----+
|Metropolitan Police|2022/23| 686921 |
|Metropolitan Police|2021/22| 638735 |
+-----+-----+-----+


What force made the most arrest across all the financial years?

(spark.sql("""SELECT ForceName, SUM(NdOff) AS TotalOffences FROM NeighbourTable
    WHERE Fyear = '2022/23' GROUP BY ForceName ORDER BY TotalOffences DESC Limit 2""").show())
(spark.sql("""SELECT ForceName, SUM(NdOff) AS TotalOffences FROM NeighbourTable
    WHERE Fyear = '2021/22' GROUP BY ForceName ORDER BY TotalOffences DESC Limit 2""").show())
(spark.sql("""SELECT ForceName, SUM(NdOff) AS TotalOffences FROM NeighbourTable
    WHERE Fyear = '2020/21' GROUP BY ForceName ORDER BY TotalOffences DESC Limit 2""").show())
(spark.sql("""SELECT ForceName, SUM(NdOff) AS TotalOffences FROM NeighbourTable
    WHERE Fyear = '2019/20' GROUP BY ForceName ORDER BY TotalOffences DESC Limit 2""").show())
+-----+-----+
|ForceName|TotalOffences|
+-----+-----+
|Metropolitan Police| 889922 |
|Metropolitan Police| 408051 |

+-----+-----+
|ForceName|TotalOffences|
+-----+-----+
|Metropolitan Police| 887815 |
|Action Fraud| 383644 |

+-----+-----+
|ForceName|TotalOffences|
+-----+-----+
|Metropolitan Police| 790885 |
|Cites| 297786 |

+-----+-----+
|ForceName|TotalOffences|
+-----+-----+
|Metropolitan Police| 748599 |
|Cites| 290965 |
+-----+-----+

```

Figure 59: Research Question 2 Solution

The metropolitan police force made the most arrests in all the financial years where most of the arrest was made in the year 2022/23.

4. Are there any inconsistencies or anomalies in the data that need to be addressed to improve the data integrity? What technologies were adopted to ensure data reliability and robustness to provide meaningful contributions?

The inconsistencies or anomalies identified in the dataset were addressed using different solution methods across the notebook. Duplicates were handled using pandas, Null values were handled, and columns were dropped using spark. The report then proceeds to show how to handle outliers and perform map-reduce functionalities using the Functools function.

5. What analytical approach proves most effective in predictive modeling for anticipating the frequency and nature of offenses, and how can this method be optimally applied to enhance proactive law enforcement strategies?

Given the predictive analysis being done is for several offenses, the regression model is adopted to predict the number of offenses and also determine the best model to use with the least number of errors involved.

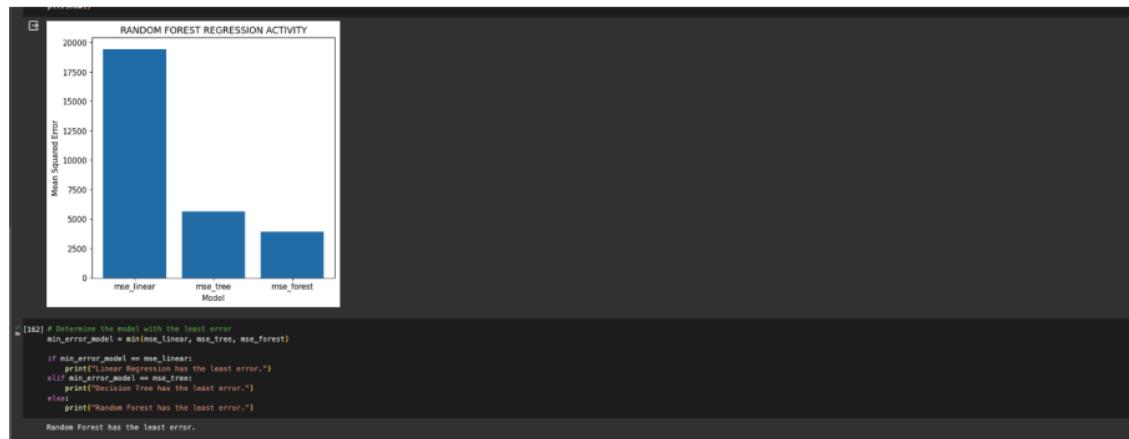


Figure 60: Regression Model Graph

B3. Issues In the Use of Big Data Analytics in Crime Analysis.

1. Data privacy and security concerns:

The increased collection of data and use in crime analysis has raised several ethical issues surrounding the way the data is being collected and used. The constant use of big data solutions in crime analysis calls for uneasiness as it raises concerns about the volatility of data being collected imposing there has to be a that separates the collection of data and violation of data privacy (Mayer-Schönberger and Cukier, 2013).

2. Bias and Discrimination Concerns:

Due to the historical crime data and algorithm prediction, there is an increased probability of bias and discriminatory behaviours towards a certain group. Bias can result from the group in charge of the algorithm or can also arise from the lack of diversity in the data collected (Diakopoulos, 2016).

3. Data Quality and Accuracy:

In crime analysis, data quality and accuracy play a vital role. The probability of a flawed prediction arises from the quality of data collected. Incoherent, incomplete, or inaccurate data can lead to misleading insights and therefore result in the ineffectiveness of the police force and can also lead to bias (Manyika *et al.*, 2011).

4. Ethical and Legal Implications:

The increased adoption of big data technologies in criminal analysis presents new opportunities to make law enforcement easier but also leads to different ethical considerations and legal implications of adopting the technologies. The registration of evidence to be used against offenders specifically evidence generated through the use of big data technologies can be challenged to be a breach of security and safety concerns and considered inadmissible (Hellman, 2020), and (Zarsky, 2016).

5. Data Transparency with Public Perception:

Big data solutions in crime analysis are being accepted by various forces and it is imperative to keep the public aware of the intention and management of the data being collected. Involving the communities in the developments in the introduction of big data, the methodologies, and its implications has become vital as it reduces the resistance and eases doubts (Lum, 2021) , and (Berk, 2021).

6. Security Risks:

Big data systems are prone to attacks and breaches which pose a significant threat to the validity and security of the data. As highlighted by (Dhillon and Torkzadeh, 2006), and (Schneier, 2016), there is an increased number of cybersecurity threats for big data systems. This can lead to tampering of data and inaccuracies of law enforcement in providing safety and justice when needed.

B4. Solution Using Various Big Data Technologies.

1. Data Privacy and Security Concerns:

To mitigate the security concerns highlighted, some techniques can be adopted such as anonymization, pseudonymization, and data encryption. These techniques can be adopted to protect against misuse of information. Also, frameworks and policies should be put in place to effective and efficient data usage (Solove, 2013).

2. Bias and Discrimination:

It should be made a framework for there to be regular checks on the algorithms used. To also ensure fairness, the data should be inclusive and there should be a responsible party managing the collection and use of the data (Selbst *et al.*, 2019).

3. Data Quality and Accuracy:

(Redman, 1996) talks about implementing rigorous practices to ensure the quality of data being used in crime analysis. Emphasizing the need for regular security audits, validation of data collected, and scrutiny to ensure standard procedures are being followed when doing data collection. In addition, we must ensure proper data cleansing to remove irregularities and address concerns that might come up. Machine learning algorithms should be incorporated into crime analysis for anomaly detection and rectifying inaccuracies in the data collected.

4. Ethical and Legal Implications:

Ethical guidelines that ensure data compliance with existing and new frameworks and principles should be put in place to ensure accountability and effectiveness of data thereby leading to accountability during the decision-making process. The frameworks adopted should ensure fairness and accountability across boards. It should incorporate collaboration between the respective stakeholders in law, technology, ethics, and community guidelines to ensure proper accountability.(Diakopoulos, 2016)

5. Data Transparency with Public Perception:

Communication in any community is key to development. There should be inclusivity and effective communication in the decision-making process. This is to ensure that information is passed across within the community about how the data will be collected and how it will be used. Proper communication further eases the doubts, privacy, and ethical concerns the public might have.

6. Security Risks:

Due to the increased number of cyberthreats in bid data, security measures should be put in place to ensure data protection. Cybersecurity measures should be introduced, the personnel should be educated, and informed on best practices to follow.(Schneier, 2016)

Conclusion

Big data utilization has become a major component of the crime analysis process. Different sectors have been adopting big data technologies which has led to an increase in efficiency and effectiveness and the crime sector is no different. Big data technology has been adopted in crime analysis, from crime patterns and predictions and automating the fraud detection process.

This report adopts a CRISP_DM methodology to answer the research questions and also a systematic review methodology to examine the challenges and solutions associated with the introduction of big data technologies.²⁷

The document examines the datasets, highlights the issues associated with working with the datasets, and provides solutions adopted to handle the issues. The report emphasizes the importance of big data technologies in crime analysis bringing us to a conclusion that there are more benefits to the introduction of big data solution.

References

- ⁴ Berk, R.A. (2021) 'Artificial Intelligence, Predictive Policing, and Risk Assessment for Law Enforcement', *Annual Review of Criminology*, 4(1), pp. 209–237. Available at: <https://doi.org/10.1146/annurev-criminol-051520-012342>.
- ¹⁴ Bhandari, P. (2022) *Triangulation in Research | Guide, Types, Examples*, Scribbr. Available at: <https://www.scribbr.co.uk/research-methods/triangulation-in-research/> (Accessed: 7 January 2024).
- ⁶ Dhillon, G. and Torkzadeh, G. (2006) 'Value-focused assessment of information system security in organizations', *Information Systems Journal*, 16(3), pp. 293–314. Available at: <https://doi.org/10.1111/j.1365-2575.2006.00219.x>.
- ¹ Diakopoulos, N. (2016) 'Accountability in algorithmic decision making', *Communications of the ACM*, 59(2), pp. 56–62. Available at: <https://doi.org/10.1145/2844110>.
- ²² Hellman, D. (2020) 'MEASURING ALGORITHMIC FAIRNESS', *Virginia Law Review*, 106(4), p. 56.
- ¹⁷ Kirk, A. (2019) *Data visualisation: a handbook for data driven design*. 2nd edition. Los Angeles London New Delhi Singapore Washington DC Melbourne: SAGE.
- ⁵ Laura, M. (2021) *How to write the methods section of a systematic review*, Covidence. Available at: <https://www.covidence.org/blog/how-to-write-the-methods-section-of-a-systematic-review/> (Accessed: 5 November 2023).
- ⁸ Lum, C. (2021) 'Perspectives on Policing: Cynthia Lum', *Annual Review of Criminology*, 4(1), pp. 19–25. Available at: <https://doi.org/10.1146/annurev-criminol-093020-023221>.
- ³ Manyika, J. et al. (2011) *Big data: The next frontier for innovation, competition, and productivity | McKinsey*. McKinsey Global Institute. Available at: <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/big-data-the-next-frontier-for-innovation> (Accessed: 8 January 2024).
- ¹² Mayer-Schönberger, V. and Cukier, K. (2013) *Big data: a revolution that will transform how we live, work and think*. 1. publ. London: Murray.

- ² Page, M.J. et al. (2021) 'PRISMA 2020 explanation and elaboration: updated guidance and exemplars for reporting systematic reviews', *BMJ*, p. n160. Available at: <https://doi.org/10.1136/bmj.n160>.
- ¹⁹ Redman, T.C. (1996) *Data quality for the information age*. Boston: Artech House.
- ⁸ Schneier, B. (2016) *Data and Goliath: the hidden battles to collect your data and control your world*. First published as a Norton paperback 2016. New York London: W.W. Norton & Company.
- ¹ Selbst, A.D. et al. (2019) 'Fairness and Abstraction in Sociotechnical Systems', in *Proceedings of the Conference on Fairness, Accountability, and Transparency. FAT* '19: Conference on Fairness, Accountability, and Transparency*, Atlanta GA USA: ACM, pp. 59–68. Available at: <https://doi.org/10.1145/3287560.3287598>.
- ⁷ Solove, D.J. (2013) *Introduction: Privacy Self-Management and the Consent Dilemma*, *Harvard Law Review*. Available at: <https://harvardlawreview.org/print/vol-126/introduction-privacy-self-management-and-the-consent-dilemma/> (Accessed: 8 January 2024).
- ²⁰ W3Schools (no date) *SQL Tutorial*, W3Schools. Available at: <https://www.w3schools.com/sql/default.asp> (Accessed: 8 January 2024).
- ¹ Zarsky, T. (2016) 'The Trouble with Algorithmic Decisions: An Analytic Road Map to Examine Efficiency and Fairness in Automated and Opaque Decision Making', *Science, Technology, & Human Values*, 41(1), pp. 118–132. Available at: <https://doi.org/10.1177/0162243915605575>.

Abbreviations

A List of Abbreviations in this document include:

- AI - Artificial Intelligence
- ML - Machine Learning
- pd - Panda
- np - numerical library
- plt - machine learning library for plotting graph
- sns - seaborn
- os - operating system

ORIGINALITY REPORT



PRIMARY SOURCES

1	Submitted to University of Bolton Student Paper	2%
2	e-space.mmu.ac.uk Internet Source	1%
3	Submitted to University College Birmingham Student Paper	1%
4	dspace.cuni.cz Internet Source	1%
5	Submitted to The University of Manchester Student Paper	1%
6	www.researchgate.net Internet Source	1%
7	Submitted to University of Liverpool Student Paper	1%
8	Submitted to University of Melbourne Student Paper	<1%
9	ouci.dntb.gov.ua Internet Source	<1%

10	www.ijirset.com Internet Source	<1 %
11	easychair-www.easychair.org Internet Source	<1 %
12	www.emerald.com Internet Source	<1 %
13	www.gov.uk Internet Source	<1 %
14	www.mdpi.com Internet Source	<1 %
15	fdotwww.blob.core.windows.net Internet Source	<1 %
16	justice.ie Internet Source	<1 %
17	Submitted to City University Student Paper	<1 %
18	Submitted to Victorian Institute of Technology Student Paper	<1 %
19	ndl.ethernet.edu.et Internet Source	<1 %
20	www.citethisforme.com Internet Source	<1 %
21	Submitted to University of North Texas Student Paper	<1 %

22	export.arxiv.org Internet Source	<1 %
23	www.crystadigitalagency.com Internet Source	<1 %
24	Submitted to Softwarica College Of IT & E-Commerce Student Paper	<1 %
25	bds.sagepub.com Internet Source	<1 %
26	docplayer.biz.tr Internet Source	<1 %
27	edoc.unibas.ch Internet Source	<1 %
28	fairmlbook.org Internet Source	<1 %
29	repository.unsri.ac.id Internet Source	<1 %
30	slashcry.com Internet Source	<1 %
31	www.scribd.com Internet Source	<1 %
32	etheses.whiterose.ac.uk Internet Source	<1 %

Exclude quotes Off

Exclude bibliography Off

Exclude matches Off

FINAL GRADE

85 /100

GENERAL COMMENTS

Your submission demonstrates a strong performance, earning an excellent mark of 85. While your presentation was excellent, there were noted issues with time management that need improvement. Additionally, the lack of an introduction and background section hampers the overall structure of your work.

Your understanding of the problem scenario and current big data tools is commendable. However, to further enhance the quality of your work, consider organizing it with a logical flow and adhering to academic writing standards and professional report structure. These elements are essential for clarity and coherence in your presentation.

Furthermore, it's important to ensure that the best model is utilized in your analysis to optimize results. Paying attention to these aspects will help you achieve even greater success in future assignments. Keep up the good work, and continue refining your skills and techniques!

PAGE 1



Comment 1

???

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

PAGE 54

PAGE 55

PAGE 56

PAGE 57
