Chapter 3
Data Collection

3.1 Introduction

In this chapter, we will be discussing about our data collection process, starting from identifying the sources of data collection, to the thought process behind the creation of some of our sources of collection, and finally, ensuring data quality by analyzing the collected data and making changes appropriate to our needs. Now, a quick reminder that the data collected is solely of the city of **Karachi** as our project SafeNav aims to make travel safer for people in Karachi.

3.2 Identifying Sources of Data Collection

Since there were no standard datasets available online regarding street crime data in Pakistan or accessible elsewhere, we had no option but to collect data manually. We came upon deciding two main sources of data collection:

3.2.1 Google Forms

Initially, our group reached a general consensus on using google forms as a means of collecting data. Now we had to decide on the format of the google form i.e. what questions shall be included in the google form that serves our purpose and how do we ensure data integrity?

We created and named our google form “SafeNav Data Collection Survey” and decided to include the following questions;

a) Email: To ensure data integrity, we chose to ask people to provide their emails while filling out the form as opposed to asking phone numbers or CNIC as people often hesitate in providing the latter information and this could ultimately scare them away from filling out the form.

b) Age: We broke down age into two parts namely “13-24 years” and “25 years and above” respectively.

c) Gender: Broken down into “Male” and “Female” categories.

d) Have you experienced accidents or safety concerns during your travel: Option of “Yes” or “No” provided to the user.

e) Please provide nearby location/landmark of incident: A short answer text describing the nearby location of incident.

f) Please provide time: We have divided time of incident into four time zones namely “5 AM -12 PM”, “12 PM – 5 PM”, “5 PM – 8 PM” and “8 PM – 5 AM”. Time of incident has been divided into four parts so that the occurrence of crime at a particular location can be studied with respect to these time zones.

g) Please provide date: Date of incident given in mm/dd/yyyy format.

h) District: We have broken down the districts of Karachi into “Gulshan”, “Nazimabad”, “Karachi South”, “Orangi”, “Korangi”, “Malir” and “Keamari”. We have also provided an “Other” option that the user can select and type the district according to his/her own understanding and ease.

i) Crime: This is basically the crime type experienced and has been divided into

"Robbery", "Murder", "Dacoity", "Motor vehicle theft", "Burglary", "Harassment" and "Kidnapping/abduction" categories.

j) Are there specific locations in your area where you avoid going due to safety concerns? if yes, please provide area/landmark's name: The user is also provided with an option to provide any other location or locations he/she feels reluctant and unsafe going to. This helps in collecting more data regarding to hot crime areas as a user can provide additional unsafe locations other than the location already provided in our fifth question (please provide location) all in one submission of the form and doesn't have to fill the form twice.

k) Which mode of transportation do you primarily use for travel: Transportation modes broken down into "Car", Bus" and "Public transportation". User can select more than one option if he/she desires.

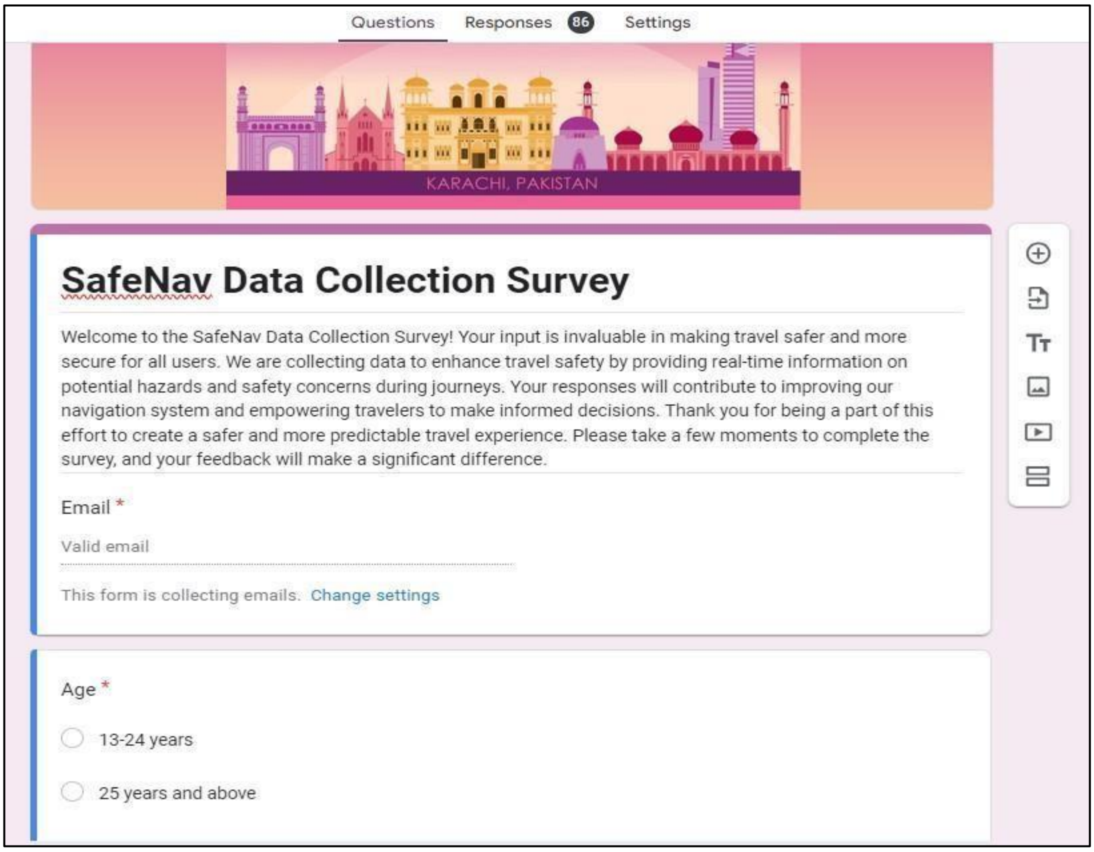
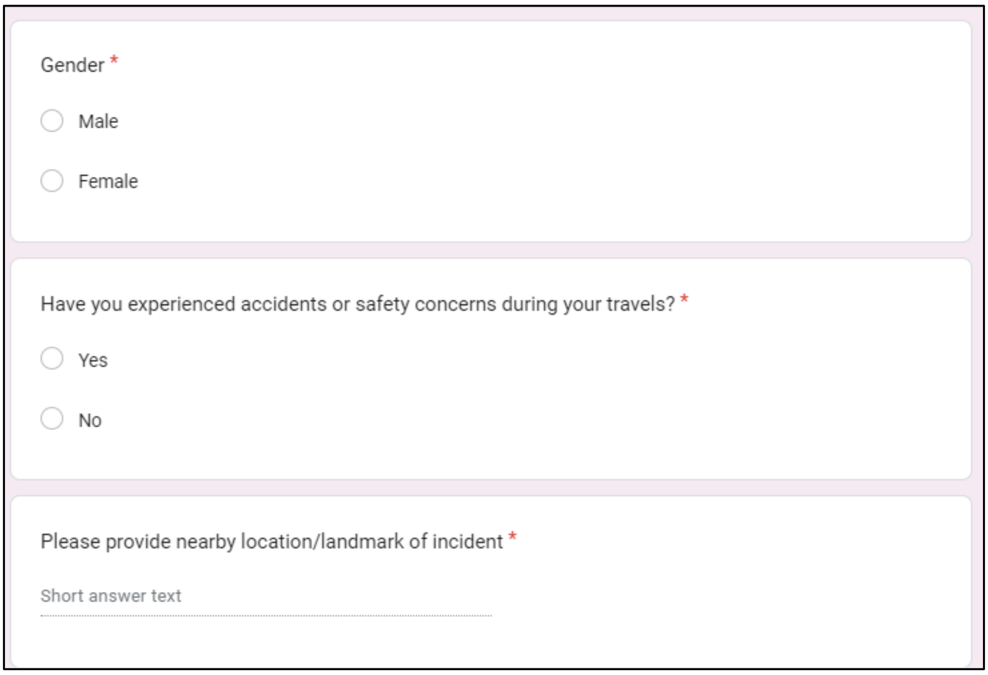


Fig 3.1: SafeNav Data Collection Survey Google form (1)



Gender *

Male

Female

Have you experienced accidents or safety concerns during your travels? *

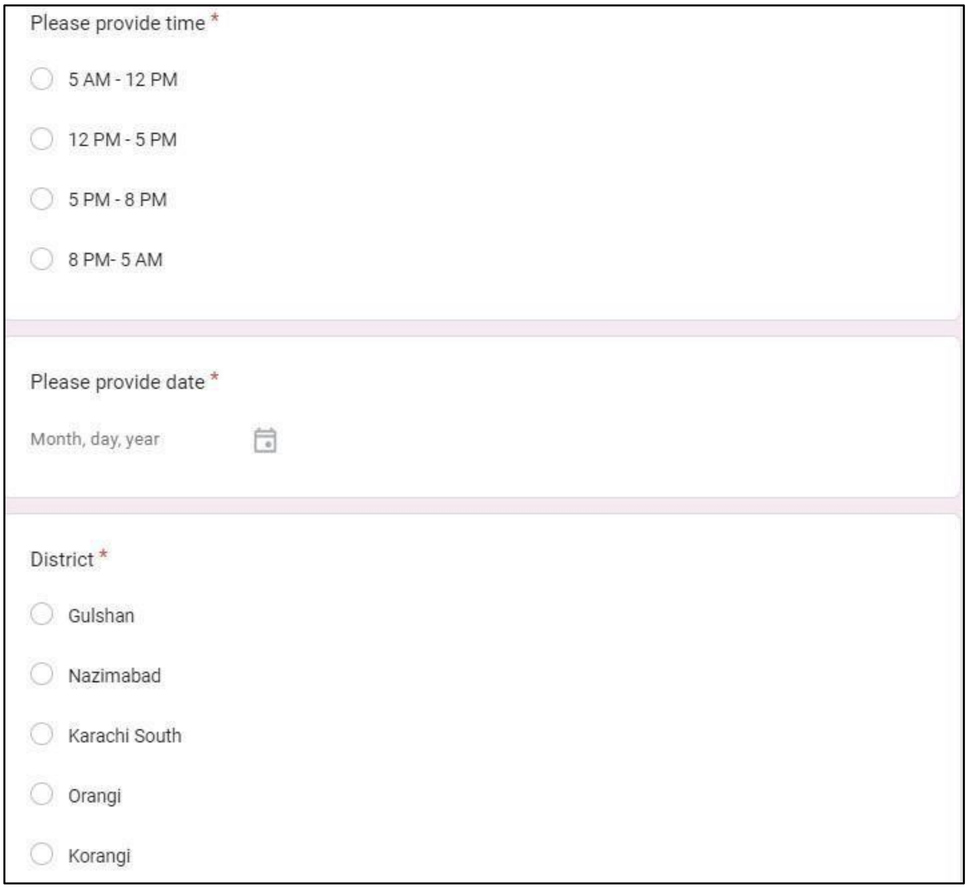
Yes

No

Please provide nearby location/landmark of incident *

Short answer text

Fig 3.2: SafeNav Data Collection Survey Google form (2)



Please provide time *

5 AM - 12 PM

12 PM - 5 PM

5 PM - 8 PM

8 PM- 5 AM

Please provide date *

Month, day, year

District *

Gulshan

Nazimabad

Karachi South

Orangi

Korangi

Fig 3.3: SafeNav Data Collection Survey Google form (3)


Korangi

Malir

Keamari

Other...

Crime *

Robbery

Murder

Dacoity

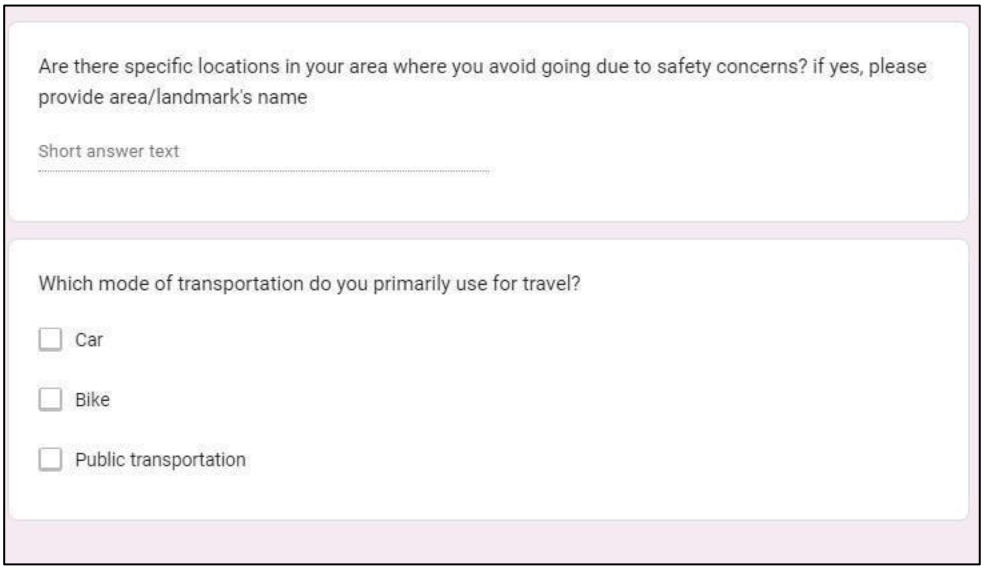
Motor vehicle theft

Burglary

Harassment

Kidnapping/abduction

Fig 3.4: SafeNav Data Collection Survey Google form (4)


Are there specific locations in your area where you avoid going due to safety concerns? if yes, please provide area/landmark's name

Short answer text

Which mode of transportation do you primarily use for travel?

Car

Bike

Public transportation

Fig 3.5: SafeNav Data Collection Survey Google form (5)

A total of 11 questions were included in our google form out of which all fields were mandatory to fill except for "Are there specific locations in your area where you avoid going due to safety concerns? if yes, please provide area/landmark's name" and

"Which mode of transportation do you primarily use for travel" fields. This google form was circulated to at most extent by all of our group members with the aim of maximum user responses.

3.2.2 Manual data entries

After studying the rate and amount of data responses received via google form, it was decided to collect more data from alternate sources. Since no street crime dataset applicable to Karachi was available online, we had to resort to manual data entries.

We obtained credible information from various areas such as **"Facebook Groups and Posts"**, multiple **"Websites"**, **"Youtube Videos"** and **"News papers"** etc. News channels such as Dawn, Ary, Geo, Hum, Dunya, Abbtakk and News International were referred for data collection. Renowned News papers such as Express Tribune were also referred. We reached a total of 500 responses after taking the necessary data quality steps as discussed in Section 3.3.

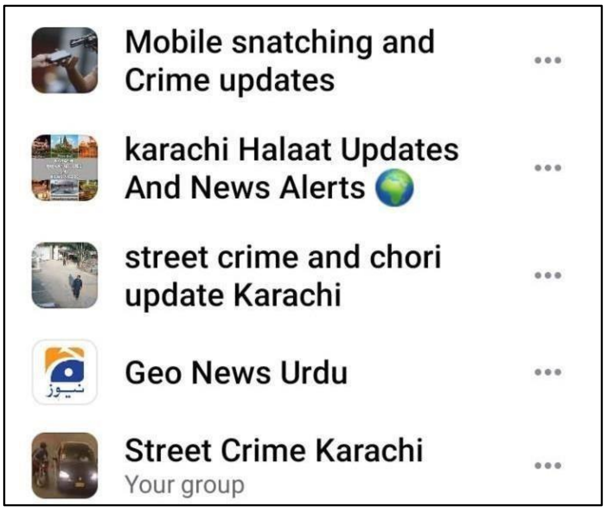
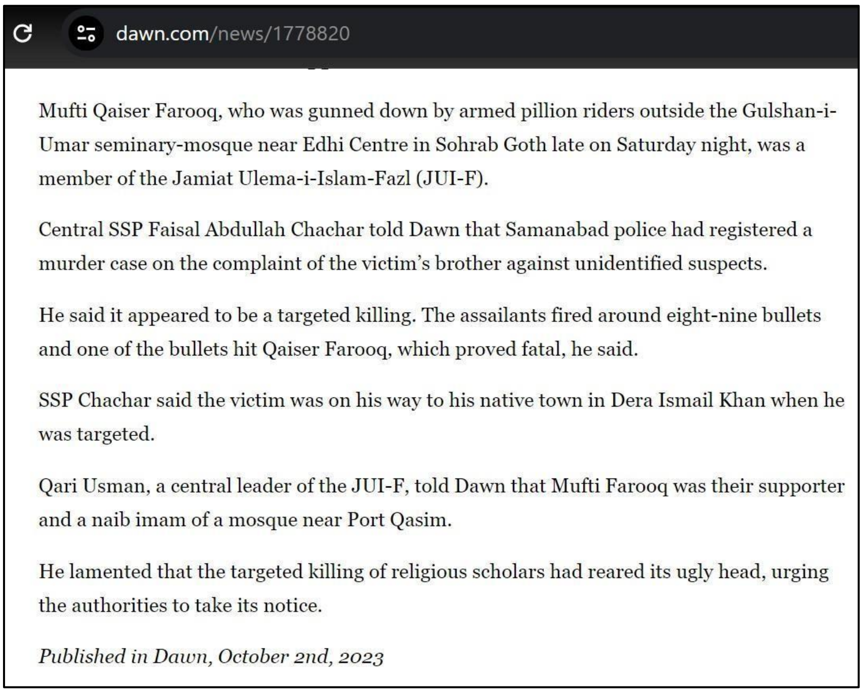


Fig 3.6: Facebook Groups used for manual data entries

Referred websites, news and youtube videos links are provided below:

- https://www.dawn.com/news/1746854
- https://www.dawn.com/news/1307257
- https://www.dawn.com/news/1778820
- https://www.dawn.com/news/1685997
- https://www.dawn.com/news/1742388/young-man-shot-dead-in-saeedabad
- https://humnews.pk/latest/karachi-mobile-snatching-gunpoint/
- https://www.thenews.com.pk/print/1140640-iphones-snatched-from-army-officer-recovered
- https://timesofislamabad.com/06-Jun-2018/nearly-280-crime-incidents-reported-across-dha-karachi-in-four-months
- https://tribune.com.pk/story/2446814/killer-of-online-cab-driver-arrested
- https://www.geo.tv/latest/126642-Street-crime-on-the-rise-in-Karachi-CPLC-identifies-danger-zones
- https://arynews.tv/cplc-identifies-13-locations-vulnerable-to-street-crimes-in-karachi/
- https://arynews.tv/man-shot-dead-by-unidentified-individuals-in-karachi/
- https://abbtakk.tv/en/robberies-increase-in-karachi-once-again-as-two-shot-dead-in-korangi/
- https://dunyanews.tv/en/Crime/676796-Robbers-injure-citizen-in-Karachis-Baldia-Town
- https://www.youtube.com/watch?v=Wc7vF1CbM3s
- https://www.youtube.com/watch?v=sppDgSov74M
- https://youtu.be/D0ARlT03P6w?si=jWE0A8vzQ7h0BrNR



dawn.com/news/1778820

Mufti Qaiser Farooq, who was gunned down by armed pillion riders outside the Gulshan-i-Umar seminary-mosque near Edhi Centre in Sohrab Goth late on Saturday night, was a member of the Jamiat Ulema-i-Islam-Fazl (JUI-F).

Central SSP Faisal Abdullah Chachar told Dawn that Samanabad police had registered a murder case on the complaint of the victim's brother against unidentified suspects.

He said it appeared to be a targeted killing. The assailants fired around eight-nine bullets and one of the bullets hit Qaiser Farooq, which proved fatal, he said.

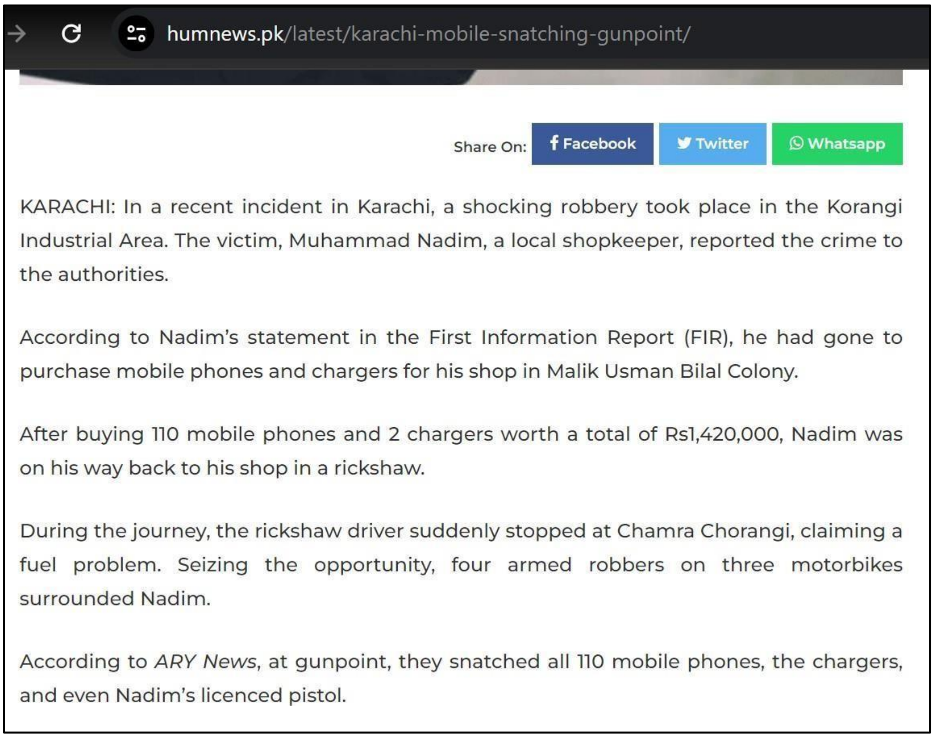
SSP Chachar said the victim was on his way to his native town in Dera Ismail Khan when he was targeted.

Qari Usman, a central leader of the JUI-F, told Dawn that Mufti Farooq was their supporter and a naib imam of a mosque near Port Qasim.

He lamented that the targeted killing of religious scholars had reared its ugly head, urging the authorities to take its notice.

Published in Dawn, October 2nd, 2023

Fig 3.7: Dawn news site



humnews.pk/latest/karachi-mobile-snatching-gunpoint/

Share On: Facebook Twitter Whatsapp

KARACHI: In a recent incident in Karachi, a shocking robbery took place in the Korangi Industrial Area. The victim, Muhammad Nadim, a local shopkeeper, reported the crime to the authorities.

According to Nadim's statement in the First Information Report (FIR), he had gone to purchase mobile phones and chargers for his shop in Malik Usman Bilal Colony.

After buying 110 mobile phones and 2 chargers worth a total of Rs1,420,000, Nadim was on his way back to his shop in a rickshaw.

During the journey, the rickshaw driver suddenly stopped at Chamra Chorangi, claiming a fuel problem. Seizing the opportunity, four armed robbers on three motorbikes surrounded Nadim.

According to *ARY News*, at gunpoint, they snatched all 110 mobile phones, the chargers, and even Nadim's licenced pistol.

Fig 3.8: Hum news site

Fig 3.9: Hums news channel

3.3 Ensuring Data Quality

The responses gathered in both google forms were viewed in Excel sheet (.xlsx) format. Our next step was to analyze our dataset constituting of 500 user responses. Main objective was to prepare the dataset and bring some sort of consistency in the data collected.

Initially, our dataset in excel was made up of 489 rows and 12 columns. The first column was "Timestamp" which provided us the time and date of user response submission. The next 11 columns were basically the answer to the 11 questions provided by the user in the Google form. Upon thoroughly reviewing the data collected, we decided to remove "Crime Experienced" column as it being 'Yes' or 'No' had no effect on the user response as long as the user had provided the location of incident. We also decided to create an additional entry in the excel sheet for those users who had provided more unsafe locations in the "Are there specific locations in your area where you avoid going due to safety concerns? if yes, please provide area/landmark's name" column. This column would then also be deleted and the location provided by the user in this column would be accommodated in the "Please provide nearby location/landmark of incident:" column in the additional entries for that respective user.

Upon reviewing the "District" column, we found out that many users had just typed the Town name or the Union Council name of the respective location instead of the District. We had facilitated the users by providing the 'Other' option in the "District" question in the Google form. To maintain consistency in data obtained, we decided to restrict District to seven types namely Karachi East, Karachi Central, Karachi South, Orangi, Korangi, Malir and Keamari. Hence, every row in dataset was checked for District value to see if it was correctly inputted with respect to the seven district types, else it was modified and corrected. We have followed https://www.wikiwand.com/en/Karachi_Division website for Karachi division of districts and district division of locations.

After analyzing the data, the last step was to add the latitude and longitude coordinates of the unsafe locations for each data entry. So, we appended 2 columns in the excel file named “Latitude” and “Longitude” respectively and started the process. We used Google Maps to input the longitude and latitude for each location in Decimal degrees (DD) format correct to 5 decimal places. There were some rows for which the location consisted of a large area such as North Nazimabad Block H, Pechs, Shahrah e Faisal, Johar Block 1. It wasn’t possible to find the longitude and latitude of these areas, a more specific location was required. These data entries were of no use to us and hence we ultimately decided to delete such rows from the excel file. Our dataset now consisted of 500 rows and 12 columns.

In this way, we aimed to bring about some consistency in the data collected. Data was analyzed thoroughly to ensure data quality in preparing the dataset.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Timestamp	Email Address	Age	Gender	Nearby location of incident	Time of Incident	Date of Incident	District	Crime	Transportation mode used	Latitude	Longitude
2	11/17/2023 19:32:28	mashalabbas58@gmail.com	13-24 years	Female	Johar Mor, Johar	5 AM - 12 PM	17/11/2023	Karachi East	Robbery	Public transportation	24.90466	67.1136
3	11/17/2023 19:33:27	s.m.jehanzeb127@gmail.com	13-24 years	Male	Dolmen Mall Hyderi	5 PM - 8 PM	17/11/2023	Karachi Central	Dacoity	Bike	24.93549	67.04049
4	11/17/2023 19:33:43	syedmaazalinaqvi@gmail.com	13-24 years	Male	Lasania service road near Bank (ATM), Gulshan-e-Iqbal	8 PM- 5 AM	02/11/2023	Karachi East	Robbery	Bike	24.90742	67.1099
5	11/17/2023 19:52:54	abdullahsiddiqi3@gmail.com	13-24 years	Male	Aziz bhatti park	8 PM- 5 AM	10/10/2020	Karachi East	Robbery	Bike	24.91315	67.09651
6	11/17/2023 19:58:24	amnasaeed383@gmail.com	13-24 years	Female	Near luckyone round about	12 PM - 5 PM	17/02/2023	Karachi East	Murder	Car	24.92603	67.09004
7	11/17/2023 19:58:24	amnasaeed383@gmail.com	13-24 years	Female	Johar Mor, Gulistan e johar	12 PM - 5 PM	17/02/2023	Karachi East	Murder	Car	24.90466	67.1136
8	11/17/2023 20:05:00	hayyan.ghauri@gmail.com	13-24 years	Male	Kamran Roundabout, Johar	8 PM- 5 AM	22/09/2023	Karachi East	Robbery	Car	24.92375	67.13778
9	11/17/2023 20:12:59	mohsinaliq69@gmail.com	13-24 years	Male	bait ul mukaram Masjid, University road Johar	5 PM - 8 PM	01/07/2023	Karachi East	Dacoity	Public transportation	24.90677	67.08339
10	11/17/2023 20:12:59	mohsinaliq69@gmail.com	13-24 years	Male	Nipa chowrangi	5 PM - 8 PM	01/07/2023	Karachi East	Dacoity	Public transportation	24.91768	67.09713
11	11/17/2023 20:17:05	maazrashid014@gmail.com	25 years and above	Male	Lalu Khait Pull 10 Number	8 PM- 5 AM	26/07/2023	Karachi Central	Robbery	Car	24.90957	67.0495
12	11/17/2023 20:17:05	maazrashid014@gmail.com	25 years and above	Male	Muhammad Hasan Road, North Nazimabad Block M	8 PM- 5 AM	26/07/2023	Karachi Central	Robbery	Car	24.94362	67.06052
13	11/17/2023 20:17:05	maazrashid014@gmail.com	25 years and above	Male	Near Masjid Faizan-e-Aisha, North Nazimabad Block I	8 PM- 5 AM	26/07/2023	Karachi Central	Robbery	Car	24.94855	67.04849
14	11/17/2023 20:26:27	sukaina4303559@cloud.neduet.edu.pk	13-24 years	Female	Near peoples chowrangi	8 PM- 5 AM	02/02/2017	Karachi Central	Robbery	Car	24.94777	67.06587
15	11/17/2023 20:26:27	sukaina4303559@cloud.neduet.edu.pk	13-24 years	Female	Zahra Classic, Garden East	8 PM- 5 AM	02/02/2017	Karachi South	Robbery	Car	24.87997	67.03434
16	11/17/2023 20:31:05	huzaifaazam24@gmail.com	13-24 years	Female	Dhamthal, Fb area	12 PM - 5 PM	14/11/2023	Karachi Central	Harassment	Public transportation	24.92688	67.06495
17	11/17/2023 20:31:05	huzaifaazam24@gmail.com	13-24 years	Female	Meezan bank	12 PM - 5 PM	03/11/2020	Orangi	Robbery	Car, Public transportation	24.9409	67.00019
18	11/17/2023 20:38:56	taibashahid14@gmail.com	13-24 years	Female	Karachi Institute of Heart Diseases (KIHD)	12 PM - 5 PM	17/10/2022	Karachi Central	Harassment	Public transportation	24.93334	67.07994
19	11/17/2023 20:41:35	hareemmalik727@gmail.com	13-24 years	Female	Near Faizan-e-Madina, University road	12 PM - 5 PM	05/06/2023	Karachi East	Robbery	Public transportation	24.89484	67.06083
20	11/17/2023 20:58:09	faseeha.rabbi07@gmail.com	13-24 years	Female	Sindh government hospital, liaquatabad	5 AM - 12 PM	03/06/2010	Karachi Central	Robbery	Car	24.91171	67.05095
21	11/17/2023 21:02:53	syedmhussain209@gmail.com	13-24 years	Male	Nipa flyover	8 PM- 5 AM	09/11/2023	Karachi East	Dacoity	Bike	24.91768	67.09713
22	11/17/2023 21:02:53	syedmhussain209@gmail.com	13-24 years	Male	Hansabad society	8 PM- 5 AM	25/10/2023	Karachi East	Dacoity	Bike	24.88043	67.18077
23	11/17/2023 21:03:39	umairali.ua5551450@gmail.com	13-24 years	Male	Al Asif square	8 PM- 5 AM	15/06/2022	Karachi East	Robbery	Bike	24.94962	67.09152
24	11/17/2023 21:06:44	zainabsalahuddin2994@gmail.com	13-24 years	Female	bait ul mukaram Masjid	5 AM - 12 PM	23/10/2023	Karachi East	Harassment	Public transportation	24.90677	67.08339
25	11/17/2023 21:13:32	tanvirmazna@gmail.com	25 years and above	Female	University of Karachi, nearby slums	8 PM- 5 AM	17/11/2023	Karachi East	Harassment	Public transportation	24.93472	67.10559
26	11/17/2023 21:17:51	kanzafarhan123@gmail.com	13-24 years	Female	Ke office, Surjani	8 PM- 5 AM	12/10/2023	Malir	Harassment	Car	25.02524	67.06119
27	11/17/2023 21:26:48	minhalh950@gmail.com	13-24 years	Male	Gulistan e Johar Police station	8 PM- 5 AM	05/07/2023	Karachi East	Robbery	Bike	24.92308	67.14768
28	11/17/2023 21:52:32	daniyalgoku4@gmail.com	13-24 years	Male	Near Aziz bhatti park	8 PM- 5 AM	17/11/2023	Karachi East	Robbery	Car	24.91315	67.09651
29	11/17/2023 21:52:32	daniyalgoku4@gmail.com	13-24 years	Male	Under NIPA bridge	8 PM- 5 AM	03/02/2023	Karachi East	Robbery	Bike	24.91768	67.09713
30	11/17/2023 21:59:45	muniba.rahman02@gmail.com	13-24 years	Female	Main Bus Stop of Malir Halt	12 PM - 5 PM	04/11/2023	Malir	Motor vehicle theft	Bike, Public transportation	24.88456	67.17535
31	11/17/2023 21:59:45	muniba.rahman02@gmail.com	13-24 years	Female	Near samama shopping mall, University Road	12 PM - 5 PM	04/11/2023	Karachi East	Motor vehicle theft	Bike, Public transportation	24.92835	67.11161
32	11/17/2023 22:17:18	atyabmanaqib567@gmail.com	13-24 years	Male	Do Darya Road, DHA PHASE 8	8 PM- 5 AM	23/12/2018	Karachi South	Robbery	Car	24.7506	67.08676
33	11/17/2023 22:20:56	naureenamir2001@gmail.com	25 years and above	Female	Sindbad	5 PM - 8 PM	17/11/2023	Karachi East	Burglary	Car	24.91585	67.0992
34	11/17/2023 22:24:10	naushaba1950@gmail.com	25 years and above	Female	Sir syed government Girls College Nazimabad No 1	8 PM- 5 AM	11/17/2023	Karachi Central	Burglary	Car	24.90207	67.02858
35	11/17/2023 22:24:10	naushaba1950@gmail.com	25 years and above	Female	Near Agha juice, Nazimabad No 1	8 PM- 5 AM	11/17/2023	Karachi Central	Burglary	Car	24.90512	67.02784
36	11/17/2023 22:25:33	ammarfar2006@gmail.com	13-24 years	Male	Sir Syed Government Girls College, Nazimabad no. 1	8 PM- 5 AM	10/09/2017	Karachi Central	Robbery	Car	24.90207	67.02858

Form Responses 1

Fig 3.10: Final form of Collected Data in Excel Sheets

3.4 Summary

In this chapter, we have discussed about our sources of data collection. Data has been collected manually as there weren't any datasets available online or elsewhere that could meet our requirements. Therefore, data collection is achieved by creation and circulation of a Google Form. The google form asks people 11 questions that are email, age, gender, crime experienced (yes or no), location of incident, time of incident, date of incident, district, crime type, any other locations where travel is unsafe, and transportation mode used. A total of 86 responses are collected from this form. Another Google form is created which is manually filled by our team in order to collect more data. Data is obtained from facebook groups, news papers, youtube videos and relevant websites. The latitude and longitude of unsafe locations are added to the excel sheet. Also, other relevant changes are made in order to maintain data consistency and ensure data quality. Our prepared dataset in its final form consists of 500 rows and 12 columns.

Chapter 4
Machine Learning

4.1 Introduction

In this chapter, we will discuss on how to employ machine learning to gain useful insights on the collected crime data and prepared dataset. Data preprocessing steps, feature engineering, selection of models and machine learning algorithms for implementation will be studied. A tabular and graphical comparison of models will also be implemented and the performance of implemented machine learning systems will be examined. All of our machine learning work is done on Jupyter Notebook.

4.2 Data Preprocessing Steps Applied

We have worked on the SafeNav Data Collection Survey (Responses) dataset. The dataset was converted from .xlsx to .csv format and read into a dataframe named df. Its size is 500 rows x 12 columns. This dataset consists of data collected from Google Forms and manual data entries.

```
#Loading fyp dataset
df=pd.read_csv("SafeNav Data Collection Survey (Responses) 500.csv")
pd.reset_option('display.max_rows')
pd.reset_option('display.max_columns')
df
```

	Timestamp	Email Address	Age	Gender	Nearby location of incident	Time of Incident	Date of Incident	District	Crime	Transportation mode used	Latitude	Longitude
0	11/17/2023 19:32:28	mashalabbas58@gmail.com	13-24 years	Female	Johar Mor, Johar	5 AM - 12 PM	11/17/2023	Karachi East	Robbery	Public transportation	24.90466	67.11360
1	11/17/2023 19:33:27	s.m.jehanzeb127@gmail.com	13-24 years	Male	Dolmen Mall Hyderi	5 PM - 8 PM	11/17/2023	Karachi Central	Dacoity	Bike	24.93549	67.04049
2	11/17/2023 19:33:43	syedmaazalinaqvi@gmail.com	13-24 years	Male	Lasania service road near Bank (ATM), Gulshan-...	8 PM- 5 AM	11/2/2023	Karachi East	Robbery	Bike	24.90742	67.10990
3	11/17/2023 19:52:53	abdullahsiddiqi3@gmail.com	13-24 years	Male	Aziz bhatti park	8 PM- 5 AM	10/10/2020	Karachi East	Robbery	Bike	24.91315	67.09651
4	11/17/2023 19:58:24	amnasaeed383@gmail.com	13-24 years	Female	Near luckyone round about	12 PM - 5 PM	2/17/2023	Karachi East	Murder	Car	24.92603	67.09004
...	...	...	...	...	...	...	...	...	...	...	...	...
495	5/11/2024 18:57:08	abdulsamikhan36912@gmail.com	13-24 years	Male	Sir Syed University	12 PM - 5 PM	4/15/2020	Karachi East	Dacoity	Bike	24.91626	67.09312
496	5/11/2024 18:58:08	abdulsamikhan36912@gmail.com	13-24 years	Male	Nazimabad near Imtiaz market	12 PM - 5 PM	4/15/2020	Karachi Central	Dacoity	Bike	24.91859	67.03210
497	5/11/2024 18:58:49	abdulsamikhan36912@gmail.com	25 years and above	Male	FM Public school Buffer zone	12 PM - 5 PM	4/14/2024	Karachi Central	Motor vehicle theft	Car, Bike	24.96151	67.06567
498	5/11/2024 18:58:49	abdulsamikhan36912@gmail.com	25 years and above	Male	Karachi Academy school (Azizabad)	12 PM - 5 PM	4/14/2024	Karachi Central	Dacoity	Car, Bike	24.91833	67.06817
499	11/17/2023 19:58:24	abdulsamikhan36912@gmail.com	25 years and above	Male	Johar Mor, Gulistan e johar	12 PM - 5 PM	2/17/2023	Karachi East	Murder	Car	24.90466	67.11360

500 rows × 12 columns

Fig 4.1: Reading dataset to dataframe

```
1 df.head(10)  #first 10 rows
```

	Timestamp	Email Address	Age	Gender	Nearby location of incident	Time of Incident	Date of Incident	District	Crime	Transportation mode used	Latitude	Longitude
0	11/17/2023 19:32:28	mashalabbas58@gmail.com	13-24 years	Female	Johar Mor, Johar	5 AM - 12 PM	11/17/2023	Karachi East	Robbery	Public transportation	24.90466	67.11360
1	11/17/2023 19:33:27	s.m.jehanzeb127@gmail.com	13-24 years	Male	Dolmen Mall Hyderi	5 PM - 8 PM	11/17/2023	Karachi Central	Dacoity	Bike	24.93549	67.04049
2	11/17/2023 19:33:43	syedmaazalinaqvi@gmail.com	13-24 years	Male	Lasania service road near Bank (ATM), Gulshan-...	8 PM- 5 AM	11/2/2023	Karachi East	Robbery	Bike	24.90742	67.10990
3	11/17/2023 19:52:54	abdullahsiddiqi3@gmail.com	13-24 years	Male	Aziz bhatti park	8 PM- 5 AM	10/10/2020	Karachi East	Robbery	Bike	24.91315	67.09651
4	11/17/2023 19:58:24	amnasaeed383@gmail.com	13-24 years	Female	Near luckyone round about	12 PM - 5 PM	2/17/2023	Karachi East	Murder	Car	24.92603	67.09004
5	11/17/2023 19:58:24	amnasaeed383@gmail.com	13-24 years	Female	Johar Mor, Gulistan e johar	12 PM - 5 PM	2/17/2023	Karachi East	Murder	Car	24.90466	67.11360
6	11/17/2023 20:05:00	hayyan.ghauri@gmail.com	13-24 years	Male	Kamran Roundabout, Johar	8 PM- 5 AM	9/22/2023	Karachi East	Robbery	Car	24.92375	67.13778
7	11/17/2023 20:12:59	mohsinaliq69@gmail.com	13-24 years	Male	bait ul mukaram Masjid, University road Johar	5 PM - 8 PM	7/1/2023	Karachi East	Dacoity	Public transportation	24.90677	67.08339
8	11/17/2023 20:12:59	mohsinaliq69@gmail.com	13-24 years	Male	Nipa chowrangi	5 PM - 8 PM	7/1/2023	Karachi East	Dacoity	Public transportation	24.91768	67.09713
9	11/17/2023 20:17:05	maazrashid014@gmail.com	25 years and above	Male	Lalu Khait Pull 10 Number	8 PM- 5 AM	7/26/2023	Karachi Central	Robbery	Car	24.90957	67.04950

Fig 4.2: First 10 rows of the dataset

```
1 df.info()     #printing info about df

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 12 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Timestamp                    500 non-null    object
 1   Email Address                500 non-null    object
 2   Age                          500 non-null    object
 3   Gender                       500 non-null    object
 4   Nearby location of incident  500 non-null    object
 5   Time of Incident             500 non-null    object
 6   Date of Incident             500 non-null    object
 7   District                     500 non-null    object
 8   Crime                        500 non-null    object
 9   Transportation mode used     500 non-null    object
 10  Latitude                     500 non-null    float64
 11  Longitude                    500 non-null    float64
dtypes: float64(2), object(10)
memory usage: 47.0+ KB
```

Fig 4.3: Dataset information

The quality of data and the useful information that can be derived from it directly affects the ability of our model to learn. To ensure that the data is free from impurities, we have applied the following preprocessing steps on the dataset:

4.2.1 Checking and removing duplicate rows

SafeNav Data Collection Survey (Responses) dataset consists of 12 columns namely Timestamp, Email Address, Age, Gender, Nearby location of incident, Time of Incident, Date of Incident, District, Crime, Transportation mode used, Latitude, and Longitude.

Now, let's discuss about our input/feature variables and output/target variable. Our main focus was on assigning a Crime Score to each unique location in the dataset according to a particular Time of Incident. So, "**Crime_Score**" will be our **target variable** which will be appended to the dataset later and its working will be discussed in detail in Sub-section 4.2.5. Therefore, there would be **three input variables** to our model namely **Time of Incident, Latitude and Longitude**.

We have used drop_duplicates() to remove duplicate rows in the dataset if any. No duplicate rows were found.

```
#removing duplicate rows, considering first duplicate row as unique
df.drop_duplicates(keep='first', inplace=True)
df=df.reset_index(drop=True)   #resetting index
df
```

	Timestamp	Email Address	Age	Gender	Nearby location of incident	Time of Incident	Date of Incident	District	Crime	Transportation mode used	Latitude	Longitude
0	11/17/2023 19:32:28	mashalabbas58@gmail.com	13-24 years	Female	Johar Mor, Johar	5 AM - 12 PM	11/17/2023	Karachi East	Robbery	Public transportation	24.90466	67.11360
1	11/17/2023 19:33:27	s.m.jehanzeb127@gmail.com	13-24 years	Male	Dolmen Mall Hyderi	5 PM - 8 PM	11/17/2023	Karachi Central	Dacoity	Bike	24.93549	67.04049
2	11/17/2023 19:33:43	syedmaazalinaqvi@gmail.com	13-24 years	Male	Lasania service road near Bank (ATM), Gulshan-...	8 PM- 5 AM	11/2/2023	Karachi East	Robbery	Bike	24.90742	67.10990
3	11/17/2023 19:52:53	abdullahsiddiqi3@gmail.com	13-24 years	Male	Aziz bhatti park	8 PM- 5 AM	10/10/2020	Karachi East	Robbery	Bike	24.91315	67.09651
4	11/17/2023 19:58:24	amnasaeed383@gmail.com	13-24 years	Female	Near luckyone round about	12 PM - 5 PM	2/17/2023	Karachi East	Murder	Car	24.92603	67.09004
...	...	...	...	...	...	...	...	...	...	...	...	...
495	5/11/2024 18:57:08	abdulsamikhan36912@gmail.com	13-24 years	Male	Sir Syed University	12 PM - 5 PM	4/15/2020	Karachi East	Dacoity	Bike	24.91626	67.09312
496	5/11/2024 18:58:08	abdulsamikhan36912@gmail.com	13-24 years	Male	Nazimabad near Imtiaz market	12 PM - 5 PM	4/15/2020	Karachi Central	Dacoity	Bike	24.91859	67.03210
497	5/11/2024 18:58:49	abdulsamikhan36912@gmail.com	25 years and above	Male	FM Public school Buffer zone	12 PM - 5 PM	4/14/2024	Karachi Central	Motor vehicle theft	Car, Bike	24.96151	67.06567
498	5/11/2024 18:58:49	abdulsamikhan36912@gmail.com	25 years and above	Male	Karachi Academy school (Azizabad)	12 PM - 5 PM	4/14/2024	Karachi Central	Dacoity	Car, Bike	24.91833	67.06817
499	11/17/2023 19:58:24	abdulsamikhan36912@gmail.com	25 years and above	Male	Johar Mor, Gulistan e johar	12 PM - 5 PM	2/17/2023	Karachi East	Murder	Car	24.90466	67.11360

500 rows × 12 columns

==> Established that there are no duplicate rows.

Fig 4.4: Removing duplicate rows

4.2.2 Removing null values

We have used isna().sum() syntax to find the count of null values in the 12 columns. The null count was zero for all columns therefore there were no null values.

```
missing=df.isna().sum()   #counting null values in dataset
missing
```

```
Timestamp                     0
Email Address                 0
Age                           0
Gender                        0
Nearby location of incident   0
Time of Incident              0
Date of Incident              0
District                      0
Crime                         0
Transportation mode used      0
Latitude                      0
Longitude                     0
dtype: int64
```

==> Established that there are no null values.

Fig 4.5: Removing null values

4.2.3 Checking and removing irrelevant attributes

During this stage, we have removed Timestamp, Email Address and Date of Incident columns since it doesn't really seem to affect the way in which a machine would learn data. Age, Gender, District, Crime, Transportation mode used features will also be removed, but after Data Visualization so we can study the underlying phenomenon that is generating the data. It was decided that all Crime types will hold equal weightage. Nearby location of incident feature will be removed after assigning the target variable “Crime_Score” as this feature will ease us in verifying the values of Crime Score by comparing it from the physical record maintained in copy. After dropping 3 columns, our dataset now consists of 500 rows and 9 columns.

```
#dropping irrelevant columns from dataset
df.drop(["Timestamp"], axis=1, inplace=True)
df.drop(["Email Address"], axis=1, inplace=True)
df.drop(["Date of Incident"], axis=1, inplace=True)
```

```
df=df.reset_index(drop=True)   #resetting index to default integer index
df   #changes incorporated permanently
```

	Age	Gender	Nearby location of incident	Time of Incident	District	Crime	Transportation mode used	Latitude	Longitude
0	13-24 years	Female	Johar Mor, Johar	5 AM - 12 PM	Karachi East	Robbery	Public transportation	24.90466	67.11360
1	13-24 years	Male	Dolmen Mall Hyderi	5 PM - 8 PM	Karachi Central	Dacoity	Bike	24.93549	67.04049
2	13-24 years	Male	Lasania service road near Bank (ATM), Gulshan-...	8 PM- 5 AM	Karachi East	Robbery	Bike	24.90742	67.10990
3	13-24 years	Male	Aziz bhatti park	8 PM- 5 AM	Karachi East	Robbery	Bike	24.91315	67.09651
4	13-24 years	Female	Near luckyone round about	12 PM - 5 PM	Karachi East	Murder	Car	24.92603	67.09004
...	...	...	...	...	...	...	...	...	...
495	13-24 years	Male	Sir Syed University	12 PM - 5 PM	Karachi East	Dacoity	Bike	24.91626	67.09312
496	13-24 years	Male	Nazimabad near Imtiaz market	12 PM - 5 PM	Karachi Central	Dacoity	Bike	24.91859	67.03210
497	25 years and above	Male	FM Public school Buffer zone	12 PM - 5 PM	Karachi Central	Motor vehicle theft	Car, Bike	24.96151	67.06567
498	25 years and above	Male	Karachi Academy school (Azizabad)	12 PM - 5 PM	Karachi Central	Dacoity	Car, Bike	24.91833	67.06817
499	25 years and above	Male	Johar Mor, Gulistan e johar	12 PM - 5 PM	Karachi East	Murder	Car	24.90466	67.11360

500 rows × 9 columns

Fig 4.6: Dropping Timestamp, Email address and Date of Incident from the dataset

4.2.4 Data Visualization (CountPlots, ScatterPlots and Histograms)

Data visualization can help in gaining better insights and patterns in data. It assists in understanding the phenomenon behind the generation of data. We have used countplots, scatterplots and histograms for data visualization.

```
# getting distribution of the input variable
plt.figure(figsize=(18, 5))
sns.countplot(x ='District', hue = 'Time of Incident', data = df)          #countplot
plt.title('DISTRICT vs TIME OF INCIDENT Data Distribition')
plt.show()
```

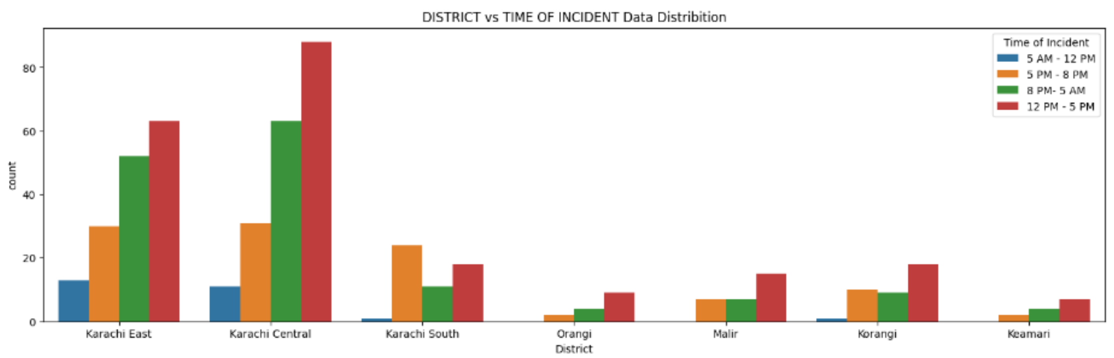


Fig 4.7: District VS Time Countplot

```python
#analysing data
df.hist(column='Latitude')    #histogram
plt.title('Histogram')
plt.xlabel('Latitude')
plt.ylabel('Latitude')
plt.show()
```

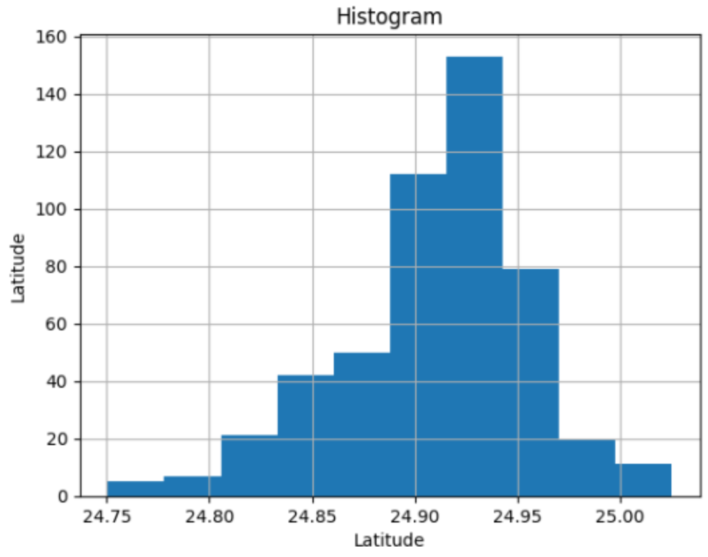
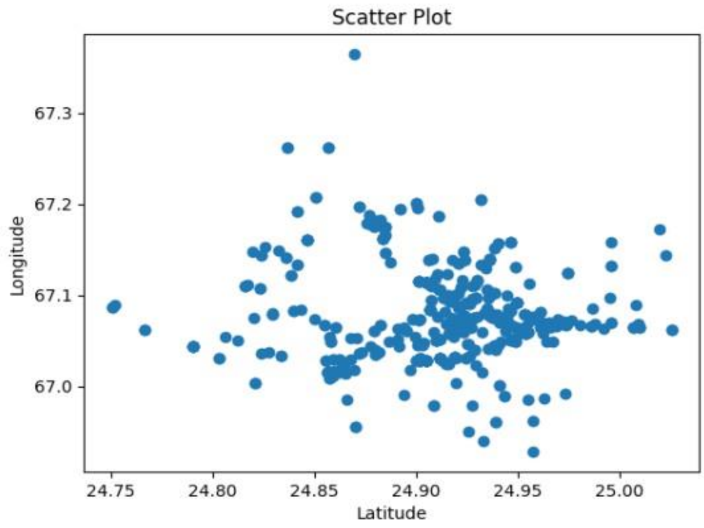


Fig 4.8: Latitude Histogram

```python
#analysing data
plt.scatter(df["Latitude"], df["Longitude"])     #scatterplot
plt.title('Scatter Plot')
plt.xlabel('Latitude')
plt.ylabel('Longitude')
plt.show()
```



==> Very low correlation can be observed between Latitude and Longitude variables

Fig 4.9: Scatterplot

Then, we have dropped the remaining irrelevant columns except for “Nearby location of incident” leaving us with 500 rows and 4 columns.

```
#dropping irrelevant columns from dataset
df.drop(["Age"], axis=1, inplace=True)
df.drop(["Gender"], axis=1, inplace=True)
df.drop(["District"], axis=1, inplace=True)
df.drop(["Crime"], axis=1, inplace=True)
df.drop(["Transportation mode used"], axis=1, inplace=True)
```

```
df=df.reset_index(drop=True)   #resetting index to default integer index
df   #changes incorporated permanently
```

	Nearby location of incident	Time of Incident	Latitude	Longitude
0	Johar Mor, Johar	5 AM - 12 PM	24.90466	67.11360
1	Dolmen Mall Hyderi	5 PM - 8 PM	24.93549	67.04049
2	Lasania service road near Bank (ATM), Gulshan-...	8 PM- 5 AM	24.90742	67.10990
3	Aziz bhatti park	8 PM- 5 AM	24.91315	67.09651
4	Near luckyone round about	12 PM - 5 PM	24.92603	67.09004
...	...	...	...	...
495	Sir Syed University	12 PM - 5 PM	24.91626	67.09312
496	Nazimabad near Imtiaz market	12 PM - 5 PM	24.91859	67.03210
497	FM Public school Buffer zone	12 PM - 5 PM	24.96151	67.06567
498	Karachi Academy school (Azizabad)	12 PM - 5 PM	24.91833	67.06817
499	Johar Mor, Gulistan e johar	12 PM - 5 PM	24.90466	67.11360

500 rows × 4 columns

Fig 4.10: Dropping remaining irrelevant columns except for Nearby location of incident

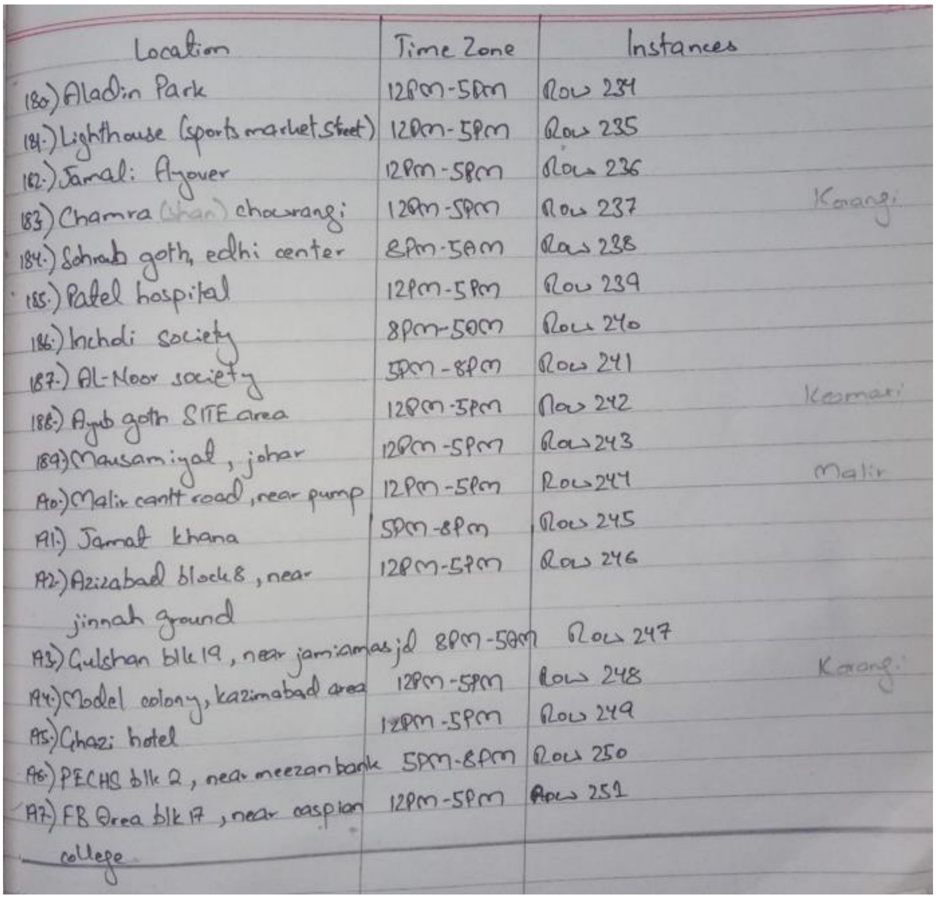
4.2.5 Adding and Evaluating the Target Variable "Crime Score"

Before adding and evaluating the target variable, we proceeded to find the number of unique locations in our dataset. Since there are different ways of writing a same location, we couldn’t rely on “Nearby location of incident” attribute to tell us about the number of unique locations. Instead, we would check the number of unique locations by counting the number of unique latitude or longitude coordinates in the dataset. A physical record has also been maintained in the copy for this purpose. The number of unique locations came up as 280.

```
1  #checking number of unique locations in dataset
2  df["Latitude"].value_counts()
```

```
Latitude
24.91768    18
24.90466    12
24.90512    12
24.91158     8
24.90207     6
            ..
24.95359     1
24.91793     1
24.96584     1
24.85948     1
24.91833     1
Name: count, Length: 280, dtype: int64
```

Fig 4.11: Count of number of unique locations in dataset



Location	Time Zone	Instances	
180) Aladin Park	12Pm-5Pm	Row 234	
181.) Lighthouse (sports market street)	12Pm-5Pm	Row 235	
182.) Jamali flyover	12Pm-5Pm	Row 236	
183) Chamra (khan) chowrangi	12Pm-5Pm	Row 237	Korangi
184.) Sohrab goth, edhi center	8Pm-5Am	Row 238	
185.) Patel hospital	12Pm-5Pm	Row 239	
186.) Inchdi society	8Pm-5Am	Row 240	
187.) Al-Noor society	5Pm-8Pm	Row 241	
188.) Ayub goth SITE area	12Pm-5Pm	Row 242	Keamari
189.) Mausamiyat, johar	12Pm-5Pm	Row 243	
190.) Malir cantt road, near pump	12Pm-5Pm	Row 244	Malir
191.) Jamat khana	5Pm-8Pm	Row 245	
192.) Azizabad block 8, near jinnah ground	12Pm-5Pm	Row 246	
193) Gulshan blk 19, near jamia masjid	8Pm-5Am	Row 247	
194.) Model colony, kazimabad area	12Pm-5Pm	Row 248	Korangi
195.) Ghazi hotel	12Pm-5Pm	Row 249	
196.) PECHS blk 2, near meezan bank	5Pm-8Pm	Row 250	
197) FB Area blk 17, near caspian college	12Pm-5Pm	Row 251	

Fig 4.12: Physical record maintained in copy

Let's now understand the meaning of our target variable. Crime Score represents the number of crimes that are committed during a period of time in a particular place. We have kept equal weightages for all crime types. Crime Score is calculated by counting the number of occurrences of the longitude & latitude of a location for a particular Time of Incident. Time of Incident is divided into four time zones as discussed before.

First, we added the Crime Score column to our dataframe and set all its values to zero. Then we used the GroupBy.cumcount() function to evaluate Crime Score. This function returns the cumulative count of occurrences within each group. The "groupby" operation in pandas is used to split a DataFrame into groups based on some criteria. It creates a GroupBy object that can be used to perform various operations on each group. The "cumcount" method is applied to a GroupBy object which in this case includes Time of Incident, Latitude and Longitude columns, and then computes the cumulative count of occurrences within each group. It starts from 0 and increments by 1 for each occurrence in the group.

Once Crime Score is evaluated, we will remove the duplicate rows keeping only the last instance of each duplicate row which will then give us the Crime Score for each unique location. We then crosscheck the returned dataframe with the physical record of data maintained in the copy to ensure validity. For this purpose, we had not yet removed the "Nearby location of incident" feature as it eased in ensuring that the returned dataframe had the correct Crime Score evaluated. Now, we can remove this feature and our dataset now consists of 500 rows and 4 columns.

4.2.6 Checking for categorical variables and encoding them

A categorical variable is a discrete variable that captures qualitative outcomes by placing observations into fixed groups/levels. These groups are mutually exclusive. Since machine learning algorithms only work with numerical values, we need to convert categorical variables to numeric values.

Categorical variables can be of three types namely binary, nominal and ordinal. Our dataset consists of only one categorical variable i.e "Time of Incident". We already know Time of Incident feature consists of 4 different outcomes, these outcomes can be considered ordinal and hence we have encoded them using LabelEncoder(), each unique category will be assigned an integer value starting from one.

Time “12 PM – 5 PM” is assigned a value of 0. Time “5 AM – 12 PM” is assigned a value of 1. Time “5 PM – 8 PM” is assigned a value of 2. Time “8 PM – 5 AM” is assigned a value of 3.

```
#checking number of instances of each unique value in Time of Incident feature after encoding
df["Time of Incident"].value_counts()

Time of Incident
0    144
3     82
2     79
1     18
Name: count, dtype: int64
```

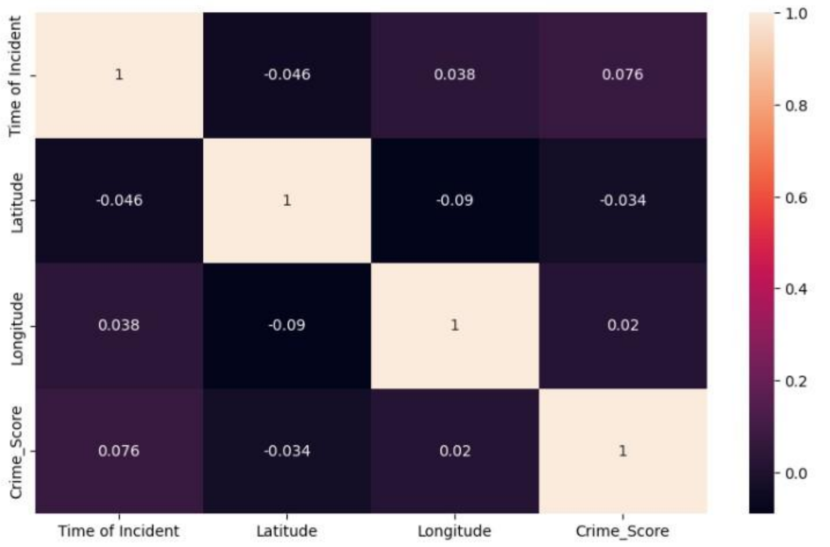
Fig 4.13: Encoded “Time of Incident” feature

4.2.7 Data Visualization (Checking for multicollinearity)

Correlation is a statistical measure that expresses the extent to which two variables are linearly related. High correlation may cause multicollinearity which can lead to unreliable estimates. We have run correlation matrices using .corr() method and represented it using a heatmap to observe the correlation among columns. It is observed none of the features are strong positively/negatively correlated with any other feature or target variable, hence there isn't need of removing any feature.

```python
#Correlation Coefficient - Pearson's Correlation Coefficient
corr = df.corr()

#Plotting Heatmap
plt.figure(figsize = (10,6))
sns.heatmap(corr, annot=True)
plt.show()
```



As observed, none of the features are strongly positively/negatively correlated, hence there isn't any need of removing columns.

Fig 4.14: Correlation Heatmap

Some preprocessing steps such as 'Checking and Removing Outliers" and "Feature Scaling" have not been implemented. Outliers can only be checked for numerical data, our dataset contains only two numerical (float) features namely Latitude and Longitude. However, we had decided not to check these features for outliers as their data was manually prepared by us. Feature Scaling was implemented at first, but it had little effect in improving the performance of Machine learning algorithms therefore it was not applied at the end.

4.3 Models and Machine Learning Algorithms chosen for implementation

We have implemented the following algorithms:

- Random Forest (non-parametric learning algorithm
- CatBoost Classifier (non-parametric learning algorithm)
- K Nearest Neighbors (KNN) (non-parametric learning algorithm)
- Logistic Regression (parametric learning algorithm)
- Multi Layer Perceptron (MLP) (neural network architecture)

A total of 5 algorithms have been implemented. Input to the model are the features "Time of Incident", "Latitude" and "Longitude". Output feature will be "Crime_Score". Our target variable had 6 different outcomes namely '1', '2', '3', '4', '8' and '11'. It was observed that our dataset was imbalanced. Feeding imbalanced data to your classifier can make it biased in favor of the majority class, simply because it did not have enough data to learn about the minority.

```
#checking number of instances of each unique value in Target variable
df["Crime_Score"].value_counts()
```

```
Crime_Score
1     200
2     101
3      12
4       6
11      2
8       2
Name: count, dtype: int64
```

Fig 4.15: Distribution of Target Variable

To resolve this issue, we decided to use a "SMOTE (Synthetic Minority Oversampling Technique)". It is a data augmentation technique. SMOTE generates synthetic samples from the minority class. This algorithm helps to overcome the overfitting problem posed by random oversampling. It focuses on the feature space to generate new instances with the help of interpolation between the positive instances that lie together. SMOTE will oversample the examples in the minority class.

In our case, the minority classes are ‘2’,’3’, ‘4’, ‘8’ and ‘11’. The minority class will have the same number of samples as the majority class which is ‘1’ in our case.

Each minority class will therefore have 200 samples.

```
#applying SMOTE oversampling technique
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state=42, k_neighbors=1)
X_res, y_res = sm.fit_resample(inp, out)
```

```
# checking number of instances of each unique value in Target variable after oversampling
y_res.value_counts()
```

```
Crime_Score
1            200
2            200
3            200
4            200
8            200
11           200
Name: count, dtype: int64
```

Fig 4.16: Distribution of Target Variable after applying SMOTE

A single train test split procedure is used for all models. 80% of data is assigned to train set and 20% to test set.

```
#Splitting data into train and test set
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X_res, y_res, test_size = 0.2, random_state=0)
print('Training set shape: ', x_train.shape, y_train.shape)
print('Testing set shape: ', x_test.shape, y_test.shape)
```

```
Training set shape:  (960, 3) (960, 1)
Testing set shape:  (240, 3) (240, 1)
```

Fig 4.17: Train Test split

Let’s discuss the algorithms implemented briefly one by one:-

Random Forest is an ensemble learning method used for both classification and regression tasks. It is based on the concept of decision trees, but instead of using a single decision tree, it will combine multiple decision trees to improve accuracy and reduce overfitting. Each tree in the forest is built on a random subset of the data and features, and the final prediction is determined by aggregating the predictions of individual trees. We have implemented four models and tuned the n_estimators, max_depth and min_samples_split hyperparameters for each model of our classifier.

```python
#Training and Testing Model 2
#Importing RandomForest Classifier
from sklearn.ensemble import RandomForestClassifier
#Creating object and fitting data onto the model
rf_2=RandomForestClassifier(n_estimators=150, max_depth=80, min_samples_split=4).fit(x_train,y_train)
y_tr=rf_2.predict(x_train)
y_pred = rf_2.predict(x_test)
acc=metrics.accuracy_score(y_test, y_pred)
wprec=metrics.precision_score(y_test, y_pred, average='weighted')
wrecall=metrics.recall_score(y_test, y_pred, average='weighted')
wf1=metrics.f1_score(y_test, y_pred, average='weighted')
print("Train Accuracy is "  + str (metrics.accuracy_score(y_train, y_tr)*100) + "%")
print("Test Accuracy is "  + str (acc*100) + "%")
print("Weighted Precision Score is "  + str (wprec*100) + "%")
print("Weighted Recall Score is "  + str (wrecall*100) + "%")
print("Weighted F1 Score is "  + str (wf1*100) + "%")
print("\n")
print("Confusion Matrix "+"\n",confusion_matrix(y_test, y_pred))
print("\n")
# target_names = ["Safe", "Unsafe"]
print("Classification Report"+"\n",classification_report(y_test, y_pred))
weighted_precisions.append(["Random Forest Model 2", wprec, acc, wrecall, wf1])   #appending models score
```

```
Train Accuracy is 99.58333333333333%
Test Accuracy is 87.91666666666667%
Weighted Precision Score is 87.94819078947368%
Weighted Recall Score is 87.91666666666667%
Weighted F1 Score is 87.89250714577379%


Confusion Matrix
 [[23  7  3  1  0  0]
 [ 9 22  0  1  0  0]
 [ 0  3 35  3  0  0]
 [ 2  0  0 43  0  0]
 [ 0  0  0  0 41  0]
 [ 0  0  0  0  0 47]]


Classification Report
              precision    recall  f1-score   support

           1       0.68      0.68      0.68        34
           2       0.69      0.69      0.69        32
           3       0.92      0.85      0.89        41
           4       0.90      0.96      0.92        45
           8       1.00      1.00      1.00        41
          11       1.00      1.00      1.00        47

    accuracy                           0.88       240
   macro avg       0.86      0.86      0.86       240
weighted avg       0.88      0.88      0.88       240
```

Fig 4.18: Random Forest Model 2

CatBoost is a powerful gradient boosting library that is specifically designed for handling categorical features in machine learning tasks. It is known for its high accuracy, robustness, and efficient handling of categorical data. We have used the CatBoost classifier for our multiclass classification problem. We have implemented four models and tuned the iterations and learning_rate hyperparameters for each model.

```
#Training and Testing Model 4
#Importing CatBoost Classifier
from catboost import CatBoostClassifier
#Creating object and fitting data onto the model
cat_4 = CatBoostClassifier(iterations=750, learning_rate=0.3).fit(x_train, y_train)
y_tr=cat_4.predict(x_train)
y_pred = cat_4.predict(x_test)
acc=metrics.accuracy_score(y_test, y_pred)
wprec=metrics.precision_score(y_test, y_pred, average='weighted')
wrecall=metrics.recall_score(y_test, y_pred, average='weighted')
wf1=metrics.f1_score(y_test, y_pred, average='weighted')
print("\n")
print("Train Accuracy is "  + str (metrics.accuracy_score(y_train, y_tr)*100) + "%")
print("Test Accuracy is "  + str (acc*100) + "%")
print("Weighted Precision Score is "  + str (wprec*100) + "%")
print("Weighted Recall Score is "  + str (wrecall*100) + "%")
print("Weighted F1 Score is "  + str (wf1*100) + "%")
print("\n")
print("Confusion Matrix "+"\n",confusion_matrix(y_test, y_pred))
print("\n")
print("Classification Report"+"\n",classification_report(y_test, y_pred))
weighted_precisions.append(["CatBoost Model 4", wprec, acc, wrecall, wf1])  #appending models score
```

```
Train Accuracy is 99.6875%
Test Accuracy is 87.91666666666667%
Weighted Precision Score is 87.73409277504105%
Weighted Recall Score is 87.91666666666667%
Weighted F1 Score is 87.78140117907436%


Confusion Matrix
 [[23  7  3  1  0  0]
 [11 20  1  0  0  0]
 [ 1  1 36  3  0  0]
 [ 0  1  0 44  0  0]
 [ 0  0  0  0 41  0]
 [ 0  0  0  0  0 47]]


 Classification Report
               precision    recall  f1-score   support

           1       0.66      0.68      0.67        34
           2       0.69      0.62      0.66        32
           3       0.90      0.88      0.89        41
           4       0.92      0.98      0.95        45
           8       1.00      1.00      1.00        41
          11       1.00      1.00      1.00        47

    accuracy                           0.88       240
   macro avg       0.86      0.86      0.86       240
weighted avg       0.88      0.88      0.88       240
```

Fig 4.19: CatBoost Model 4

The **K-Nearest Neighbors classifier (k-NN)** is one of the most commonly used classifiers in supervised machine learning. An observation is predicted to be the class of that of the largest proportion of the k nearest observations. To make a prediction for a new data point, the algorithm finds the closest data points in the training dataset. When a new piece of data is given without a label, that new piece of data is compared to every piece of existing data. Then, the most similar k pieces of data (the nearest neighbors) are taken and their labels are focused. Lastly, a majority vote is taken from the k most similar pieces of data, and the majority is the new class assigned to the data which were asked to classify. We have tuned the n_neighbors, metric and weights hyperparameters for each model.

```
#Training and Testing Model 1
#Importing KNeighbors Classifier
from sklearn.neighbors import KNeighborsClassifier
#Creating object and fitting data onto the model
knn_1=KNeighborsClassifier(n_neighbors=3,metric='minkowski',weights='uniform').fit(x_train,y_train);
y_tr=knn_1.predict(x_train)
y_pred=knn_1.predict(x_test)
# print(y_pred)
print("Train Accuracy is "  + str (metrics.accuracy_score(y_train, y_tr)*100) + "%")
acc=metrics.accuracy_score(y_test, y_pred)
wprec=metrics.precision_score(y_test, y_pred, average='weighted')
wrecall=metrics.recall_score(y_test, y_pred, average='weighted')
wf1=metrics.f1_score(y_test, y_pred, average='weighted')
print("Test Accuracy is "  + str (acc*100) + "%")
print("Weighted Precision Score is "  + str (wprec*100) + "%")
print("Weighted Recall Score is "  + str (wrecall*100) + "%")
print("Weighted F1 Score is "  + str (wf1*100) + "%")
print("\n")
print("Confusion Matrix "+"\n",confusion_matrix(y_test, y_pred))
print("\n")
print("Classification Report"+"\n",classification_report(y_test, y_pred))
# weighted_precisions.append(["KNeighbors Classifier Model 1", wprec, acc, wrecall, wf1])   #appending models score
```

```
Train Accuracy is 90.72916666666667%
Test Accuracy is 86.25%
Weighted Precision Score is 85.51856863477703%
Weighted Recall Score is 86.25%
Weighted F1 Score is 85.78063952614706%


Confusion Matrix
 [[18  8  4  3  1  0]
 [ 8 22  0  1  1  0]
 [ 1  2 37  1  0  0]
 [ 1  0  0 42  0  2]
 [ 0  0  0  0 41  0]
 [ 0  0  0  0  0 47]]


Classification Report
               precision    recall  f1-score   support

           1       0.64      0.53      0.58        34
           2       0.69      0.69      0.69        32
           3       0.90      0.90      0.90        41
           4       0.89      0.93      0.91        45
           8       0.95      1.00      0.98        41
          11       0.96      1.00      0.98        47

    accuracy                           0.86       240
   macro avg       0.84      0.84      0.84       240
weighted avg       0.86      0.86      0.86       240
```

Fig 4.20: KNN Model 1

Logistic regression (also called logit regression) is commonly used to estimate the probability that an instance belongs to a particular class. Logistic regression is a widely used supervised classification technique. Logistic regression is used for predicting the probability that an observation is of a certain class. Since we are dealing with a multi classification problem, we will create our models with our "multi_class" parameter set to 'ovr' or 'multinomial' indicating which strategy to use for multi classification. We have tuned the solver and multi_class hyperparameters for each of our model.

```python
#Training and Testing Model 1
#Importing Logistic Regression Classifier
from sklearn.linear_model import LogisticRegression
#Creating object and fitting data onto the model
logreg_1 = LogisticRegression(solver='sag', multi_class='ovr').fit(x_train , y_train)
y_tr=logreg_1.predict(x_train)
y_pred=logreg_1.predict(x_test)
print("Train Accuracy is "  + str (metrics.accuracy_score(y_train, y_tr)*100) + "%")
acc=metrics.accuracy_score(y_test, y_pred)
wprec=metrics.precision_score(y_test, y_pred, average='weighted')
wrecall=metrics.recall_score(y_test, y_pred, average='weighted')
wf1=metrics.f1_score(y_test, y_pred, average='weighted')
print("Test Accuracy is "  + str (acc*100) + "%")
print("Weighted Precision Score is "  + str (wprec*100) + "%")
print("Weighted Recall Score is "  + str (wrecall*100) + "%")
print("Weighted F1 Score is "  + str (wf1*100) + "%")
print("\n")
print("Confusion Matrix "+"\n",confusion_matrix(y_test, y_pred))
print("\n")
print("Classification Report"+"\n",classification_report(y_test, y_pred))
# weighted_precisions.append(["Logistic Regression Model 1", wprec, acc, wrecall, wf1])   #appending models score
```

```
Train Accuracy is 34.6875%
Test Accuracy is 37.916666666666664%
Weighted Precision Score is 18.203712285037586%
Weighted Recall Score is 37.916666666666664%
Weighted F1 Score is 24.02898944786412%


Confusion Matrix
 [[ 0  4  9  0  9 12]
 [ 0  0  8  0 17  7]
 [ 0  0  3  0 16 22]
 [ 0  9  7  0  0 29]
 [ 0  0  0  0 41  0]
 [ 0  0  0  0  0 47]]


Classification Report
              precision    recall  f1-score   support

           1       0.00      0.00      0.00        34
           2       0.00      0.00      0.00        32
           3       0.11      0.07      0.09        41
           4       0.00      0.00      0.00        45
           8       0.49      1.00      0.66        41
          11       0.40      1.00      0.57        47

    accuracy                           0.38       240
   macro avg       0.17      0.35      0.22       240
weighted avg       0.18      0.38      0.24       240
```

Fig 4.21: Logistic Regression Model 1

Multi Layer Perceptron (MLP), also called feed forward networks are artificial neural networks. Neural networks can be visualized as a series of connected layers that form a network, they connect an observation's feature values at one end, and the target value at the other end. The name feedforward comes from the fact that an observation's feature values are fed "forward" through the network, with each layer successively transforming the feature values with the goal that the output at the end is the same as the target's value. Feedforward neural networks contain three types of layers of units. At the start of the neural network is an input layer where each unit contains an observation's value for a single feature. At the end of the neural network is the output layer, which transforms the output of the hidden layers into values useful for the task in hand. Between the input and output layers are the hidden layers. These hidden layers successively transform the feature values from the input layer to something that, once processed by the output layer, resembles the target class. We have tuned hidden_layer_size, activation, solver, learning_rate and early_stopping for each model.

```python
#Training and Testing Model 1
# Importing MLPClassifer
from sklearn.neural_network import MLPClassifier
# Create model object and fitting data onto the model
mlp_1 = MLPClassifier(hidden_layer_sizes=(80,80), activation='relu', solver='sgd',
                      learning_rate='adaptive', early_stopping=True).fit(x_train,y_train)
y_tr=mlp_1.predict(x_train)
y_pred=mlp_1.predict(x_test)
print("Train Accuracy is "  + str (metrics.accuracy_score(y_train, y_tr)*100) + "%")
acc=metrics.accuracy_score(y_test, y_pred)
wprec=metrics.precision_score(y_test, y_pred, average='weighted')
wrecall=metrics.recall_score(y_test, y_pred, average='weighted')
wf1=metrics.f1_score(y_test, y_pred, average='weighted')
print("Test Accuracy is "  + str (acc*100) + "%")
print("Weighted Precision Score is "  + str (wprec*100) + "%")
print("Weighted Recall Score is "  + str (wrecall*100) + "%")
print("Weighted F1 Score is "  + str (wf1*100) + "%")
print("\n")
print("Confusion Matrix "+"\n",confusion_matrix(y_test, y_pred))
print("\n")
print("Classification Report"+"\n",classification_report(y_test, y_pred))
# weighted_precisions.append(["MultiLayer Perceptron Model 1", wprec, acc, wrecall, wf1])
```

```
Train Accuracy is 36.041666666666664%
Test Accuracy is 39.58333333333333%
Weighted Precision Score is 21.166675248000548%
Weighted Recall Score is 39.58333333333333%
Weighted F1 Score is 26.167469840020978%


Confusion Matrix
 [[ 0  4  0  9  9 12]
 [ 0  0  0  8 17  7]
 [ 0  0  0  3 16 22]
 [ 0  9  0  7  0 29]
 [ 0  0  0  0 41  0]
 [ 0  0  0  0  0 47]]


Classification Report
               precision    recall  f1-score   support

           1       0.00      0.00      0.00        34
           2       0.00      0.00      0.00        32
           3       0.00      0.00      0.00        41
           4       0.26      0.16      0.19        45
           8       0.49      1.00      0.66        41
          11       0.40      1.00      0.57        47

    accuracy                           0.40       240
   macro avg       0.19      0.36      0.24       240
weighted avg       0.21      0.40      0.26       240
```

Fig 4.22: MLP Model 1

4.4 Tabular and Graphical Comparison Of Models

Graphical comparison is done by making a bar plot. 8 models are compared, 4 each of Random Forest and 4 of CatBoost. Graphical comparison is made on the basis of “test accuracy score”.

```python
print('-'*126)
text = "|{:<34}|{:^14}|{:^24}|{:^24}|{:^24}|"
print(text.format("                 Models","Accuracy", "Weighted Precision", "Weighted Recall", "Weighted F1-score"))
print('-'*126)

for i in range(len(weighted_precisions)):
    print(text.format("  "+weighted_precisions[i][0], round(float(weighted_precisions[i][2]),3), round(float(weighted_precis

print('-'*126)
```

Models	Accuracy	Weighted Precision	Weighted Recall	Weighted F1-score
Random Forest Model 1	0.867	0.869	0.867	0.865
Random Forest Model 2	0.879	0.879	0.879	0.879
Random Forest Model 3	0.875	0.875	0.875	0.874
Random Forest Model 4	0.875	0.876	0.875	0.875
CatBoost Model 1	0.846	0.836	0.846	0.838
CatBoost Model 2	0.879	0.874	0.879	0.875
CatBoost Model 3	0.858	0.85	0.858	0.853
CatBoost Model 4	0.879	0.877	0.879	0.878

Fig 4.23: Tabular Comparison of Models

```python
import matplotlib.pyplot as plt
accuracy=[]
models_names=["RF 1", "RF 2", "RF 3", "RF 4", "CB 1",  "CB 2",  "CB 3",  "CB 4"]
for i in weighted_precisions:
    accuracy.append(float(i[2]))

plt.figure(figsize=(16, 6))
plt.bar(models_names, accuracy)
plt.xlabel('Model Names')
plt.ylabel('Test Accuracy')
plt.title('Comparison of Model Accuraccies')
for i, score in enumerate(accuracy):
    plt.text(i, score, str(round(float(score), 3)), ha='center', va='bottom')
plt.show()
plt.show()
```

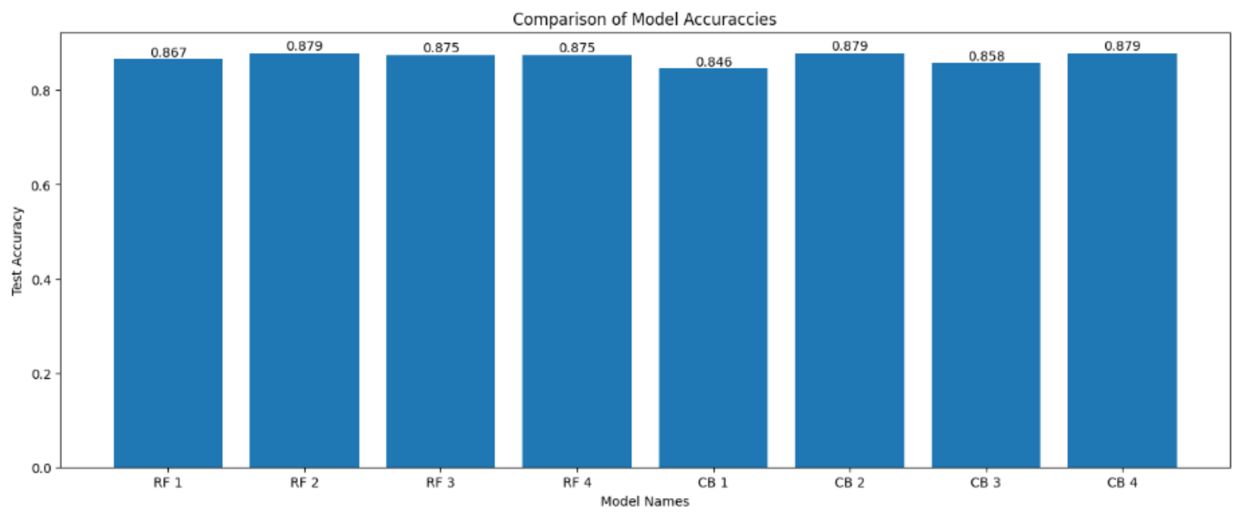


Fig 4.24: Graphical Comparison of Models

It can be observed that Random Forest 2, CatBoost Model 2 and CatBoost Model 4 have the best test accuracy scores (0.879) among the 8 models compared. However, when taking train accuracy into account, CatBoost Model 4 has a train accuracy of (0.996) as compared to CatBoost Model 2 (0.969) and Random Forest 2 (0.995). Therefore, CatBoost Model 4 is our best model.

4.5 Summary

In this chapter, we discussed the usage of ML to predict crime score for each location wrt time. We have performed 7 data pre-processing steps to ensure the data is free from impurities. First, we removed duplicate rows, null values, and irrelevant attributes. Then, we performed data visualization using countplots, histograms and scatterplots to gain insights in data. Our target variable “Crime_Score” was then added and evaluated. A heatmap for observing correlation was also plotted. Lastly, we encoded categorical variables.

It was observed that our dataset was imbalanced. To resolve this, we used SMOTE (Synthetic Minority OverSampling Technique). The data was then split into input and output, and then into train and test set. Input variables were “Time of Incident”, “Latitude” and Longitude” while the output variable was “Crime_Score”. We implemented 5 different algorithms namely Random Forest, CatBoost, KNeighbors, Logistic Regression and Multi- Layer Perceptron. Random Forest and CatBoost models were outperforming KNeighbors, Logistic Regression and MultiLayer Perceptron models. Tabular and Graphical comparison was made for the Random Forest and CatBoost Models which showed Catboost Model 4 to be the best among all.