# DS8003 – FINAL REPORT (GROUP 7)

Analysis of Crime in Chicago

Syed Saad Hussain
saad2.hussain@ryerson.ca
Pubali Das Chowdhury
pdaschowdhury@ryerson.ca

## Contents

## Problem Definition

In today's world, there is a need to identify patterns in criminal activity to better inform government policy-making and policing strategies. Furthermore, we often hear from the media that there is a connection between crime, socioeconomic indicators, and population distribution. If government planners can gain a better understanding of the underlying factors that influence levels and types of criminal behaviour, it would greatly assist in key decision-making related to jobs, education, community housing, and transportation. Having access to this type of analysis may lead to a safer and more equitable society.

For our project, we utilized the tools taught in the course to find key insights relating to criminal activity in Chicago and aligned it with socioeconomic factors and population statistics to discover key insights about the interconnectedness of these three elements.

## Data Description

For our project, we drew insights from the following three datasets:

1. Chicago Crimes (2012-2017) – available here: https://www.kaggle.com/currie32/crimes-in-chicago

2. Socioeconomic indicators – available here: https://data.cityofchicago.org/Health-Human-Services/Per-Capita-Income/r6ad-wvtk

3. Population by Ward – estimated by an independent researcher using census statistics, available here: https://docs.google.com/spreadsheets/d/1sxM-JajdrC7R1VZ_sHjUwkTQ0qs2z7a7jFbCblTii3Q/edit#gid=1503084939

### Attribute Descriptions

Here are the attribute descriptions of the Chicago Crimes dataset:

| Attributes | Description |
|---|---|
| ID | Unique identifier for the record. |
| Case Number | The Chicago Police Department RD Number (Records Division Number), which is unique to the incident. |
| Date | Date when the incident occurred. this is sometimes a best estimate. |
| Block | The partially redacted address where the incident occurred, placing it on the same block as the actual address. |
| IUCR | The Illinois Uniform Crime Reporting code. This is directly linked to the Primary Type and Description. |
| Primary Type | The primary description of the IUCR code. |
| Description | The secondary description of the IUCR code, a subcategory of the primary description. |
| Location Description | Description of the location where the incident occurred. |
| Arrest | Indicates whether an arrest was made. |
| Domestic | Indicates whether the incident was domestic-related as defined by the Illinois Domestic Violence Act. |
| Beat | Indicates the beat where the incident occurred. A beat is the smallest police geographic area – each beat has a dedicated police beat car. Three to five beats make up a police sector, and three sectors make up a police district. The Chicago Police Department has 22 police districts. |
| District | Indicates the police district where the incident occurred. |
| Ward | The ward (City Council district) where the incident occurred |
| Community Area | Indicates the community area where the incident occurred. Chicago has 77 community areas. |
| X Coordinate | The x coordinate of the location where the incident occurred in State Plane Illinois East NAD 1983 projection. This location is shifted from the actual location for partial redaction but falls on the same block. |
| Y Coordinate | The y coordinate of the location where the incident occurred in State Plane Illinois East NAD 1983 projection. This location is shifted from the actual location for partial redaction but falls on the same block. |
| Year | Year the incident occurred. |
| Updated On | Date and time the record was last updated. |
| Latitude | The latitude of the location where the incident occurred. This location is shifted from the actual location for partial redaction but falls on the same block. |

| | |
|---|---|
| Longitude | The longitude of the location where the incident occurred. This location is shifted from the actual location for partial redaction but falls on the same block. |
| Location | The location where the incident occurred in a format that allows for creation of maps and other geographic operations on this data portal. This location is shifted from the actual location for partial redaction but falls on the same block. |

Here are the attribute descriptions that were used from the socioeconomic indicator dataset:

| **Attributes** | **Description** |
|---|---|
| Community Area Number | Indicates the community area where the incident occurred. Chicago has 77 community areas. |
| Community Area Name | Name of the community area |
| Poverty Rate | Percentage of households below the poverty line. |
| Unemployment Rate | Percentage of constituents over 16 that are unemployment. |
| Educational Attainment | Percentage of adults over the age of 25 without a high school diploma |

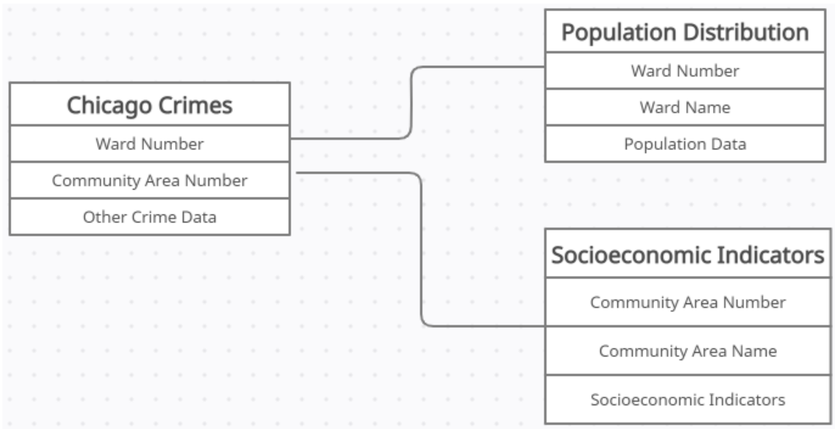
Here are the attribute descriptions that were used from the population distribution dataset:

| **Attributes** | **Description** |
|---|---|
| Ward Number | The ward (City Council district) where the incident occurred |
| Ward Name | Name of the ward |
| Total Population | Total number of residents in the ward |
| White Population | Number of white residents in the ward |
| Black Population | Number of black residents in the ward |
| Asian Population | Number of Asian residents in the ward |
| Hispanic Population | Number of Hispanic residents in the ward |
| Other Population | Number of residents who have not self-identified in the ward |

The socioeconomic factors and population distribution datasets were primarily used as lookup tables to check for correlation with criminal activity.

Here is a simplified relational schema that illustrates how the tables were joined for the queries that required it.

We joined the ‘Chicago Crimes & Population’ Distribution data sets on ward number. We joined the ‘Chicago Crimes’ and ‘Socioeconomic Indicators’ data sets on community area number.




## Data Statistics

Here is a sample of the data statistics for the population distribution dataset:

```
>>> df_population.select("White2016").summary("count","mean","stddev", "min", "max").show()
+-------+-----------------+
|summary|        White2016|
+-------+-----------------+
|  count|               50|
|   mean|            510.0|
| stddev|174.4327950816589|
|    min|            1,114|
|    max|            9,436|
+-------+-----------------+
```

```
>>> df_population.select("Black2016").summary("count","mean","stddev", "min", "max").show()
+-------+-----------------+
|summary|        Black2016|
+-------+-----------------+
|  count|               50|
|   mean|           627.25|
| stddev|62.56396726551155|
|    min|            1,072|
|    max|            9,845|
+-------+-----------------+
```

```
>>> df_population.select("Latino2016").summary("count","mean","stddev", "min", "max").show(
+-------+----------------+
|summary|      Latino2016|
+-------+----------------+
|  count|              50|
|   mean|          598.75|
| stddev|82.2734262484957|
|    min|          10,496|
|    max|           9,699|
+-------+----------------+
```

Here are the statistics for the socioeconomic indicators dataset. We only provided one sample statistic because they are all very similar.

```
>>> from pyspark.sql import SQLContext
>>> sqlContext = SQLContext(sc)
>>> df_income = sqlContext.read.format('com.databricks.spark.csv').options(header='true', inferschema='true').load('/user/root/Per_Capita_Income.csv')
>>> df_income.printSchema()
root
 |-- Community Area Number: integer (nullable = true)
 |-- COMMUNITY AREA NAME: string (nullable = true)
 |-- PERCENT OF HOUSING CROWDED: double (nullable = true)
 |-- PERCENT HOUSEHOLDS BELOW POVERTY: double (nullable = true)
 |-- PERCENT AGED 16+ UNEMPLOYED: double (nullable = true)
 |-- PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA: double (nullable = true)
 |-- PERCENT AGED UNDER 18 OR OVER 64: double (nullable = true)
 |-- PER CAPITA INCOME : integer (nullable = true)
 |-- HARDSHIP INDEX: integer (nullable = true)


>>> df_income.select("PERCENT HOUSEHOLDS BELOW POVERTY").summary("count","mean", "min", "max").show()
+-------+--------------------------------+
|summary|PERCENT HOUSEHOLDS BELOW POVERTY|
+-------+--------------------------------+
|  count|                              78|
|   mean|               21.739743589743597|
|    min|                             3.3|
|    max|                            56.5|
+-------+--------------------------------+
```

Below is the schema for the Chicago Crimes dataset. Since there were no aggregable integer fields, it was not possible to determine the data statistics. All integer fields in this data set were reference IDs.

```
>>> from pyspark.sql import SQLContext
>>> sqlContext = SQLContext(sc)
>>> df = sqlContext.read.format('com.databricks.spark.csv').options(header='true', inferschema='true').load('/user/root/Chicago_Crimes_2012_to_2017.csv')
```

```
>>> df.printSchema()
root
 |-- _c0: integer (nullable = true)
 |-- ID: integer (nullable = true)
 |-- Case Number: string (nullable = true)
 |-- Date: string (nullable = true)
 |-- Block: string (nullable = true)
 |-- IUCR: string (nullable = true)
 |-- Primary Type: string (nullable = true)
 |-- Description: string (nullable = true)
 |-- Location Description: string (nullable = true)
 |-- Arrest: boolean (nullable = true)
 |-- Domestic: boolean (nullable = true)
 |-- Beat: integer (nullable = true)
 |-- District: double (nullable = true)
 |-- Ward: double (nullable = true)
 |-- Community Area: double (nullable = true)
 |-- FBI Code: string (nullable = true)
 |-- X Coordinate: double (nullable = true)
 |-- Y Coordinate: double (nullable = true)
 |-- Year: integer (nullable = true)
 |-- Updated On: string (nullable = true)
 |-- Latitude: double (nullable = true)
 |-- Longitude: double (nullable = true)
 |-- Location: string (nullable = true)
```

## Work Distribution

As our third group member dropped the course two weeks before the project deadline, the work was redistributed as follows. Please note that we also had to readjust our own expectations on what was achievable. Thus, we reverted to using Tableau instead of Elasticsearch for our data visualizations which was what we originally planned.

Syed Saad Hussain

- Leveraged HiveQL queries to draw insights that required joining different tables
- Created Tableau charts for visualizing correlations between poverty, socioeconomic factors, and population statistics

Pubali Das Chowdhury:

- Leveraged Spark SQL and Spark RDDs to preprocess the datasets and discover insights specific to the Crime in Chicago dataset
- Created Tableau charts for creating visualizations specific to the Crimes in Chicago dataset

## Solution Description

The following section presents a brief account of the tools used in the project and the rationale behind employing them.

### Tool 1 – Hadoop HDFS & Filezilla

We used FileZilla to upload files to the distributed file system, HDFS, so that we could employ analytics tools to find our insights.

Here are some screencaps of us loading data into HDFS:

```
[root@sandbox-hdp lab]# cd /root/project
```

```
[root@sandbox-hdp project]# ls
Chicago_Crimes_2012_to_2017.csv  chicago_income.csv  socioeconomic_indicators.csv
```

```
[root@sandbox-hdp project]# hadoop fs -mkdir /user/root/project
```

```
[root@sandbox-hdp project]# hadoop fs -put /root/project/Chicago_Crimes_2012_to_2017.csv /user/root/project
[root@sandbox-hdp project]# hadoop fs -put /root/project/socioeconomic_indicators.csv /user/root/project
[root@sandbox-hdp project]# hadoop fs -put /root/project/chicago_income.csv /user/root/project
```

```
[root@sandbox-hdp project]# hadoop fs -ls /user/root/project
Found 3 items
-rw-r--r--   1 root root  366804568 2021-11-22 22:21 /user/root/project/Chicago_Crimes_2012_to_2017.csv
-rw-r--r--   1 root root       3865 2021-11-22 22:21 /user/root/project/chicago_income.csv
-rw-r--r--   1 root root       9064 2021-11-22 22:21 /user/root/project/socioeconomic_indicators.csv
```

## Tool 2 –Spark RDD

We used spark to preprocess our dataset and perform some low-level transformations to inform our preliminary analysis. We found that pySpark was very robust and allowed us to leverage python commands and libraries such as pandas.

For example, here is a snippet of us using Spark RDDs to group community wards to aggregate the number of crimes that occurred in Chicago between 2012 and 2017.

```
>>> textFile = sc.textFile("/user/root/Chicago_Crimes_2012_to_2017.csv")
>>> text = textFile.map(lambda line: line.split(",")).map(lambda fields: (fields[14]))
>>> count = text.flatMap(lambda line: line.split()).map(lambda location: (location,1) ).reduceByKey(lambda v1,v2: v1+v2)
>>> swap_kv = count.map(lambda x: (x[1], x[0]))
>>> arranging_descending_order = swap_kv.sortByKey(False)
>>> final_rdd = arranging_descending_order.map(lambda x: (x[1], x[0]))
>>> final_rdd.take(10)
[(u'25.0', 95002), (u'8.0', 53102), (u'43.0', 48114), (u'23.0', 45977), (u'29.0', 45020), (u'28.0', 42288), (u'71.0', 40603), (u'67.0', 40495), (u'24.0', 40368), (u'32.
0', 39028)]
>>> final_rdd.take(20)
[(u'25.0', 95002), (u'8.0', 53102), (u'43.0', 48114), (u'23.0', 45977), (u'29.0', 45020), (u'28.0', 42288), (u'71.0', 40603), (u'67.0', 40495), (u'24.0', 40368), (u'32.
0', 39028), (u'49.0', 36858), (u'68.0', 36689), (u'69.0', 35888), (u'66.0', 34014), (u'6.0', 33535), (u'44.0', 32509), (u'26.0', 31077), (u'22.0', 29954), (u'27.0', 277
22), (u'19.0', 27348)]
>>> 
```

## Tool 3 –Spark SQL

We found Spark SQL to be a powerful tool to query the type of crimes and locations of the occurrence of crime in the above found communities. We used top five community areas where the count of crime is the highest and wanted to check if there are specific types of crime which are common in those community areas.

Creating a data frame to use SQL queries in spark.

```
>>> from pyspark.sql import SQLContext
>>> sqlContext = SQLContext(sc)
>>> df = sqlContext.read.format('com.databricks.spark.csv').options(header='true', inferschema='true').load('/user/root/Chicago_Crimes_2012_to_2017.csv')
```

- Crime type and Location for Community Area 25:

```
>>> df.where (F.col("Community Area") == "25").groupBy("Location Description", "Primary Type").agg(F.count("Primary Type").alias("count")).orderBy(F.col("count").desc()
).show()
+--------------------+------------------+-----+
|Location Description|      Primary Type|count|
+--------------------+------------------+-----+
|            SIDEWALK|         NARCOTICS| 8088|
|           APARTMENT|           BATTERY| 5681|
|              STREET|         NARCOTICS| 4900|
|            SIDEWALK|           BATTERY| 3856|
|              STREET|             THEFT| 3171|
|              STREET|   CRIMINAL DAMAGE| 3051|
|           RESIDENCE|           BATTERY| 2953|
|              STREET|           BATTERY| 2726|
|              STREET|MOTOR VEHICLE THEFT| 2679|
|           RESIDENCE|     OTHER OFFENSE| 2087|
|           RESIDENCE|             THEFT| 1948|
|            SIDEWALK|           ROBBERY| 1657|
|           APARTMENT|   CRIMINAL DAMAGE| 1515|
|              STREET|      PROSTITUTION| 1464|
|           APARTMENT|             THEFT| 1397|
|           RESIDENCE| DECEPTIVE PRACTICE| 1396|
|           RESIDENCE|   CRIMINAL DAMAGE| 1379|
|              STREET|     OTHER OFFENSE| 1312|
|           APARTMENT|          BURGLARY| 1269|
|    DEPARTMENT STORE|             THEFT| 1239|
+--------------------+------------------+-----+
only showing top 20 rows
```

- Crime type and Location for Community Area Community Area 8:

```
>>> df.where (F.col("Community Area") == "8").groupBy("Location Description", "Primary Type").agg(F.count("Primary Type").alias("count")).orderBy(F.col("count").desc())
.show()
+--------------------+------------------+-----+
|Location Description|      Primary Type|count|
+--------------------+------------------+-----+
|              STREET|             THEFT| 3333|
|    DEPARTMENT STORE|             THEFT| 3089|
|          RESTAURANT|             THEFT| 2474|
|  SMALL RETAIL STORE|             THEFT| 1945|
|               OTHER|             THEFT| 1667|
|       BAR OR TAVERN|             THEFT| 1568|
|            SIDEWALK|           BATTERY| 1263|
|            SIDEWALK|             THEFT| 1237|
|PARKING LOT/GARAG...|             THEFT| 1235|
|              STREET|   CRIMINAL DAMAGE| 1079|
|         HOTEL/MOTEL|             THEFT| 1062|
|              STREET|           BATTERY|  887|
|       BAR OR TAVERN|           BATTERY|  766|
|  GROCERY FOOD STORE|             THEFT|  729|
|           RESIDENCE| DECEPTIVE PRACTICE|  710|
|               OTHER| DECEPTIVE PRACTICE|  671|
|           RESIDENCE|             THEFT|  659|
|              STREET|MOTOR VEHICLE THEFT|  603|
|           APARTMENT|           BATTERY|  582|
|    DEPARTMENT STORE| DECEPTIVE PRACTICE|  581|
+--------------------+------------------+-----+
only showing top 20 rows

>>>
```

- Crime type and Location for Community Area Community Area 43:

```
>>> df.where (F.col("Community Area") == "43").groupBy("Location Description", "Primary Type").agg(F.count("Primary Type").alias("count")).orderBy(F.col("count").desc()
).show()
+--------------------+------------------+-----+
|Location Description|      Primary Type|count|
+--------------------+------------------+-----+
|           APARTMENT|           BATTERY| 4789|
|           APARTMENT|          BURGLARY| 2963|
|              STREET|             THEFT| 2288|
|           APARTMENT|   CRIMINAL DAMAGE| 1824|
|            SIDEWALK|         NARCOTICS| 1639|
|              STREET|   CRIMINAL DAMAGE| 1576|
|            SIDEWALK|           BATTERY| 1536|
|           RESIDENCE|           BATTERY| 1388|
|           APARTMENT|             THEFT| 1305|
|              STREET|MOTOR VEHICLE THEFT| 1288|
|           RESIDENCE|     OTHER OFFENSE| 1270|
|              STREET|           BATTERY| 1169|
|           RESIDENCE|             THEFT| 1088|
|           APARTMENT|           ASSAULT| 1078|
|              STREET|         NARCOTICS| 1042|
|           RESIDENCE|   CRIMINAL DAMAGE|  971|
|            SIDEWALK|           ROBBERY|  931|
|           APARTMENT|     OTHER OFFENSE|  882|
|           RESIDENCE|          BURGLARY|  831|
|           RESIDENCE| DECEPTIVE PRACTICE|  674|
+--------------------+------------------+-----+
only showing top 20 rows
```

## Tool 4 –HiveQL

We found HiveQL to be the most robust tool of them all, thanks to its similarity to SQL. Furthermore, we were able to achieve superior performance in our queries by partitioning data on the ‘community areas’ and ‘ward number’ fields. For most of our HiveQL queries, we were interested in top 20 aggregations by those fields, and HiveQL proved to be a powerful tool that allowed us to optimize our queries.

Creating new hive database for project:

```
hive> create database project;
OK
Time taken: 1.03 seconds
hive> use project;
OK
Time taken: 0.217 seconds
```

Creating empty tables in the new database:

```
hive> CREATE TABLE project.chicago_crimes (row_num INT, id INT, case_num STRING, time_stamp STRING, block STRING,
    > iucr INT, crime_type STRING, crime_desc STRING, location_desc STRING, arrest STRING,
    > domestic STRING, beat INT, district INT, ward INT, community_area INT, fbi_code STRING,
    > x_coord INT, y_coord INT, year INT, update_date STRING, latitude FLOAT, longitude FLOAT, location STRING)
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE;
OK
Time taken: 0.561 seconds
```

```
hive> CREATE TABLE project.chicago_income (community_area INT, community_name STRING,
    > overcrowded_housing FLOAT, poverty_rate FLOAT, unemployment_rate FLOAT,
    > no_education FLOAT, under18_over64 FLOAT, income FLOAT, hardship INT)
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE
    > TBLPROPERTIES ("skip.header.line.count"="1");
OK
Time taken: 0.552 seconds
```

```
hive> CREATE TABLE project.soc_ind (ward INT, ward_name STRING, total_pop FLOAT,
    > white_pop FLOAT, black_pop FLOAT, asian_pop FLOAT, latino_pop FLOAT,
    > other_pop FLOAT)
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE;
OK
Time taken: 0.548 seconds
```

Loading data from HDFS into the new empty tables:

```
hive> load data inpath '/user/root/project/Chicago_Crimes_2012_to_2017.csv' overwrite into table project.chicago_crimes;
Loading data to table project.chicago_crimes
chgrp: changing ownership of 'hdfs://sandbox-hdp.hortonworks.com:8020/apps/hive/warehouse/project.db/chicago_crimes/Chicago_Crimes_2012_to_2017.csv': User null does not belong to hadoop
Table project.chicago_crimes stats: [numFiles=1, numRows=0, totalSize=366804568, rawDataSize=0]
OK
Time taken: 1.145 seconds
```

```
hive> load data inpath '/user/root/project/chicago_income.csv' overwrite into table project.chicago_income;
Loading data to table project.chicago_income
chgrp: changing ownership of 'hdfs://sandbox-hdp.hortonworks.com:8020/apps/hive/warehouse/project.db/chicago_income/chicago_income.csv': User null does not belong to hadoop
Table project.chicago_income stats: [numFiles=1, numRows=0, totalSize=3865, rawDataSize=0]
OK
Time taken: 1.098 seconds
```

```
hive> load data inpath '/user/root/project/socioeconomic_indicators.csv' overwrite into table project.socioeconomic_indicators;
Loading data to table project.socioeconomic_indicators
chgrp: changing ownership of 'hdfs://sandbox-hdp.hortonworks.com:8020/apps/hive/warehouse/project.db/socioeconomic_indicators/soc
adoop
Table project.socioeconomic_indicators stats: [numFiles=1, numRows=0, totalSize=2903, rawDataSize=0]
OK
Time taken: 1.098 seconds
```

Ensuring that the data types in each table are correct:

```
hive> describe chicago_crimes;
OK
col_name        data_type       comment
row_num                 int
id                      int
case_num                string
time_stamp              string
block                   string
iucr                    int
crime_type              string
crime_desc              string
location_desc           string
arrest                  string
domestic                string
beat                    int
district                int
ward                    int
community_area          int
fbi_code                string
x_coord                 int
y_coord                 int
year                    int
update_date             string
latitude                float
longitude               float
location                string
Time taken: 0.436 seconds, Fetched: 23 row(s)
```

```
hive> describe soc_ind;
OK
col_name        data_type       comment
ward                    int
ward_name               string
total_pop               float
white_pop               float
black_pop               float
asian_pop               float
latino_pop              float
other_pop               float
Time taken: 0.436 seconds, Fetched: 8 row(s)
```

```
hive> describe chicago_income;
OK
col_name        data_type       comment
community_area          int
community_name          string
overcrowded_housing     float
poverty_rate            float
unemployment_rate       float
no_education            float
under18_over64          float
income                  float
hardship                int
Time taken: 0.437 seconds, Fetched: 9 row(s)
```

Sample query for calculating crime per capita:

This query required data from two tables, the Chicago Crimes dataset, and the population distribution dataset. Thus, an inner join on the common key, ward number, was used. A calculated field was queried to determine crime per capita. This was executed by taking the count of crimes from one dataset, and dividing it by the corresponding population. Then, the results were grouped by ward and placed in descending order so that the top ten wards with the highest crime rates were visible. In the example below, the year 2012 was filtered out. We repeated this query for different years to check for patterns.

```
hive> SELECT s_i.ward_name, ROUND(COUNT(c_c.id)/(s_i.total_pop),2) AS crime_per_capita
    > FROM c_c JOIN s_i ON (c_c.ward=s_i.ward)
    > WHERE c_c.year=2012
    > GROUP BY s_i.ward_name,s_i.total_pop ORDER BY crime_per_capita DESC LIMIT 10;
Query ID = root_20211124015513_aa5a2e67-e9bd-49eb-94d3-d3b60d908eab
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1637705561973_0006)

----------------------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 4 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
Reducer 3 ......   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 4.04 s
----------------------------------------------------------------------------------------------
OK
Ervin   0.28
Reilly  0.25
Scott   0.24
Moore   0.22
Burnett 0.22
Sawyer  0.21
Hopkins 0.21
Cochran 0.21
Lopez   0.19
Foulkes 0.18
Time taken: 4.591 seconds, Fetched: 10 row(s)
```

The query below on the left-hand side was used to extract the communities with the highest number of crimes. Since the community names were not present in the Chicago Crimes dataset, we performed an inner join on the socioeconomic indicators dataset to look up the name. The name and count of crimes, grouped by name, were presented in descending order.

The basic query on the right-hand side was used to find the queries with the highest poverty rates. We used these two queries to determine the overlap between crime and poverty rates. We also queried for the communities with the lowest poverty rates (not shown in the report). The only difference is that the results were displayed in ascending order.

```
hive> SELECT c_i.community_name, COUNT(c_c.id) AS cnt
    > FROM c_c JOIN c_i ON (c_c.community_area = c_i.community_area)
    > GROUP BY community_name
    > ORDER BY cnt DESC LIMIT 20;
Query ID = root_20211201221841_5cc08de0-f04c-4721-a57f-2b4a345b0ef8
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1638396867645_0002)

----------------------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 4 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
Reducer 3 ......   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 25.34 s
----------------------------------------------------------------------------------------------
OK
Austin  95002
Near North Side 53102
South Shore     48114
Humboldt park   45977
North Lawndale  45020
Near West Side  42288
Auburn Gresham  40603
West Englewood  40495
West Town       40368
Loop    39028
Roseland        36858
Englewood       36689
Greater Grand Crossing  35888
Chicago Lawn    34014
Lake View       33535
Chatham 32509
West Garfield Park      31077
Logan Square    29954
East Garfield Park      27722
Belmont Cragin  27348
Time taken: 26.694 seconds, Fetched: 20 row(s)
```

```
hive> SELECT community_name, poverty_rate FROM c_i ORDER BY poverty_rate DESC LIMIT 20;
Query ID = root_20211123044520_ccc23ae5-8267-4843-9c3b-6a2a7930fe53
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1637618229753_0009)

----------------------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 1.69 s
----------------------------------------------------------------------------------------------
OK
community_name  poverty_rate
Riverdale       56.5
Fuller Park     51.2
Englewood       46.6
North Lawndale  43.1
East Garfield Park      42.4
Washington Park 42.1
West Garfield Park      41.7
Armour Square   40.1
Oakland 39.7
West Englewood  34.4
Humboldt park   33.9
Burnside        33.0
South Shore     31.1
South Lawndale  30.7
Woodlawn        30.7
South Chicago   29.8
Douglas 29.6
Greater Grand Crossing  29.6
Grand Boulevard 29.3
South Deering   29.2
Time taken: 2.105 seconds, Fetched: 20 row(s)
```

In the query below, once again we queried the count of crimes by community. However, we identified a set of crime types that appeared violent and filtered on them, displaying the top 20 communities where these crimes were most prevalent. This was used in conjunction with the query above on the right-hand side to see if there was a relationship between violent crimes and poverty.

```
hive> SELECT c_i.community_name, COUNT(c_c.id) as cnt
    > FROM c_c JOIN c_i ON (c_c.community_area = c_i.community_area)
    > WHERE c_c.crime_type IN ('ASSAULT','BATTERY','CRIM SEXUAL ASSAULT','HOMICIDE','KIDNAPPING','HUMAN TRAFFICKING')
    > GROUP BY c_i.community_name ORDER BY cnt DESC LIMIT 20;
Query ID = root_20211129072739_7008da1e-603b-48af-9519-005ea673d177
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1638169808689_0002)

----------------------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 4 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
Reducer 3 ......   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 11.72 s
----------------------------------------------------------------------------------------------
OK
Austin  25873
South Shore     14291
North Lawndale  13012
West Englewood  12304
Auburn Gresham  12033
Englewood       11741
Humboldt park   11519
Greater Grand Crossing  11006
Roseland        10356
Near North Side 9228
Chicago Lawn    9221
Chatham 8747
South Chicago   8403
Near West Side  8287
New City        7948
East Garfield Park      7518
West Garfield Park      7209
West Town       7154
South Lawndale  6976
Belmont Cragin  6971
```

The query below on the left-hand side was used to determine the percentage crimes that led to an arrest for violent crimes and grouped by the communities that had the highest arrest rates for those crime types. The results were displayed in descending order.

The query below on the right-hand side is identical except for that is filtered for non-violent crimes.

These two queries, in conjunction with the query on highest poverty rates, were used to determine whether there was an asymmetry in arrests in poor/wealthy communities for violent and non-violent crimes.

```
hive> SELECT c_i.community_name, ROUND(SUM(CASE WHEN c_c.arrest='True' THEN 1 ELSE 0 END)/COUNT(*),2) AS arrest_percentage
    > FROM c_c JOIN c_i on (c_c.community_area = c_i.community_area)
    > WHERE c_c.crime_type IN ('ASSAULT','BATTERY','CRIM SEXUAL ASSAULT','HOMICIDE','KIDNAPPING','HUMAN TRAFFICKING')
    > GROUP BY c_i.community_name
    > ORDER BY arrest_percentage
    > DESC LIMIT 20;
Query ID = root_20211123225754_ab96be7f-8688-477c-8be5-a07a4c6407c7
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1637705561973_0003)

----------------------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 4 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
Reducer 3 ......   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 5.33 s
----------------------------------------------------------------------------------------------
OK
c_i.community_name      arrest_percentage
Loop    0.33
Near South Side 0.28
Lake View       0.28
O'Hare  0.27
East Side       0.26
Armour Square   0.26
South Lawndale  0.25
Garfield Ridge  0.25
Edgewater       0.25
Pullman 0.25
Bridgeport      0.25
Roseland        0.25
Chicago Lawn    0.25
Clearing        0.25
Avondale        0.25
Near North Side 0.25
Woodlawn        0.24
New City        0.24
Rogers Park     0.24
Gage Park       0.24
Time taken: 9.018 seconds, Fetched: 20 row(s)
```

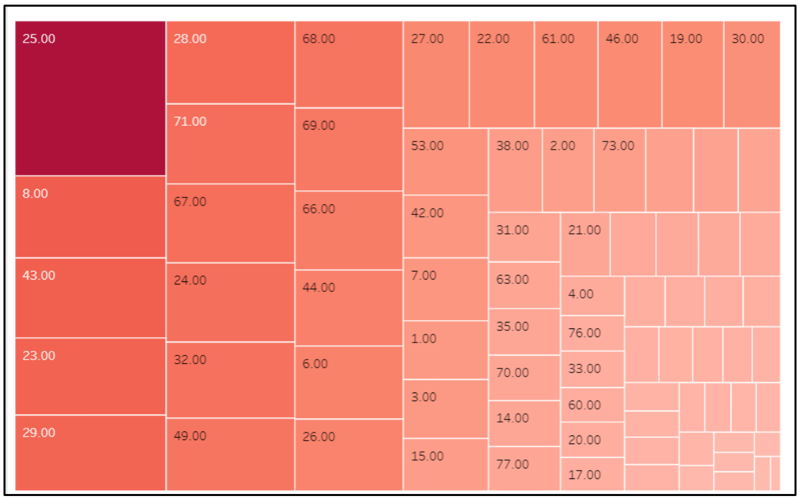
```
hive> SELECT c_i.community_name, ROUND(SUM(CASE WHEN c_c.arrest='True' THEN 1 ELSE 0 END)/COUNT(*),2) AS arrest_percentage
    > FROM c_c JOIN c_i on (c_c.community_area = c_i.community_area)
    > WHERE c_c.crime_type NOT IN ('ASSAULT','BATTERY','CRIM SEXUAL ASSAULT','HOMICIDE','KIDNAPPING','HUMAN TRAFFICKING')
    > GROUP BY c_i.community_name
    > ORDER BY arrest_percentage
    > DESC LIMIT 20;
Query ID = root_20211123230938_b9789de2-ea4f-4e62-a7d0-7f1f990ff36f
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1637705561973_0004)

----------------------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 4 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
Reducer 3 ......   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 5.49 s
----------------------------------------------------------------------------------------------
OK
c_i.community_name      arrest_percentage
West Garfield Park      0.58
East Garfield Park      0.48
Humboldt park   0.45
North Lawndale  0.44
Austin  0.43
West Englewood  0.36
Fuller Park     0.36
Englewood       0.35
New City        0.34
Woodlawn        0.31
Chicago Lawn    0.3
Greater Grand Crossing  0.3
Roseland        0.29
South Chicago   0.29
West Pullman    0.29
Auburn Gresham  0.29
Chatham 0.29
Uptown  0.28
South Lawndale  0.27
Washington Park 0.27
Time taken: 9.137 seconds, Fetched: 20 row(s)
```

## Key Insights

Throughout this project, we found many insights that would be worth mentioning. However, we believe that the following five insights are the most impactful.

### Insight 1

We utilized a heatmap to display the community areas with the highest occurrence of crime between 2012 and 2017. The top five communities with the highest rates of crime are 25, 8, 43, 23, and 29.




Furthermore, we dove deeper into the individual communities to find the most common crime types and locations within those areas.

- The most common crimes in those communities were: theft, battery, assault, criminal damage, and deceptive practice. These crimes accounted for 68% of all crime in the area.
- The most common locations for those types of crimes are: street, sidewalks, residence, and apartment

### Insight 2

We leveraged the queries mentioned in previous sections to discover a few links between crime and poverty. The table on the left displays the twenty communities with the highest crime rates, in descending order. The table on the right displays the twenty most impoverished communities. Here is what we can glean from the tables:

a. 40% of the poorest 20 communities are among those with the highest crime rates

b. 50% of the poorest 10 communities are among those with the highest crime rates

c. 60% of the top 5 most crime ridden communities are among the poorest.

We believe this somewhat illustrates that there is a strong link between poverty and crime.

| Most Crime-Ridden Communities |
|---|
| Austin |
| Near North Side |
| South Shore |
| Humboldt Park |
| North Lawndale |
| Near West Side |
| Auburn Gresham |
| West Englewood |
| West Town |
| Loop |
| Roseland |
| Englewood |
| Greater Grand Crossing |
| Chicago Lawn |
| Lake View |
| Chatham |
| West Garfield Park |
| Logan Park |
| East Garfield Park |
| Belmont Cragin |

| Poorest 20 Communities |
|---|
| Riverdale |
| Fuller Park |
| Englewood |
| North Lawndale |
| East Garfield Park |
| Washington Park |
| West Garfield Park |
| Armour Square |
| Oakland |
| West Englewood |
| Humboldt Park |
| Burnside |
| South Shore |
| South Lawndale |
| Woodlawn |
| South Chicago |
| Douglas |
| Greater Grand Crossing |
| Grand Boulevard |
| South Deering |

## Insight 3

This insight builds upon the previous one. We wanted to analyze the link between arrest rates and poverty. Using similar queries, we extracted the list of communities with the highest arrest rates in descending order. We compared it with the list of the 20 poorest communities. Here is what we gleaned from the data:

a. 60% of the poorest 20 communities are among those with the highest arrest rates

b. 70% of the poorest 10 communities are among those with the highest arrest rates

c. 80% of the 10 communities with the highest arrest rates are also among those that are the poorest

We believe this is significant because the table on the left-hand side displays the arrest percentage, and not the raw number of arrests. One would expect the number of arrests to correlate positively with the number of crimes. However, these arrest percentages are normalized for the occurrence of crime. This shows that the poorest communities may be disproportionately targeted by police.

| Communities with High Arrest Rates |
|---|
| West Garfield Park |
| East Garfield Park |
| Humboldt Park |
| Austin |
| North Lawndale |
| Fuller Park |
| West Englewood |
| New City |
| Englewood |
| Woodlawn |
| Chicago Lawn |
| Roseland |
| Greater Grand Crossing |
| South Chicago |
| West Pullman |
| Uptown |
| Auburn Gresham |
| Chatham |
| South Lawndale |
| Washington Park |

| Poorest 20 Communities |
|---|
| Riverdale |
| Fuller Park |
| Englewood |
| North Lawndale |
| East Garfield Park |
| Washington Park |
| West Garfield Park |
| Armour Square |
| Oakland |
| West Englewood |
| Humboldt Park |
| Burnside |
| South Shore |
| South Lawndale |
| Woodlawn |
| South Chicago |
| Douglas |
| Greater Grand Crossing |
| Grand Boulevard |
| South Deering |

## Insight 4

This insight also builds upon the previous one. We wanted to compare arrest rates for the type of crime being committed. We subjectively grouped together the following crime types into the 'violent' bucket: Assault, battery, criminal sexual assault, homicide, kidnapping, human trafficking. We assumed that all crimes that do not fall into this group are 'non-violent'.

a. Only 15% of the poorest communities are among those with the highest arrest rates for violent crimes

The hypothesis was that poorer communities would have a lot of arrests for violent crimes, but that does not appear to be the case.

| Communities with High Arrest Rates for Violent Crimes |
|---|
| Loop |
| Near South Side |
| Lake View |
| O'Hare |
| East Side |
| Armour Square |
| South Lawndale |
| Garfield Ridge |
| Edgewater |
| Pullman |
| Bridgeport |
| Roseland |
| Chicago Lawn |
| Clearing |
| Avondale |
| Near North Side |
| Woodlawn |
| New City |
| Rogers Park |
| Gage Park |

| Poorest 20 Communities |
|---|
| Riverdale |
| Fuller Park |
| Englewood |
| North Lawndale |
| East Garfield Park |
| Washington Park |
| West Garfield Park |
| Armour Square |
| Oakland |
| West Englewood |
| Humboldt Park |
| Burnside |
| South Shore |
| South Lawndale |
| Woodlawn |
| South Chicago |
| Douglas |
| Greater Grand Crossing |
| Grand Boulevard |
| South Deering |

The following set of tables display the communities with the highest arrest rates for non-violent crimes, compared with the poorest 20 communities. The results are shockingly different from what was displayed above.

a. 60% of the poorest 20 communities are among those with the highest arrest rates for non-violent crimes

b. 70% of the poorest 10 communities are among those with the highest arrest rates for non-violent crimes

c. 80% of the 10 communities with the highest arrest rates are among the poorest

These results contradicted our hypotheses. We hypothesized that violent crimes in poor communities would lead to more arrests than non-violent ones. The data shows the opposite to be true.

| Communities with High Arrest Rates for non-Violent Crimes | Poorest 20 Communities |
|---|---|
| West Garfield Park | Riverdale |
| East Garfield Park | Fuller Park |
| Humboldt Park | Englewood |
| North Lawndale | North Lawndale |
| Austin | East Garfield Park |
| West Englewood | Washington Park |
| Fuller Park | West Garfield Park |
| Englewood | Armour Square |
| New City | Oakland |
| Woodlawn | West Englewood |
| Chicago Lawn | Humboldt Park |
| Greater Grand Crossing | Burnside |
| Roseland | South Shore |
| South Chicago | South Lawndale |
| West Pullman | Woodlawn |
| Auburn Gresham | South Chicago |
| Chatham | Douglas |
| Uptown | Greater Grand Crossing |
| South Lawndale | Grand Boulevard |
| Washington Park | South Deering |

Furthermore, we queried the 20 wealthiest communities to cross-reference the list against those with the highest arrest rates for both violent and non-violent crimes.

a. Not a single wealthy community was among the list of communities with the highest arrest rates

This is significant because we are comparing arrest rates which is a normalized value, and not the raw number of arrests. These set of tables suggest that asymmetric policing policies may be at play in the City of Chicago.

| Communities with High Arrest Rates for Violent Crimes |
|---|
| Loop |
| Near South Side |
| Lake View |
| O'Hare |
| East Side |
| Armour Square |
| South Lawndale |
| Garfield Ridge |
| Edgewater |
| Pullman |
| Bridgeport |
| Roseland |
| Chicago Lawn |
| Clearing |
| Avondale |
| Near North Side |
| Woodlawn |
| New City |
| Rogers Park |
| Gage Park |

| Communities with High Arrest Rates for non-Violent Crimes |
|---|
| West Garfield Park |
| East Garfield Park |
| Humboldt Park |
| North Lawndale |
| Austin |
| West Englewood |
| Fuller Park |
| Englewood |
| New City |
| Woodlawn |
| Chicago Lawn |
| Greater Grand Crossing |
| Roseland |
| South Chicago |
| West Pullman |
| Auburn Gresham |
| Chatham |
| Uptown |
| South Lawndale |
| Washington Park |

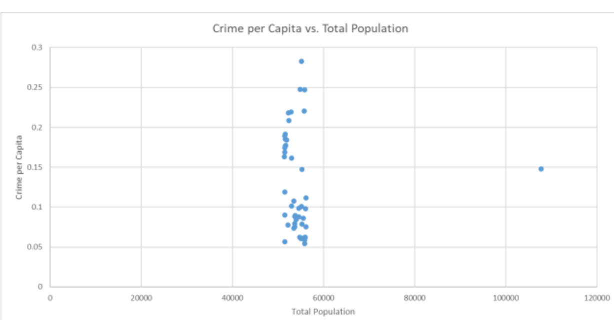
| Wealthiest 20 Communities |
|---|
| Edison Park |
| Mount Greenwood |
| Beverly |
| Norwood Park |
| Forest Glen |
| North Center |
| Jefferson Park |
| Garfield Ridge |
| Clearing |
| Ashburn |
| Dunning |
| Lincoln Square |
| Lake View |
| Calumet Heights |
| Portage Park |
| Lincoln Park |
| Near North Side |
| Irving Park |
| Morgan Park |
| North Park |

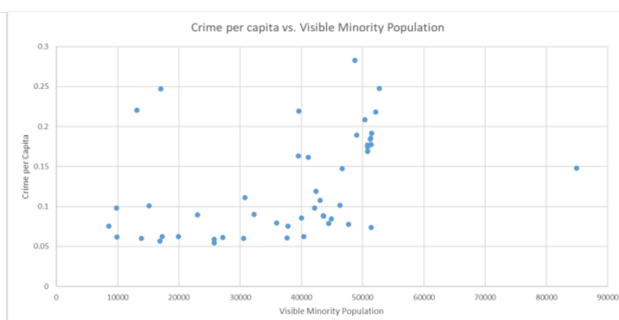
## Insight 5

We wanted to look at some charts to better understand the link between crime, population distribution, and socioeconomic factors.

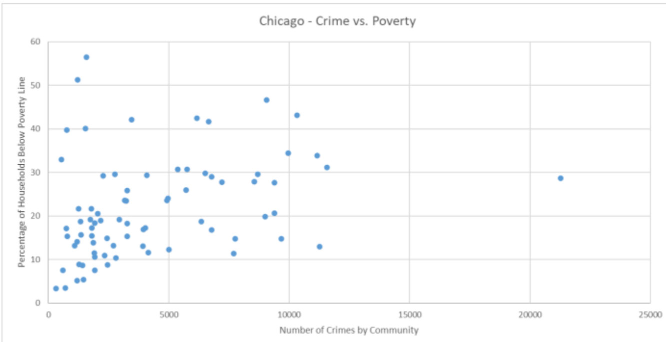
On the left-hand side, we plot the crime per capita against the total population. Each dot indicates a community area. Based on this chart, there does not appear to be a correlation between crime rates and population as seen from the columnar cluster.

On the right-hand side, we plot crime per capita against the visible minority population. there appears to be a positive correlation between the two.

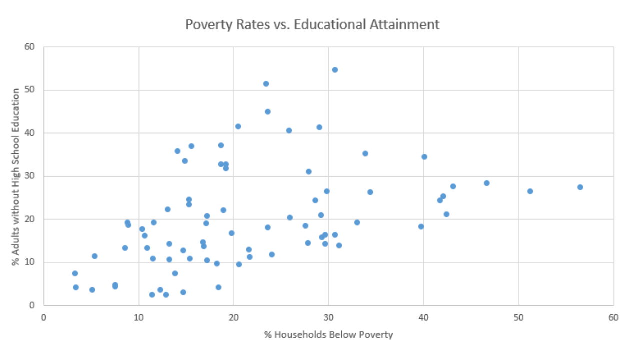


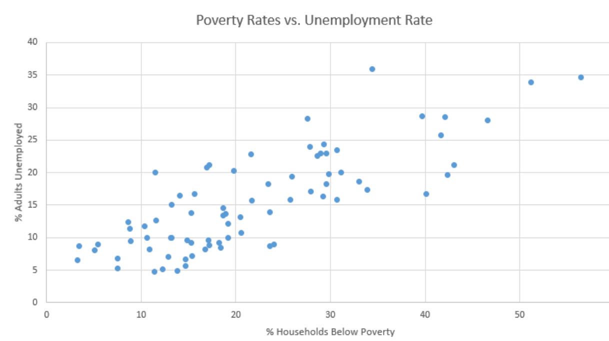




We wanted to explore potential underlying socioeconomic factors behind this correlation between crime per capita and the visible minority population. The chart below plots the percentage of households below the poverty line against the number of crimes. There is a clear positive correlation between the two.




We also wanted to see if there is correlation between poverty rates and other socioeconomic factors. In the charts below, we plotted the poverty rate against the educational attainment and unemployment. Both charts appear to show strong positive correlation.




From the visuals illustrated in this section, there appears to be a positive correlation between crime and relevant socioeconomic factors.

## Future Work

For our project, we extracted insights from a combination of three data sets. We leveraged the Chicago Crimes data and overlayed it with socioeconomic factors and population breakdown. In our analysis, we found evidence of a link between crime rates and underlying socioeconomic factors. We discovered some evidence to suggest that policing practices disproportionately impacted poorer communities in a negative manner.

For future work, it would be interesting to compare levels of policing across different communities to check if impoverished communities are policed more heavily than their wealthier counterparts. Furthermore, it may be prudent to do analysis on the types of jobs available in these communities and whether education levels of poorer communities meet the needs of the local market. This information can be leveraged by planners in the state of Illinois to better understand how to allocate their resources more effectively.

## References

The references we used were only for the purposes of building out queries for the tools we used in this course. No other materials were used for any other purpose in this report.


https://www.kaggle.com/currie32/crimes-in-chicago(Chicago Crimes dataset)

https://data.cityofchicago.org/Health-Human-Services/Per-Capita-Income/r6ad-wvtk(socioeconomic indicators dataset)

https://docs.google.com/spreadsheets/d/1sxM-JajdrC7R1VZ_sHjUwkTQ0qs2z7a7jFbCblTii3Q/edit#gid=1503084939(Population distribution dataset)

https://spark.apache.org/docs/latest/rdd-programming-guide.html(Spark RDD guide for queries)

https://www.tutorialspoint.com/apache_spark/apache_spark_rdd.htm(Spark RDD guide for queries)

https://sparkbyexamples.com/spark/different-ways-to-create-a-spark-dataframe/ (Spark SQL guide for queries)

https://towardsdatascience.com/spark-essentials-how-to-read-and-write-data-with-pyspark-5c45e29227cd (Spark SQL guide for queries)