Strategies for Large Scale Data Cleaning

Yash Limbachiya
yvl208@nyu.edu
New York University

Akshita Lakkad
acl10003@nyu.edu
New York University

Shubham Gundawar
ssg9763@nyu.edu
New York University

ABSTRACT

The project focuses on various methods and strategies used to clean data within the NYPD Complaint Data Historic dataset. We have presented strategies and methods in our project which aim to effectively clean data in other similar datasets. We also take into consideration various factors such as Date, Location, time of occurrence of events, Suspect race, and many other factors to effectively clean data. The cleaned data can be used to generate a descriptive summary of the features and contents. The dataset used (NYPD Complaint Data Historic) includes all valid felony, misdemeanor, and violation crimes reported to the New York City Police Department (NYPD) from 2006 to the end of last year (2020). Further analysis of this cleaned data can be done keeping in mind a variety of factors such as socio-economic impact, income groups, age groups, gender, race.

KEYWORDS

complaint, data cleaning, profiling, NYPD, crime, arrest, statistics

1 INTRODUCTION AND APPROACH

We have acquired data sets containing data related to the complaints to the NYPD in New York City. Some other data sets have information about the shootings in NYC, other crimes, and arrest data by the NYPD. We will use all of this data to analyze and get some conclusions/answers for some questions, only a few are listed below:

- What techniques are useful for data cleaning?
- How can data cleaning be scaled to multiple datasets?
- Can similar techniques for data cleaning be used across different datasets having similar columns?
- To what extent can you generalize the techniques of data cleaning that can be used across various disparate datasets?

2 DATA ACQUISITION

Data acquisition is the process of gathering and storing data. We have followed the four V's of Big Data during data acquisition and gathering. They are Volume, Variety, Velocity, and Value. The datasets are acquired in such a way that they relate to one another and cleaning strategies used in one can be replicated to another.

Data acquisition consists of one major area - Finding similar datasets. For our project, we have used Socrata Open Data Network to access similar datasets. Open Data Network is a global search engine that allows you to search across thousands of datasets from hundreds of open data catalogs. We used the Open Clean python library to find all the datasets that contain similar columns like that of the NYPD Complaint Data Historic to make a comprehensive list of similar datasets on which the cleaning strategies could be tested.

Following is the list of a few datasets we gathered:

NYPD Complaint Data (Historic)
NYPD Complaint Data Current (Year To Date)
NYPD Arrest Data (Year to Date)
NYPD Arrests Data (Historic)
NYPD Shooting Incident Data
NYPD Criminal Court Summons
NYPD Criminal Court Summons Incident Level Data
NYPD Hate Crimes
NYC Crime Stats

2.1 List and Description of Data sets

We have listed down a description of a few of the important datasets required for our project:

- Dataset 1: NYPD Complaint Data (Historic): This dataset includes all valid felony, misdemeanor, and violation crimes reported to the New York City Police

Department (NYPD) from 2006 to the end of last year (2020).

- Dataset 2: NYPD Shooting Incident Data (Historic): List of every shooting incident that occurred in NYC going back to 2006 through the end of the previous calendar year. This is a breakdown of every shooting incident that occurred in NYC going back to 2006 through the end of the previous calendar year. This data is manually extracted every quarter and reviewed by the Office of Management Analysis and Planning before being posted on the NYPD website. Each record represents a shooting incident in NYC and includes information about the event, the location, and the time of occurrence.
- Dataset 3: NYPD Shooting Incident Data (Year To Date): List of every shooting incident that occurred in NYC during the current calendar year. This is a breakdown of every shooting incident that occurred in NYC during the current calendar year. This data is manually extracted every quarter and reviewed by the Office of Management Analysis and Planning before being posted on the NYPD website.
- Dataset 4: NYPD Arrests Data (Historic): List of every arrest in NYC going back to 2006 through the end of the previous calendar year. This is a breakdown of every arrest affected in NYC by the NYPD going back to 2006 through the end of the previous calendar year. This data is manually extracted every quarter and reviewed by the Office of Management Analysis and Planning before being posted on the NYPD website. Each record represents an arrest affected in NYC by the NYPD and includes information about the type of crime, the location, and the time of enforcement. In addition, information related to suspect demographics is also included. This data can be used by the public to explore the nature of police enforcement activity.
- Dataset 5: NYPD Criminal Court Summons Incident Level Data: List of every criminal summons issued in NYC during the current calendar year. This is a breakdown of every criminal summons issued in NYC by the NYPD during the current calendar year. This data is manually extracted every quarter and reviewed by the Office of Management Analysis and Planning before being posted on the NYPD website. Each record represents a criminal summons issued in NYC by the NYPD and includes information about the type of crime, the location, and the time of enforcement. In addition, information related to suspect demographics is also included. This data can be used by the public to explore the nature of police enforcement activity.

2.2 Challenges Involved in acquiring data

- Purpose and Approach:

  Dataset 1: According to the requirement of the topic of the project, we had to acquire this data from the NYC open data portal. We were able to get the required columns in this data set. The data set contained the columns which were suitable for answering the questions which our title concludes.
  Dataset 2: This data is of the shooting incident data. This dataset was also acquired from the NYC open data portal.
  Rest of the datasets were available through the NYC Open Data and Socrata global search engine.

- Challenges faced:

  The goal of this project is to scale the data cleaning techniques on other similar datasets. So, naturally, we needed to find datasets that had a significant number of common columns like Date, Borough, Age Group, Sex, Race, Lat_Lon, etc.

  We were able to find a few datasets which had such common columns, like the NYPD Arrest Data, NYPD Shooting Data, NYPD Criminal Court Summons, etc.
  Also, all these datasets were available in 2 timelines. One was historic meaning it contained data from a long time ago (from 2006, in most cases) and the other being current year to date.

  We were easily able to find the NYC open datasets, so we went with those. We found many datasets for Gun Violence, but those were only available from paid sources, so we acquired the datasets which were readily available. For eg: NYPD Arrest Data had columns like ARREST_DATE, KY_CD, OFNS_DESC, LAW_CAT_CD, ARREST_BORO, AGE_GROUP, PERP_SEX, PERP_RACE, etc. which were very similar to the columns we had in our original given dataset (NYPD Complaint Data Historic). And they were chosen so that the cleaning techniques which we were using in our dataset could be reused and checked whether they are working correctly on varied different datasets.

  Almost all other datasets had such similar columns but a few datasets did not have anything in common. We

were able to find 7-8 datasets easily on NYC Open Data. We used Auctus - NYU VIDA's dataset searching tool where we found similar datasets on NYC Open Data.

3 TECHNOLOGIES USED

- Google Colab
- Jupyter Notebook
- PySpark
- Pandas
- Numpy
- Open Clean Python Library

4 ARCHITECTURE

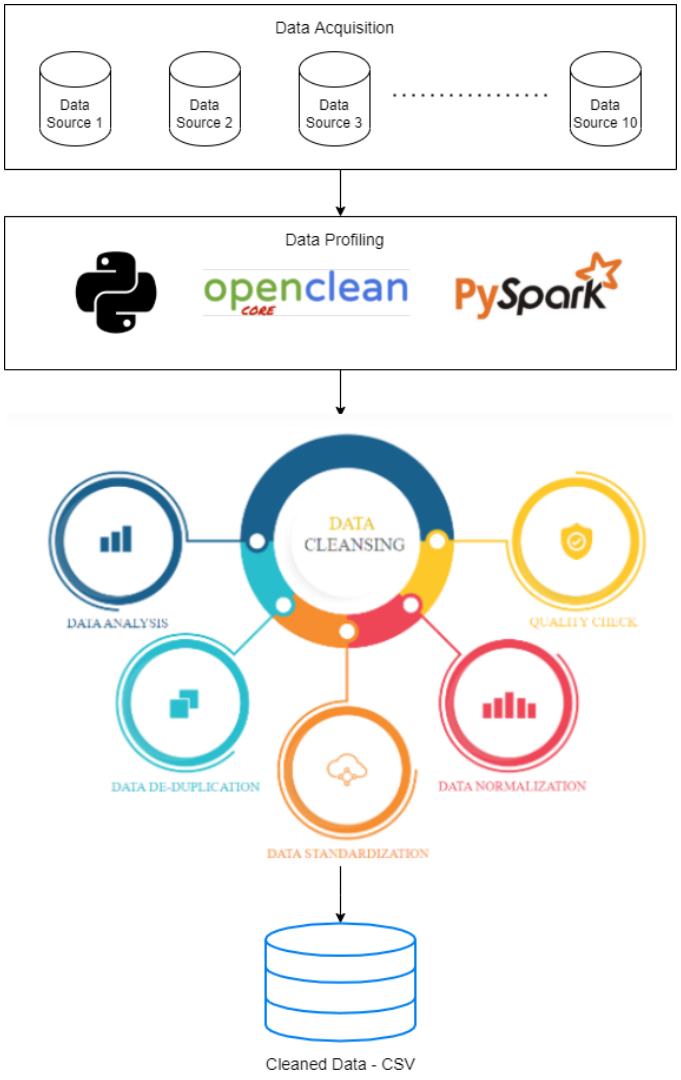

Figure 1: Architecture

5 DATA PROFILING

It is challenging to ingest data from external sources particularly when we are not aware of the structure and the contents of the data. Thus, it becomes essential to perform data profiling to understand how the data looks before we begin cleaning the data in the actual sense. Data profiling is the process of running an analysis on source data to understand its structure and content.

Data profiling is performed to typically address:

- Before ingesting data from a new source, to discover if the data is in a suitable form
- Identify data quality issues
- Identify the necessary corrections to be made in order to make the data more consistent and uniform

We performed data profiling on our data to get the following insights:

- Structure Discovery:
  - Shape of the data
  - Names of columns and what the data represents
- Content Discovery:
  - Data types of the columns
  - Check if the column has multiple data types
  - Identify nullable columns
- Cardinality of Data:
  - Number of distinct values in each column
  - Count of most frequent values pertaining to a certain column
- Statistical Analysis:
  - Extract minimum, maximum, mean values of numerical columns
  - Calculate the uniqueness and entropy of each column in the dataset

We have used openclean python library's Column Profiler package to extract information about the number of null and distinct values for each column in the dataset. We have extracted the minimum and maximum values of datasets. Thus, through thorough data profiling, we were able to draw conclusions about the columns that have maximum null values and the importance of the columns and take necessary steps in cleaning to minimize the null values and make the dataset as concise as possible.

	total	empty	distinct	uniqueness	entropy
CMPLNT_NUM	7375993	0	7373143	9.996136e-01	22.813633
CMPLNT_FR_DT	7375993	655	8606	1.166862e-03	12.425578
CMPLNT_FR_TM	7375993	48	1441	1.953648e-04	8.136466
CMPLNT_TO_DT	7375993	1704204	6825	1.203324e-03	12.417984
CMPLNT_TO_TM	7375993	1699541	1441	2.538558e-04	8.862856
ADDR_PCT_CD	7375993	2166	77	1.044234e-05	6.148690
RPT_DT	7375993	0	5479	7.428152e-04	12.405384
KY_CD	7375993	0	74	1.003255e-05	4.170727
OFNS_DESC	7375993	18823	71	9.650450e-06	4.006583
PD_CD	7375993	6278	432	5.861828e-05	5.913459
PD_DESC	7375993	6278	422	5.726137e-05	5.904814
CRM_ATPT_CPTD_CD	7375993	7	2	2.711502e-07	0.124481
LAW_CAT_CD	7375993	0	3	4.067249e-07	1.373922
BORO_NM	7375993	11329	5	6.789176e-07	2.162442
LOC_OF_OCCUR_DESC	7375993	1543800	5	8.573104e-07	1.241785
PREM_TYP_DESC	7375993	40745	74	1.008828e-05	3.579655
JURIS_DESC	7375993	0	25	3.389374e-06	0.668693
JURISDICTION_CODE	7375993	6278	25	3.392261e-06	0.668795
PARKS_NM	7375993	3229345	1206	2.908373e-04	0.110850
HADEVELOPT	7375993	7029181	279	8.044704e-04	7.277703
HOUSING_PSA	7375993	1241432	5103	8.318444e-04	1.216803
X_COORD_CD	7375993	17339	71343	9.695115e-03	14.595751
Y_COORD_CD	7375993	17339	73933	1.004708e-02	14.744968
SUSP_AGE_GROUP	7375993	4795235	111	4.301062e-05	2.113260
SUSP_RACE	7375993	3426694	8	2.025676e-06	2.224526
SUSP_SEX	7375993	3560008	3	7.861666e-07	1.323637
TRANSIT_DISTRICT	7375993	7212494	12	7.339494e-05	3.437520
Latitude	7375993	17339	205539	2.793160e-02	15.828371
Longitude	7375993	17339	201311	2.735704e-02	15.797580
Lat_Lon	7375993	17339	198552	2.698211e-02	15.677861
PATROL_BORO	7375993	6735	8	1.085591e-06	2.898560
STATION_NAME	7375993	7212494	372	2.275243e-03	7.617040
VIC_AGE_GROUP	7375993	1638445	202	3.520668e-05	2.201239
VIC_RACE	7375993	309	8	1.084645e-06	2.323077
VIC_SEX	7375993	308	5	6.779031e-07	1.846472

Figure 2: Profiling statistics results

6 DATA CLEANING

Data cleansing or data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a recordset, table, or database and refers to identifying incomplete, incorrect, inaccurate, or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.

The 'NYPD Complaint Data (Historic)' dataset was fairly clean with just a few anomalies. As mentioned in the footnotes provided with the data set, it was pre-processed to a certain degree to ensure removal of inconsistencies and allow for a greater degree of accuracy as was possible. Having said that, there were still errors in terms of null values and date values that fell out of the January 2006 - December 2020 range.

• Dropping irrelevant or redundant columns:
There were many columns in the dataset that conveyed the same data like `X_COORD_CD`, `Y_COORD_CD`, `Latitude`, and `Longitude`. These were redundant and hence, we drop those columns and keep only the most relevant column (eg: `Lat_Lon` column) in our cleaned dataset.

These kinds of redundant columns were present in other datasets we examined as well. Thus, after manual inspection columns conveying the same information could be dropped.

• Removing all the duplicate entries:
Next, we check for entirely duplicate rows and drop them. In almost all datasets, we did not find any duplicate rows.

• Unique ID column:
There is always a key identifier column that identifies a single row in the dataset uniquely. In our chosen datasets, it was not the case all the time. There were rows that had the same unique identifier when we checked for the count of distinct identifiers. In the datasets we inspected, even though the unique identifier values were duplicated, the rest of the values were un-identical and unique.
Thus, we found that in some datasets these column values were entirely different, and in a few datasets, these column values were almost identical. Hence, we were in a dilemma whether to drop such rows or not and

finally decided not to drop such columns since we did not want to lose important information.

• Date Columns:
There were 2 things to handle in date columns. One was handling the null values and the other was checking the range of the dates. There were some datasets where date columns could be 'null'. But in some cases, we needed the date columns to not be 'null'. So we dropped the rows having 'null' dates in such scenarios. And in others, we decided to keep the 'null' values. For example, in the NYPD Complaint Data Historic dataset, we need to remove 'null' values in the CMPLNT_FR_DT column but not in the CMPLNT_TO_DT column. For checking the range, the historic datasets were from 2006 to the end of 2020 (last year) and the current year-to-date datasets were, as the name suggests, from January 2021 to Sep 30, 2021 as the datasets are updated quarterly. So we filtered the dataset using the appropriate date range.

• Specific Columns:
Columns like LOC_OF_OCCUR_DESC, LAW_CAT_CD, CRM_ATPT_CPTD_CD were fairly consistent and were present in a few of the datasets. They had a fixed set of values like LAW_CAT_CD will only have one of FELONY, MISDEMEANOR & VIOLATION as its values. Similar is the case for the other mentioned columns. Hence, specific cleaning was not required for such columns.

• Borough Name:
There are five boroughs in New York City: Manhattan, Brooklyn, Queens, Staten Island, and Bronx. And surprisingly, all the datasets had consistent data. We expected spelling mistakes to be there but there weren't any. Some datasets had 'null' values which we dropped.

However, for making a more generalized function in case of misspellings, we used openclean library's String Matcher class. This class helps detect anomalous values. For this it uses:

- Fuzzy Matching: The implementation of fuzzy string matching in the openclean library uses n-gram and levenshtein/cosine distance similarity

Thus, we gave a vocabulary that contains all the boroughs with correct spellings. A matching algorithm is then deployed that identifies dataset values that are misspelled. We created a map that contains all the misspelled words that match with at least one of the values from the vocabulary along with a similarity score.

After manually investigating we then fixed the misspelled words with the word from vocabulary with the maximum score.

• Key Code:
KY_CD is a three-digit offense classification code. So, its valid values are 100 to 999. We filter only the rows which have their key codes in the given range. Other invalid rows are dropped since it is clearly mentioned that the column value is supposed to be three-digit.

• Age Group Columns:
NYPD Complaint Data Historic included columns like SUSP_AGE_GROUP and VIC_AGE_GROUP. There were many inconsistencies in the Age Group columns in all the datasets. There were unusually high values, negative values, etc. The valid groups are: '<18', '18-24', '25-44', '45-64', '65+' and 'NaN'. And, we replaced all the invalid values with 'NaN'.

This script was replicated in all the other similar datasets in question and correctly reproduced the results.

• Race Columns:
NYPD Complaint Data Historic included columns like SUSP_RACE and VIC_RACE. There were no invalid values in the race columns but there were unknown and null values. Keeping consistency in mind, we replaced all unknown and null values with NaN.

• Sex Columns:
NYPD Complaint Data Historic included columns like SUSP_SEX and VIC_SEX. There were no invalid values in the sex columns. There was an observation we made while checking for the count of each unique sex. The number of unknown values in SUSP_SEX was very high which is expected since it is a suspect. On the other hand, the same number was low for the VIC_SEX column because that is expected to know.

• Offense Description Column:
In the NYPD Complaint Data Historic dataset, we found that there was a one-to-one mapping between KEY_CD and OFNS_DESC columns. Each key code represents a particular offense description. So we can use key code for future analysis instead of offense description.

Every key code corresponds to an offense description. However, a deeper look into the dataset revealed inconsistencies. For example, key code 124 corresponds to 'Kidnapping' as well as 'Kidnapping and other offenses'. To bring about consistency in the dataset, we identified the functional dependencies between key code and offense description column. Now, using open clean we identify the violations in the functional dependencies. We then use a repair strategy to replace the violation value with the maximum frequency value of the group and resolve the conflict.

In other datasets that had offense description columns there was a corresponding code. using the script we realized that one-to-one mapping is consistent.

6.1 Challenges involved in cleaning data

1. NYPD Complaint Data Consists of some columns like X_CORD, Y_CORD, Latitude, and Longitude which at the end convey the same information which a georeferenced column would provide. So dropping these columns would make the dataset clean from unnecessary columns which at the end as a whole would provide a single piece of information. Hence we decided to tackle this problem by only keeping the new georeferenced column in the cleaned dataset
2. Complaint number is an inherently unique identifier in the dataset in which some rows identify each and every complaint. After analyzing the Dataset we found that there are instances where the complaint numbers are not unique to certain rows. These duplicate complaint numbers still reference unique rows which contain useful information so a better approach of not losing important information was to not directly drop these duplicate data rows.
3. The date format columns in the Dataset like `CMPLNT_FR_DT`,`CMPLNT_FR_TM`,`CMPLNT_TO_TM`, `CMPLNT_TO_DT` have null values in them and all these columns could have null values in any combination for a particular row. The challenge here was to strategize which rows to keep and decide if missing Time and Date Values would still point to meaningful data or not.
4. Borough information is a very important column from the point of view of the New York City Dataset. Some rows in the dataset had null values in this column so deciding on dropping these rows was a challenge as rows consisted of all other fields having valid and important data
   One of the other challenges with the Borough name column was, in one of the datasets there were 6 unique values instead of 5 (one of them being New York). We propose using the precinct code information or latitude and longitude information to determine which Borough it is.
   Borough name was also represented in different datasets with different labels.
5. Some columns like SUSP_AGE_GROUP and VIC_AGE_GROUP had outliers like negative values and unrealistic high-value age groups so deciding on the strategy to clean these data rows by keeping the other useful information intact was a challenge
6. A key code uniquely identifies an offense description and there is enforcement of having a one-to-one mapping for these two fields so using key code for further analysis would challenge the existence of having the offense description column.
7. One of the challenges we faced while resolving the functional dependency violations in offense description corresponding to a particular key code, was while taking a voting strategy there was a tie.
   We introduced a tiebreaker to resolve this issue.

7 STEPS TO REPRODUCE

This section explains in detail the steps needed to be taken to run the pyspark programs present in the google colab notebook on a given dataset

1. Download the dataset to be cleaned and upload it to your google drive
2. Once the upload is complete, keep the path to the dataset in google drive handy and it will be required to read the dataset using pyspark
   a. The dataset uploaded should have a valid path similar to this example: gdrive/MyDrive/NYPD_DATASET.csv
3. Run the first cell in the google colab notebook which will mount your google drive in the notebook environment
4. Installation and Importing of Required Libraries and Packages have to be done next before running the Pyspark Programs
5. Installation of Dependencies include Apache Spark 2.3.2 with Hadoop 2.7, Java 8, and Findspark to locate the spark in the system

   a. NOTE: It is better to avoid Spark 2.4.0 version since some people have already complained about its compatibility issue with python.
6. We have also used OpenClean and Humanfriendly package for statistical inference and data profiling so these two libraries also are needed to be installed using pip package manager inside the google collab notebook
7. Following the google notebook provided step by step will make sure that all libraries and dependencies are installed in the correct order and correct version
8. After all the dependencies and Libraries are Installed, We need to set environment variables that will enable pyspark to run in the google colab notebook
   a. Two environment variables need to be set
      i. SPARK_HOME
      ii. JAVA_HOME
9. Reading the dataset from google drive as a CSV into pyspark will be the next task. Follow along with the Colab Notebook and ensure you pass the correct Google Drive Path to the spark.read.csv() function
10. Follow the boxes in the google colab notebook in the specific order mentioned in the notebook to successfully clean the data.
    a. NOTE: Some Datasets may contain different Column Names for Similar Data in the Datasets
       eg. Borough columns in different datasets may have the Column name as BORO or BOROUGH_NAME. Please ensure that you provide the correct column name in the pyspark programs to reproduce appropriate data cleaning techniques mentioned in the google colab notebook
11. We have replicated our cleaning Strategies and cleaning steps on various other notebooks. Below is the link to the excel sheet which contains the links to the google colab notebooks
    a. Link to Replicated Google Colab Notebooks shorturl.at/bvOPQ
12. At the end of the google colab notebook in the last cell, you will be able to export the cleaned final dataset into a CSV file

8 FUTURE SCOPE

• Strategies to run our script on all NYC Open Datasets:
Data Cleaning depends on the type of data we have. So first of all, we can find the distinct data types that a dataset has. And define a basic first-level cleaning strategy for a data type. Like for date, the format can be validated. If the dataset has a specific timeline, filter the date column accordingly. Similar strategies can be formulated for other columns. Functions can be defined that have techniques for validating and cleaning a particular column. Then, whenever a new dataset has to be cleaned, and it has a similar column, we just call that previously defined function, and our job should be done. This can help us by reusing the same strategy for a large number of datasets.

• Strategy for finding similarity between columns using Name Labels:
We observed that some columns in different datasets convey the same information but have different column names. eg Dataset 1 might use BOROUGH_NAME as the column name and the second dataset could use only BORO as the column name and both these columns give borough name as the information. To tackle this issue we can create a mapper function that can map similar column names to a single column label and this column label can be used interchangeably among all datasets.

• Generalizing Cleaning for Borough Name:
In all the datasets, the Borough Name column was consistent but in one of the datasets 'NYPD Criminal Court Summons Incident Level Data (Year To Date)', we found that there was one additional invalid value, 'New York'. We cannot filter out such entries, instead, we have thought of a future enhancement for this. We have different columns like Precinct code, Latitude, Longitude, etc. A set of precinct codes belong to a particular borough. So when we do not have a valid Borough Name, we can check the precinct code and replace it with the appropriate borough. Another approach is to get the latitude and longitude boundaries and check which borough's boundaries the current entry's data belongs to and replace the invalid borough name with the correct one.

9 GITHUB REPOSITORY

All the datasets, python files(codes), Jupyter, and Google Colab notebooks used to profile and clean the different datasets can be found at the link: shorturl.at/suILO

10 CONCLUSION

Strategizing and choosing effective and generic data cleaning methods for the datasets was a major challenge in the project due to the fact of the diverse and varied data columns that each dataset possesses. Our Data cleaning methods work effectively on similar datasets and we have presented our findings in this report. We have replicated our programs on different google collab notebooks which have different data sets and the programs written in spark work effectively on every dataset and do the task of effectively cleaning and sanitizing data wherever possible without losing crucial data points. We have considered all the columns and their interdependence on each other to develop the strategy to clean columns in a given dataset. The resultant cleaned data can be effectively used for analyzing and visualizing the dataset in the future and also major outliers within each column are taken care of by our programs and their logical approach. The approach we have presented in this paper is scalable and can be used as a reference to clean datasets that fall into different categories. This report has detailed analysis and process that went under to effectively and efficiently clean data in the datasets mentioned in this paper

11 REFERENCES

[1] https://openclean.readthedocs.io/
[2] https://github.com/VIDA-NYU/auctus
[3] https://opendata.cityofnewyork.us/
[4] https://dev.socrata.com/data/
[5]https://towardsdatascience.com/analysis-of-nyc-reported-crime-data-using-pandas-821753cd7e22