Data Cleaning: Historical DOB Permit Issuance Housing & Development

NAMAN KAPOOR, New York University, USA
PIYUSH DOKE, New York University, USA
RAHUL BARHATE, New York University, USA

This project aims to analyze the Historical DOB Permit Issuance Housing & Development dataset, which is publicly made available at NYU OpenData. It puts forward various techniques implemented to clean the given dataset, and scaling methods to generalize these techniques on similar datasets.

1 INTRODUCTION

Data cleansing or data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, 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 [2]. Data is a critical resource that supports business processes and managerial decision making. As data volume increases, so does the complexity of managing it and the risks of poor data quality. This project puts forth various techniques to clean the given dataset, and scaling methods to generalize these techniques on similar datasets.

The Department of Buildings (DOB) issues permits for construction and demolition activities in the City of New York. The construction industry must submit an application to DOB with details of the construction job they would like to complete. The primary types of application, also known as job types, are: New Building, Demolition, and Alterations Type 1, 2, and 3. Each job type can have multiple work types, such as general construction, boiler, elevator, and plumbing. Each work type will receive a separate permit. (The DOB Job Application Filings dataset gives more information about each job application.) Each row/record in this dataset represents the life cycle of one permit for one work type. The dataset is updated daily with new records, and each existing record will be updated as the permit application moves through the approval process to reflect the latest status of the application. The Historical DOB Permit Issuance dataset [3] has been used to identify various data quality issues. The dataset is further explored and information is extracted to learn about the inconsistencies in the data. Different quality issues were identified after analysing the data and state of the art techniques were used to clean and create a new version of the dataset. Cleaning decisions are discussed when there was no clear 'right' approach to be followed.

2 PROBLEM STATEMENT

Fig. 1 represents the data cleaning model on a high level. The uncleaned data has several errors and anomalies. Cleaning strategies are applied to the dataset to get consistent and correct data. The effectiveness of our cleaning strategy will be judged using Recall and Precision.

(1) **Recall**: This is the fraction of relevant instances that were retrieved. It is also known as percentage hits. It is defined as the percentage of duplicate records being correctly identified. Suppose we have 7 records A1, A2, A3,

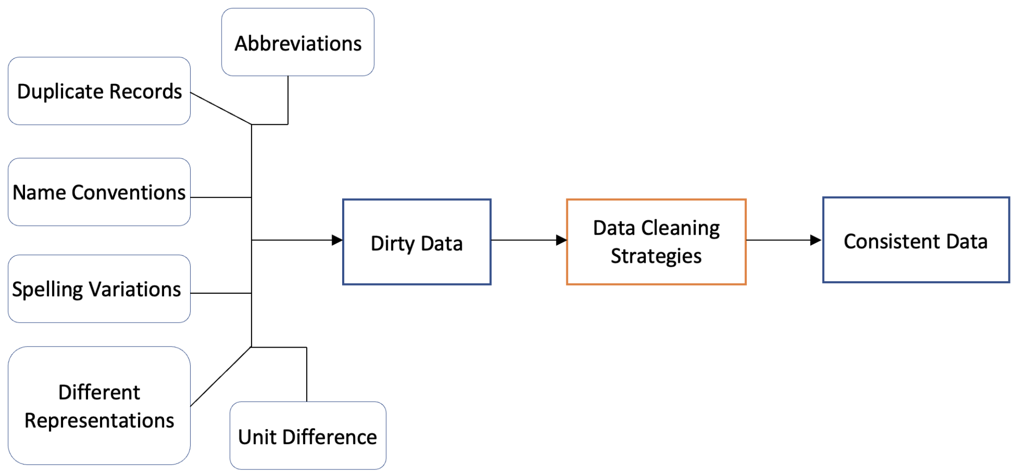

Fig. 1. Data Cleaning Model

B1, B2, B3, C1, with A1, A2, A3 and B1, B2, B3 being different representations of records A and B respectively. A cleaning process which identifies A1, A2, C1 and B1, B2 as duplicates would have a recall of $\frac{4}{6} \times 100\% = 66.7\%$. In our context we have defined it by:

$$Recall = \frac{\textbf{TP}}{\textbf{TP+FN}}$$

(2) **Precision**: It is the fraction of relevant instances among the retrieved instances.

$$Precision = \frac{\textbf{TP}}{\textbf{TP+FP}}$$

TP: Correctly cleaned the required data.
TN: Did not clean the data that did not require cleaning.
FP: Cleaned the data incorrectly.
FN: Did not clean the data that need cleaning.

The Historical DOB Permit Issuance dataset contains 2.43 million rows and contains 60 columns, where each row represents a construction permit between 1989 and 2013. The dataset is full of typographical errors, validation faults, null values, invalid entries, redundant values. Mentioned below are issues present in the data set.

Issues in the given dataset:

(1) ***Syntactical Consistency:*** The data in a single column should be represented in the similar format. In the above dataset the "Number" column representing Street Number has values like "3", "THREE", "3-46" and "3-date".
(2) ***Uniqueness:*** The data set contains multiple rows having exact same data.
(3) ***Completeness/Missing Records:*** Some rows have more than 90% of the columns empty, including cols that are mandatory.
(4) ***Invalid Values:*** Postcode has a value of 0. Phone number less than 10 digits. Work Type has an invalid value. Name field has values like "", "-" or "—-".
(5) ***Violation of attribute dependencies:*** Expiry date is less than issue date.
(6) ***Distance Based Outliers:*** A value is considered an outlier if at least fraction p of the values in a column lies greater than distance D from the value. "Owner's House #" column has values like "P.O.", "One", "N/A".
(7) ***Issue in representation:*** In column "Owner's House City" same city is represented in different literals. eg, Brooklyn is represented as "Brklyn", "Bkyn", New York is represented as "New York", "NY" or "N.Y.".

The data cleaning methods are based on the following approach.

(1) Profiling the data: Data profiling includes a broad range of methods to efficiently analyse a given dataset. [https://dl.acm.org/doi/abs/10.1145/2590989.2590995] Profiling activities range from ad-hoc approaches, such as eye-balling random subsets of the data or formulating queries, to systematic inference of information and statistics of a dataset using dedicated profiling tools. [https://dl.acm.org/doi/abs/10.1145/3035918.3054772] We create a small summary of the database that can be used to audit our data set.
(2) Further, different types of data quality issues are discovered and corrected, including incorrect values (e.g., typos – brklyn), inconsistency between values (e.g., zipcodes or city names), missing data, outliers
(3) The data is cleaned and a new version of the dataset is created.

The problem statement involved is to come up with a strategy to clean the above database and to generalize the strategy for similar data sets.

2.1 List and Description of Similar Datasets

(1) **DOB NOW: Build Approved Permits:** This dataset contains a list of all approved permits in DOB NOW
(2) **DOB Permit Issuance:** The Department of Buildings (DOB) issues permits for construction and demolition activities in the City of New York. The construction industry must submit an application to DOB with details of the construction job they would like to complete. The dataset is updated daily with new records, and each existing record will be updated as the permit application moves through the approval process to reflect the latest status of the application.
(3) **DOB Job Application Filings:** This dataset contains all job applications submitted through the Borough Offices, through eFiling, or through the HUB, which have a "Latest Action Date" since January 1, 2000.
(4) **DOB Complaints Received:** This dataset contains complaints received by the Department of Buildings (DOB). It includes complaints that come from 311 or that are entered into the system by DOB staff.
(5) **DOB NOW-Build-Job Application Filings:** This dataset contains a list of most job filings filed in DOB NOW excluding electrical, elevator and LAA jobs

(6) **DOB NOW: Build Elevator Permit Applications:** This collection contains applications submitted online via "DOB NOW: BUILD" portal by Design Professionals to get an authorization to begin work on an elevator project.
(7) **DOB After Hour Variance Permits:** This dataset contains a list of all after-hours variances issued in DOB NOW
(8) **Property Data:**
(9) **DOB License Info:** This dataset gives DOB licenses and registrations issued by the Department to individuals working with the Department of Housing and Development and/or within the construction trades in New York City.
(10) **DOB Stalled Construction Sites:** This dataset gives a list of sites across the five boroughs where a complaint was made because of which the construction activity has come to an abrupt halt.

3 RELATED WORK

Pre-processing dirty data prior to data cleaning process leads to consistent data and better de-duplication. One of the most reliable way to detect inexact duplicates is to compare each record with every other record. The Sorted Neighbourhood Method (SNM) reduces this O(N^2) complexity by sorting the database on an application-specific key and making pairwise comparisons of nearby records by sliding a window of fixed size over the sorted database. If the window size is w, then every new record entering the window is matched with the previous w − 1 records. The first record then moves out of the window. Although this method requires wN comparisons, its effectiveness depends heavily on the ability of the chosen key to bring the inexact duplicates close together. The Duplicate Elimination SNM (DE-SNM) improves on the SNM by processing records with exact duplicate keys first. But the drawback of the SNM still persists. The clustering method avoids sorting the database by partitioning the database into independent clusters of approximately duplicate records. However, this will result in many singleton clusters as the proportion of duplicates in a database is typically small.
Multi-pass algorithms assumes that no single key is unique enough to bring all inexact duplicates together and employs the SNM cycle several times, each time using a different key to sort the database to reduce the chances of missed duplicates. These algorithms also compute the transitive closure over the identified duplicates [4, 8], that is, if A is equivalent to B, and B is equivalent to C, then A is also equivalent to C under the assumption of transitivity. In this way, A and C can be detected as duplicates without being in the same window during any of the SNM runs. While this can increase the recall, it is also likely to lower the precision. The Incremental Merge/Purge Procedure use “representative records” to avoid re-processing data when increments of data need to be merged with previously processed data. The main difficulty with this method is the choice of representative records which characterize the database. [8] gives a domain-independent technique to detect approximate duplicate records which is applicable to textual databases.

4 MODEL, ARCHITECTURE AND DESIGN

Pre-processing stage is followed by a processing stage and then by a validation & verification stage.

(1) *Data Auditing and Profiling*. As the dataset is huge in order to analyse it we take a sample from our dataset. Sample size is calculated using [1]. In order to evaluate sample size we have to the confidence interval and confidence level we are going to use. We assume confidence interval to be 10 and confidence level to be 95%. We will use this sample size to profile our data. We have used OpenClean’s "DefaultColumnProfiler", that evaluates

various columns for empty and distinct attributes, uniqueness percentage and entropy.

(2) *Pre-processing*. The profile of the data created is used to calculate percentage of emptiness in columns. The columns that are empty below a given CUTOFF percentage are removed. We chose the CUTOFF percentage as 60%.

(3) *Processing*. Earlier we created strategies that were specific to Historical DOB Permit Issuance Housing, which used hard-coded column values to clean the data. The previous strategy cannot be used now as same data is represented using different column names, though having same cleaning strategy. For example, "BIN", Building Identification Number is represented as "Bin #", "BIN", etc., in different datasets. "Street Number" is represented as "Street", "Street #", "Owner's Street #", etc. We have tried to combine strategies for data such as street number, name, zipcode and phone number, so that columns containing similar information can use same cleaning strategy irrespective of its name representation.

(a) **Method: Data Standardization**
Sometimes, users are faced with situations where the dataset contains certain variations of spellings (with extra/missing punctuations) for example, a Business or a street name. Openclean provides **Token Signature Outliers**. Token Signature Outliers, this functionality helps identify anomalies, when values in the dataset are expected to have a signature token present. For e.g. an address column often requires values to contain street suffixes. The Token Signature class will ensure any values that don't have one are highlighted.
Clean Street Related Columns: We have used StandardizeUSStreetName tokenizer to tokenize the street values.

(b) **Method: Syntactical Consistency and Missing Values**
Null and empty values are taken care of and syntactical flaws in the data is taken care of.
***1. Clean Name/ Business Related Columns*:** We replace null values to "N/A". Strip data for spaces or non alphanumeric characters. All the extra values other than letters like "-" or "/" are removed from the name data.
***2. Clean Phone number Related Columns*:** We replace null values to "N/A". Check if the number is equal to 10 digits, else we impute the data and replace it with "N/A". Country code is stripped in case present.
***3. Clean Zip Related Columns*:** We replace null and 0 values to "N/A". Check if the zip is equal to 5 digits, else we impute the data and replace it with "N/A". Country code is stripped in case present. We split the zip in case "-" is present and take the first half.
***4. Clean House Number Related Columns*:** We replace null and 0 values to "N/A". Check if the house number contains values like "3-date" and remove it. All the values in the form of "3" or "3-56" are kept.
***5. Check for "other" columns*:** Replace empty values to "N/A".

(c) **Method: Validation from External Dataset**
Validating values of columns like street code and city name from external datasets.
Clean State Code Related Columns: We validate the state code values from "nyc.gov:dof:state_codes" dataset, else replace using "N/A".

(d) **Method: Misspellings Fuzzy Matching with Validation from External Dataset**
Implementation of fuzzy string matching using n-gram overlaps and Levenshtein or cosine distance.
Clean City Name Related Columns: We validate the values from "encyclopaedia_britannica:us_cities" dataset with fuzz logic 70%. Resulting in correction of similar spellings like "Brookyln" and "Brookyl".

(e) **Method: Misspellings Soundex**
Similar sounding words are grouped and can be corrected accordingly.
Clean Borough Related Columns: Find entries where the Soundex of the City is the same as the soundex for 'BROOKLYN' but where the city name is not 'BROOKLYN', i.e., potential misspellings. The borough data under consideration does not exhibit this error in any dataset considered.

(f) **Method: Stastical Outliers**
Openclean provides many statistical anomaly detection operators that are implemented by the scikit-learn machine learning library.
Clean NTA Related Columns: We used DBSCAN, Isolation Forests, Local Outlier Factors, One Class SVM, Robust Covariance, then count for each value, the number of operators that classified the value as an outlier.

(g) **Method: Functional Dependency Violations**
FD is a constraint that describes the relationship between sets of attributes in a relation, detect pairs of records that violate FD and Repair FD violations by applying data modifications.
Approach 1:Street Name, NTA –> Borough
For example, FD(BROADWAY, SoHo-TriBeCa-Civic Center-Little Italy)   QUEENS is incorrect. Therefore we correct it by replacing the violation value with the maximum frequency value of the group.

(4) *Validation and and Verification Stage*. To analyze the effectiveness of our cleaning strategy, human intervention is needed to check and look for data cells that are getting cleaned correctly, total cells cleaned and the number of cells that need to be cleaned. This information will help us to calculate precision and recall values over the datasets in consideration.

Table 1. Precision and Recall values for datasets

Dataset Name	Precision	Recall
Historical DOB Issuance Permit	0.9445709947	0.955453149
DOB Job Application Filings	0.9215686275	0.9444072338
DOB NOW: Build – Approved Permits	0.741046832	0.9962962963
DOB Complaints Received	1	1
DOB NOW: Build – Job Application Filings	0.7261613692	0.9082568807
DOB NOW: Build Elevator Permit Applications	1	0.9989023052
DOB After Hour Variance Permits	0.4595397179	0.9364599092
DOB Stalled Construction Sites	1	1
DOB License Info	0.7551724138	0.9954545455

Table 2. Graph Legend

Sr. No	Dataset Name
1	Historical DOB Issuance Permit
2	DOB Job Application Filings
3	DOB NOW: Build – Approved Permits
4	DOB Complaints Received
5	DOB NOW: Build – Job Application Filings
6	DOB NOW: Build Elevator Permit Applications
7	DOB After Hour Variance Permits
8	DOB Stalled Construction Sites
9	DOB License Info

5 RESULTS

Low precision on some datasets is largely due to the data in identifier columns like document number, job number, etc being alpha-numberic in nature. The algorithms are designed to treat and clean all such number like columns as only having numeric data. This results in stripping out any non-numeric characters. This can be improved upon by applying advanced techniques like Shingling, Minhashing and Locality-Sensitive Hashing in the future. Table 1 mentions precision and recall values of all the datasets.

6 REFERENCES

[1] [2] [3] [4]

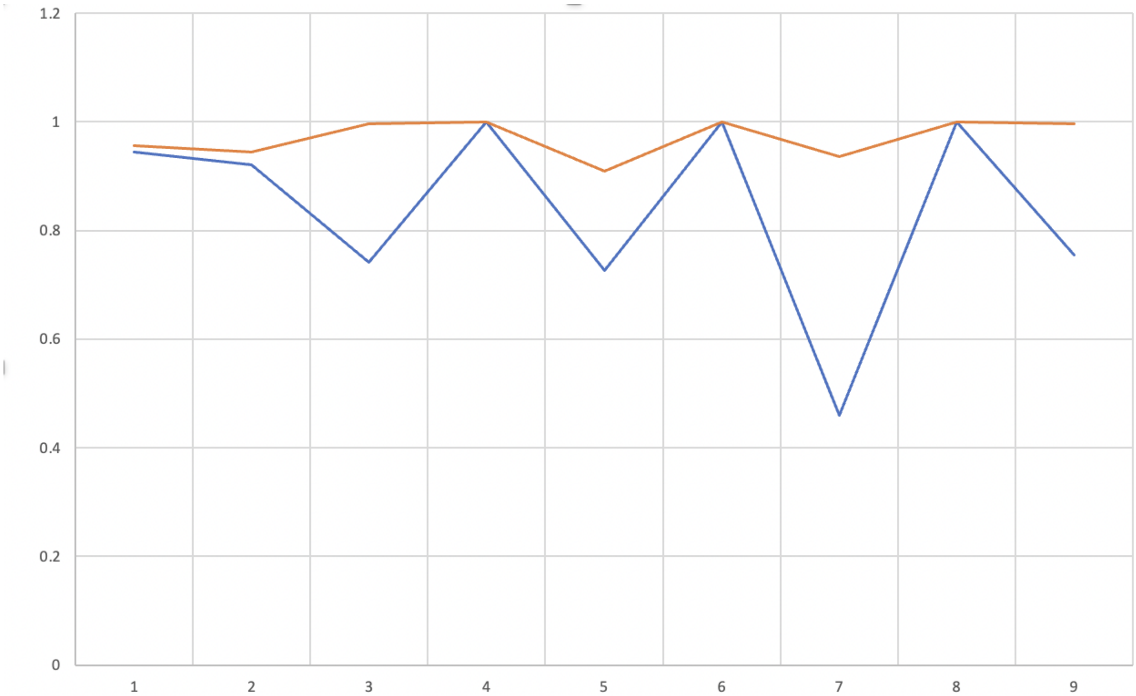

Fig. 2. Precision and Recall values for datasets