CSGY-6513 Project - Group 18

Harrison Hahn - hgh2023
Wei Chen - wc2276
Virend Patil- vap2030

Project Github Repo: https://github.com/HahnNYU/CSGY-6513-Project

I. Introduction

A lot of the time, the data an engineer receives will have quality issues such as misspellings, missing information, invalid formats, etc. In order to improve the accuracy of knowledge gained through analyzing the data, the correctness of the data must be as high as possible. This is where data cleaning comes into play. The goal of data cleaning is to identify and eliminate as many errors and inconsistencies as possible from the data, thus improving the quality of the data [1]. Not only does data cleaning help improve the accuracy of analysis, but it also normalizes the format of data. This makes it easier to integrate data from multiple sources that store the same information but in different formats [1].

Data cleaning is a large topic that has a broad range of use cases. An example of just the broad scope of data profiling, the first step in data cleaning, is shown in Fig. 1 [2]. This Project focuses primarily on using single column profiling and cleaning. Methods are developed to clean a specific dataset. Then we attempt to use those methods on other datasets with overlapping columns and evaluate their effectiveness. The goal is to take the work done to clean one dataset, and improve on those strategies so that the methods are applicable to a much larger scope of datasets with similar data types.

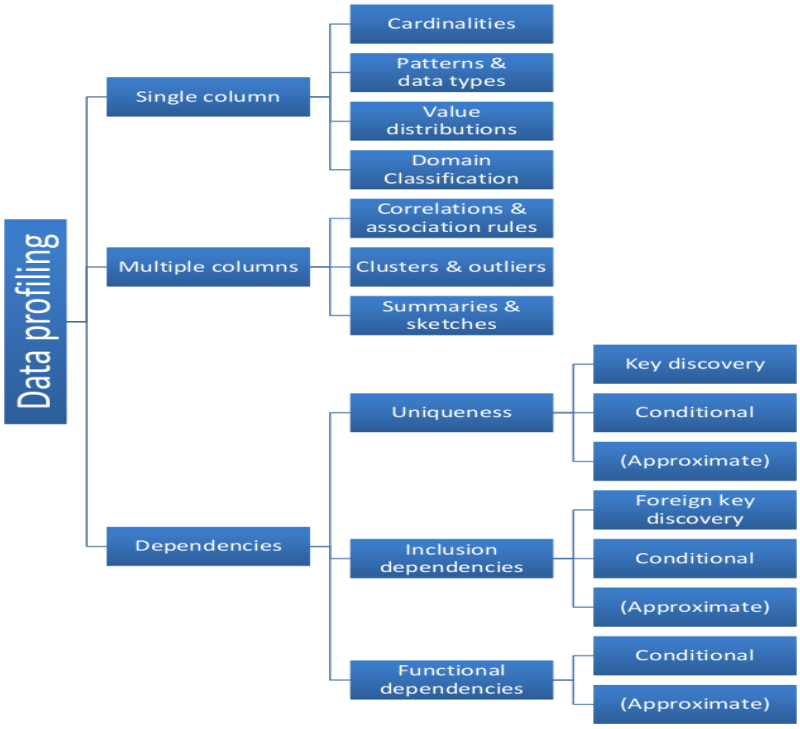


Fig. 1 A Classification of traditional data profiling tasks [2].

II. Problem Formulation

Previously, our group developed methods to clean the Historical DOB Permit Issuance dataset found on NYC Open Data. The first task for this project is to find ten other datasets in NYC Open Data that have columns that overlap with our original dataset. The goal is to then apply the same cleaning techniques on these new datasets and evaluate how they perform. To solve this problem we have to refactor our previous code so that you can easily plug in any dataset with overlapping fields.

To evaluate the effectiveness of our methods on these other datasets, we use the formulas to measure precision and recall shown in Fig. 2 [3]. In our scenario, “true positive” values are inputs that are correctly changed, “false positive” values are inputs that are incorrectly changed, and “false negative” values are inputs that are not changed but should have been.

After measuring the effectiveness of our methods, we must improve and refine our cleaning methods and demonstrate improved effectiveness. Then we have to create reference data for the data types that were cleaned. And finally, we have to outline an approach to run our cleaning techniques on an arbitrary list (potentially all) of the datasets from NYC Open Data.

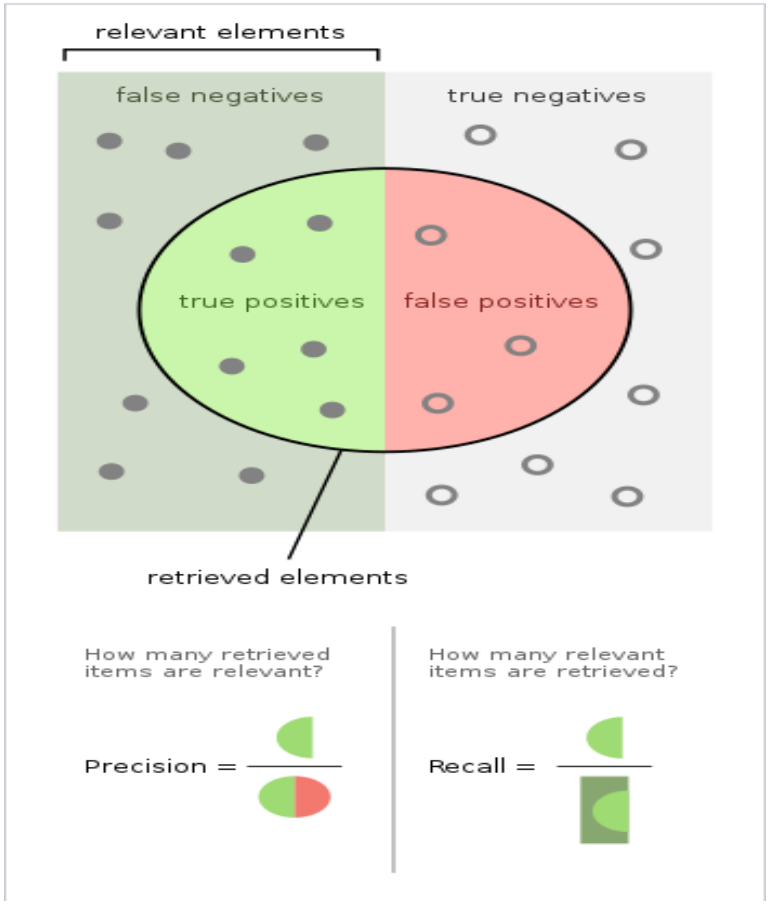


Fig. 2 Formulas for measuring effectiveness of data cleaning [3].

III. Related Work

Data cleaning involves several steps that flow into each other. These “phases” are data analysis (data profiling), formulation of a transformation workflow, verification of transformation workflow, application of transformation to data, and finally backflow of the cleaned data [1]. Common strategies for data profiling involve analyzing histograms of the data to investigate outliers, discover how missing values are represented, and detect if there is any variance in how the same value is represented in the dataset. After profiling the data, you move on to defining a plan that will transform (clean) the data. This step involves defining methods to correct instances of invalid data, and can also involve defining a schema mapper to translate other sources to be able to reuse the same methods. After a transformation workflow is defined, it is implemented on a sample set of the dataset. This starts the verification stage where correctness and effectiveness of the transformation workflow is evaluated. If the effectiveness is desirable, then the transformation is implemented on the entire dataset. Finally,once the dataset is fully cleaned, the original data source is replaced with the improved, cleaned data.

In this project, we validated the values for those overlapping columns in our ten datasets by performing column property analysis to conclude some significant potential issues Fig. 3 [15], including issues like containing a missing value, misspelling, unwanted or weird string characters before or after certain values, etc [15]. Then, different methods were applied to each of the columns for fixing the missing values, and invalid values, etc. Lastly, the dirty values in the data source were replaced by cleaned values.

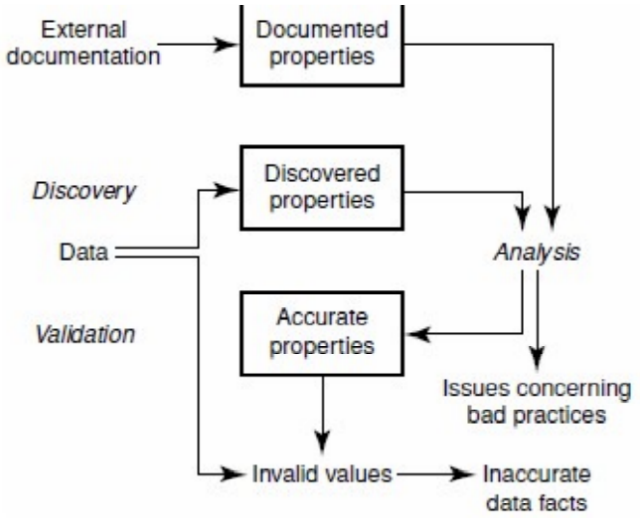


Fig.3 Column Property Analysis [15]

IV. Methods, Architecture, and Design

We originally developed methods to clean all columns in our original dataset, [4]. For this Project, we found ten new datasets in NYC Open Data [5, 6, 7, 8, 9, 10, 11, 12, 13, 14] with

fields that overlap with our original dataset. The fields from the original dataset whose data types overlap with fields in the new datasets are NYC Borough, U.S. City, U.S. Street name, House/Building number, Block and Lot (values assigned by Department of Finance for the building), State, Community Board, a person's first name, and a person's last name. We refactored the code from our previous assignment so that the only thing you need to define to run our cleaning methods on a new dataset is the NYC Open Data id for that dataset, and an entry in our dictionary "column_name_mapping" for that dataset. Each dataset has a different name for the same data type, so in order to reuse methods without altering them, we created a mapper, column_name_mapping (found in the first cell of the Jupyter notebook in our GitHub repository). The key is the NYC Open Data id, and the value is another Dictionary containing a mapper for all the overlapping data types that dataset has. For example, dataset [4] for the column with data type U.S. street name has the identifier "Street", while dataset [7] uses identifier "Street Name" for the same column. To allow the reuse of code between datasets, the mapper will have a key "Street" whose value will be the identifying name for the column in that dataset. Since the value of Street and Number are used to clean values for Block and Lot, if there are other fields that are Street names and building numbers that don't relate to the Block and Lot, they are put in lists with keys "Additional Street" and "Additional Number". Comments in our Jupyter notebook outline which keys have string values and which have list values.

For NYC Borough data, some datasets represented the value as a number from 1 to 5, and others had the name of the borough in either upper or lower case letters. To normalize this across all datasets, our clean_borough_data method converts all those into upper-case representation of the NYC borough. Empty values or values that don't map to a value are changed to "N/A"

For U.S. City data, we use the reference data set encyclopaedia_britannica:us_cities from the openclean RefStore. We first apply strip() to the input to eliminate any trailing or leading white space. Then we apply the upper() method to the city input. Then using the Python library fuzzywuzzy, we iterate through the cities in the reference set and use the value with the highest percent fuzzy match with the input value. We then store this mapping in the file city_ref_lookup.json to make future executions of this method quicker. Empty values or values that don't map to a value are changed to "N/A".

For U.S. State data, we use the reference data set nyc.gov:dof:state_codes from the openclean RefStore. If the input value is in the reference data set, then we output that value, else we change it to "N/A".

For U.S. Street data, we utilize the StandardizeUSStreetName method from openclean geo. Then we have some logic that standardizes certain things, like representing street as "ST" or place as "PL", and catches some common typos that were observed while analyzing outliers in the histogram of the original data set. Empty values are simply returned as "N/A".

For building numbers, viewing the histogram revealed that many numbers were padded with a lot of leading zeros. We simply strip all 0's from the left of the number and return it. Empty values return as "N/A".

The Block and Lot data rely on values for the borough, building number, and street name. It attempts to make a get request to f'https://stevemorse.org/vital/nycblocklot.php?borough={borough.title()}&number={number}&street={urllib.parse.quote(street.title())}' to try and get a value for the block and lot if there is not one provided. If the request fails to return a block and lot, then "N/A" is returned.

Community Board is a 3-digit identifier, where the first digit refers to the borough code, and the last 2 digits refer to the community board for the area the building is in. We were unable to find an API to identify the correct community board code for an address, so we are just replacing all empty and invalid data with “N/A”. We can tell if an input is invald if the first digit is not a number between 1 and 5 or the number is larger than 3 digits.

To clean first name data, we took a histogram of dataset [4] and observed some of the weird things the outliers had. We wrote Python code to strip these artifacts (like adding “MR.” title or random dashes and underscores) to make the data more uniform and what you’d expect. This same strategy was used to clean the last name data.

We read in the data using openclean Socrata and stream methods. At the end of the Jupyter notebook we have a cell that uses the column_name_mapper to identify which overlapping fields are present in the current dataset and the name of those fields in that dataset and runs the appropriate cleaning method on that field.

We used the sample size calculator in [16] to determine that for a Confidence Level of 95% and Confidence Interval of 10, we would need to sample 96 rows from a given dataset. So for each dataset, we sampled 96 rows and calculated the precision and recall of each overlapping field that dataset contained. We found that City typically had a precision of 50% and House/Building number had a recall of about 25%. All the other fields had precision recall close to 100%.

The recall with building numbers is low due to the fact that some of the datasets had numbers spelled out. Another reason is that some inputs were heavily padded with extra white space. These were not issues in our original dataset. To fix the extra white space issue, we simply applied strip() method to each input. The cases where numbers were spelled out were limited to number 1 through 10. So to fix this we simply made a dictionary mapping the spelling of each of those numbers to the numeric representation. So if the input is a key in the dictionary, then the method swaps it with the numeric representation.

One cause of the low precision with cleaning City values was that the reference dataset we used was missing a lot of U.S. cities. To fix this, instead of returning “N/A” if a match wasn’t found in the reference dataset, we only return “N/A” for empty values. If a match wasn’t found in the reference dataset then we just return the input but in uppercase letters. We also added missing cities that appeared in the sample sets of all our datasets into the reference dataset. In addition, we added mapping for some common abbreviations like BX and LIC to the full name of the city, BRONX and LONG ISLAND CITY.

V. Results

The changes made to the method that cleans building numbers raised the recall value close to 100% for our datasets. The changes made to the method that cleans U.S. Cities also raised the precision value close to 100% for our datasets. These changes do not make the coverage complete though, and it’s highly likely that new datasets might have a lower precision. Gathering feedback from multiple datasets has definitely helped improve the effectiveness of these cleaning methods.

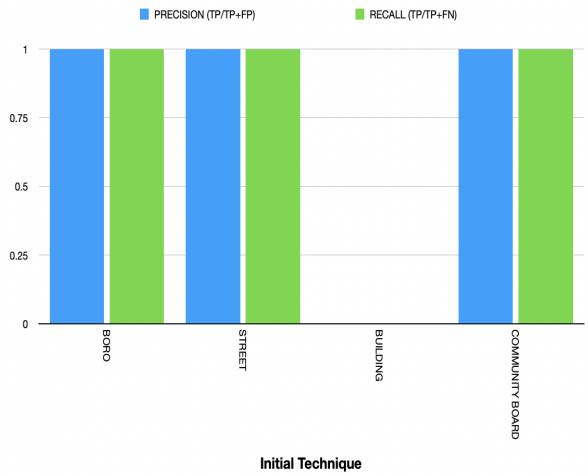
When we say 100% in the effectiveness measurements, that doesn’t represent a true score of 100% for the entire dataset. That is just what was calculated using a sample of that

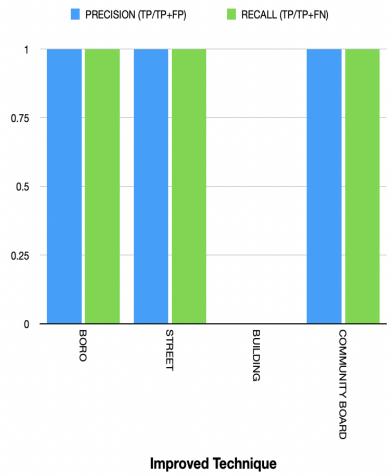
data set. Since we took a sample size to have a Confidence level of 95% and a Confidence interval of 10, we can claim that we are 95% confident that if the sample set has a measurement of 100%, then the entire dataset will have an effectiveness measure from 90% to 100%. For columns with effectiveness measurements of 0, the data was already perfect. There were no transformations needed to fix anything in those columns.

To run our cleaning methods on a list of all NYC Open Data datasets, the only thing we would need to do is create an entry in our column_name_mapping dictionary for each dataset. Then writing a simple loop to iterate through the keys in column_name_mapping and run all the applicable cleaning methods on each key would clean every dataset.To improve the speed of this, you can set up multiple machines, and divide the datasets among those machines to clean all the datasets quicker.

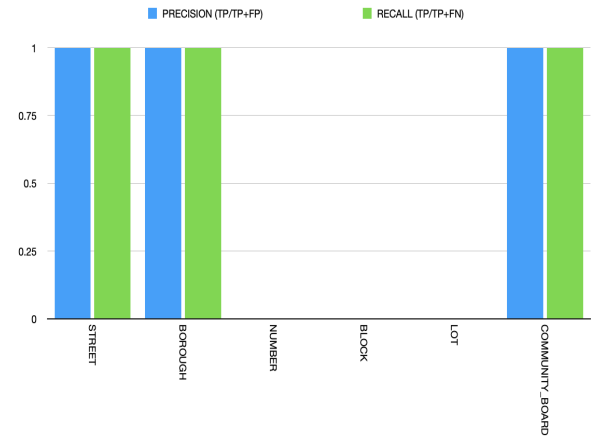
VISUALIZATION OF EFFECTIVENESS MEASURES FOR ALL 10 DATASETS (PRECISION AND RECALL).

1. DOHMH New York City Restaurant Inspection.

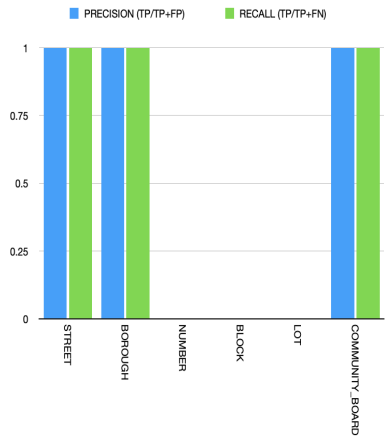




2. DOB Certificate of occupancy.

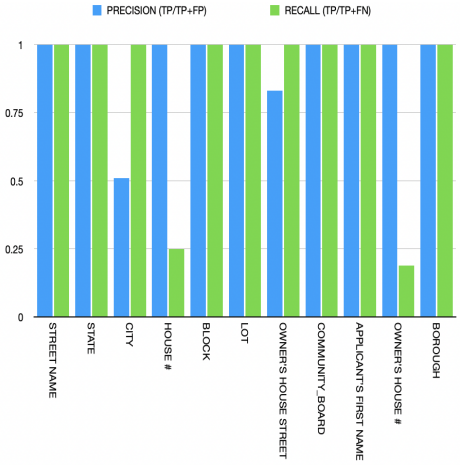


Initial Technique

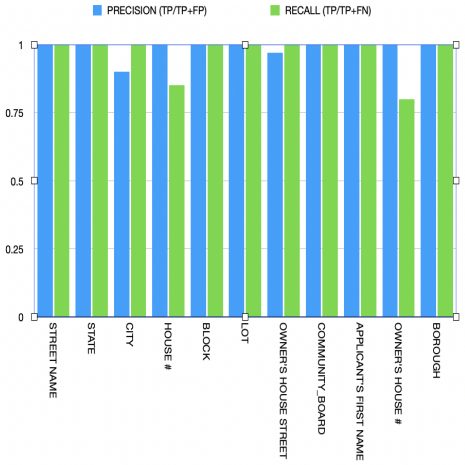


Improved Technique

3. DOB Cellular Antenna Filings.

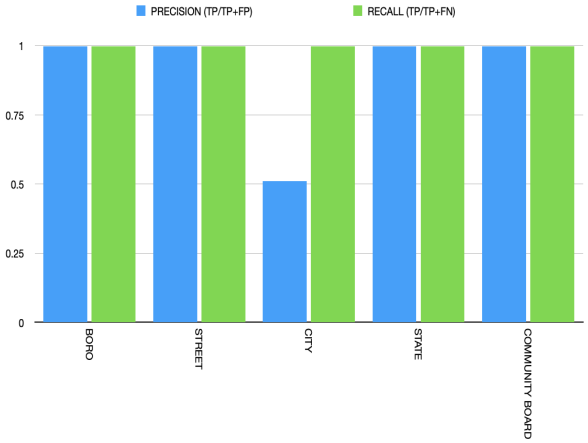


Initial Technique

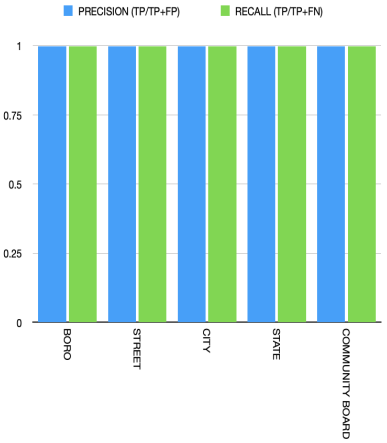


Improved Technique

4. Self Hauler Registrants.

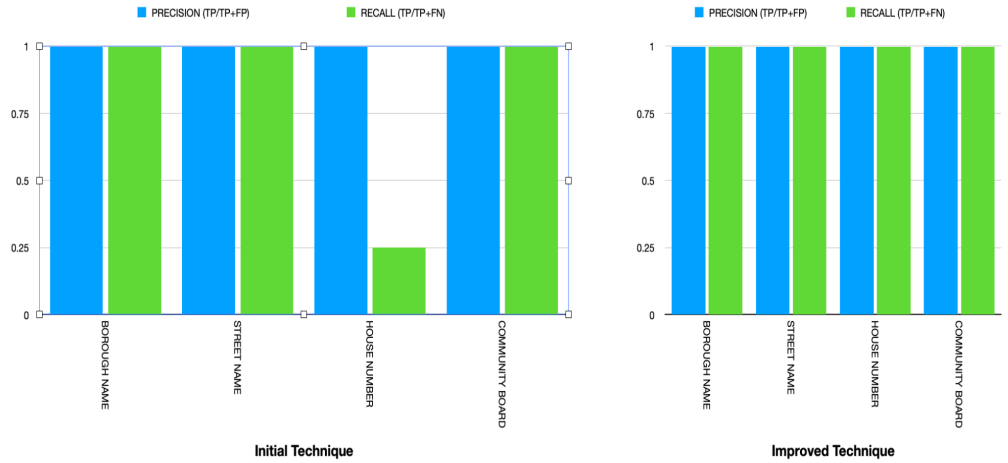


Initial Technique

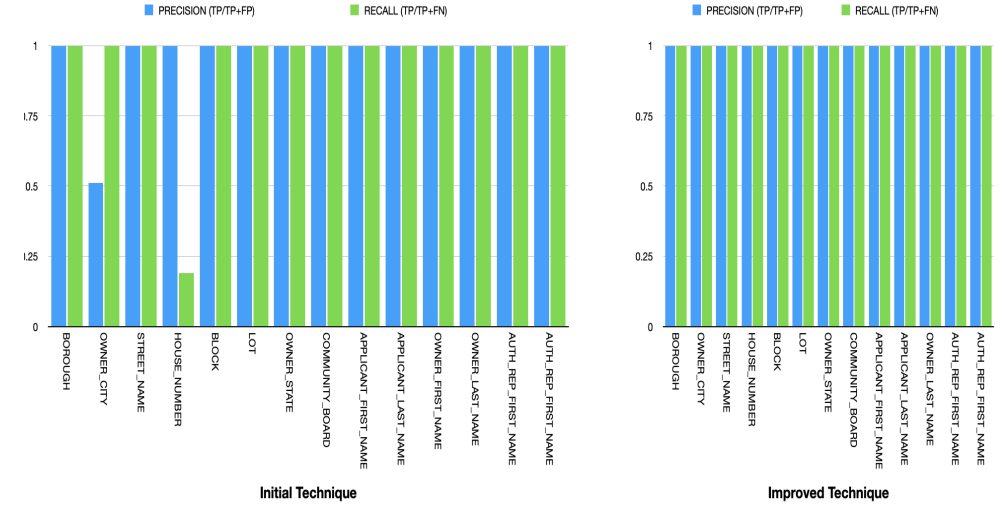


Improved Technique

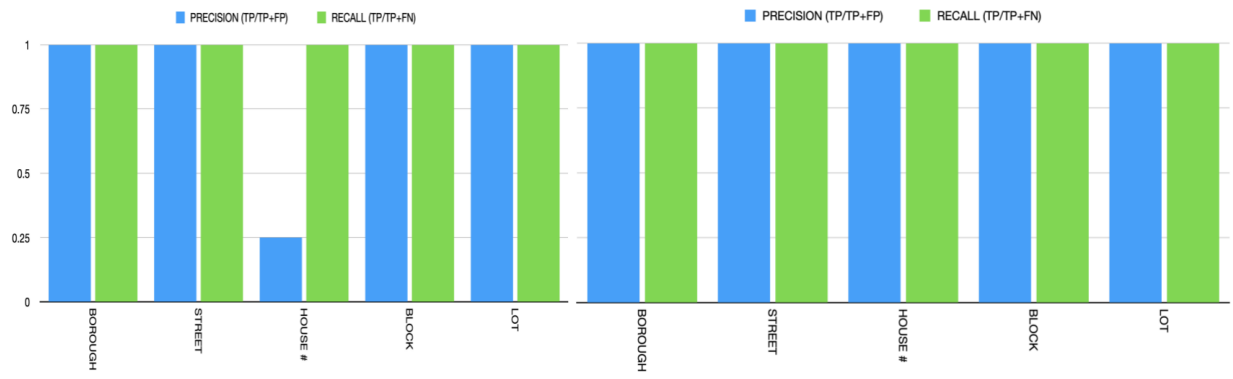
5. DOB Stalled Construction Sites.



6. DOB NOW: Electrical Permit Applications.



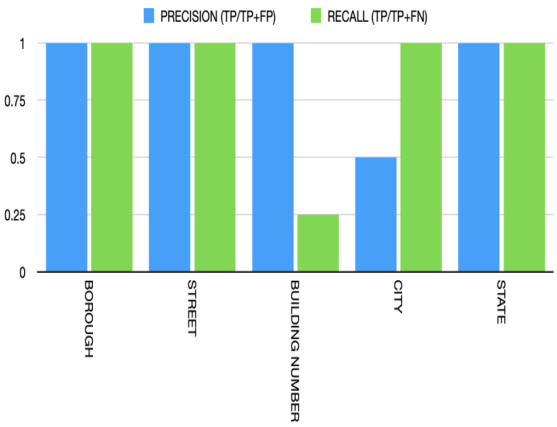
7. Residential Address.



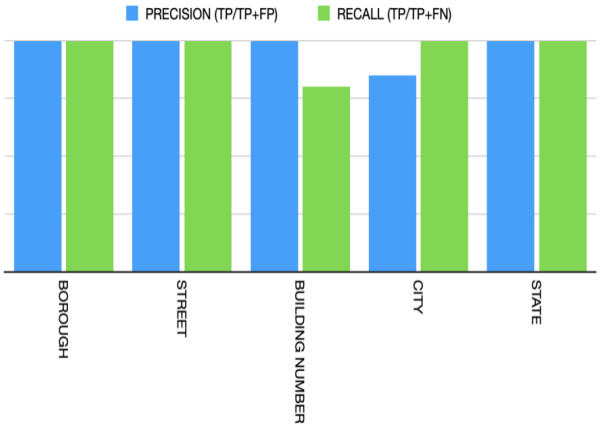
Initial Technique

Improved Technique

8. Charges.

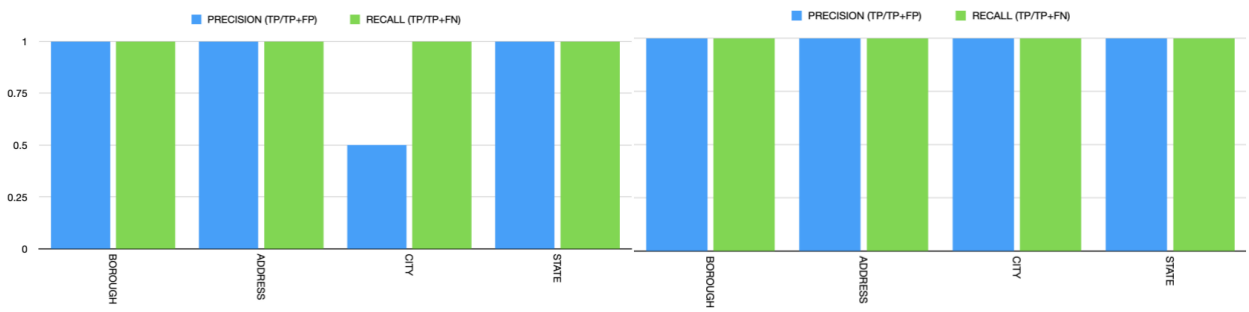


Initial Technique



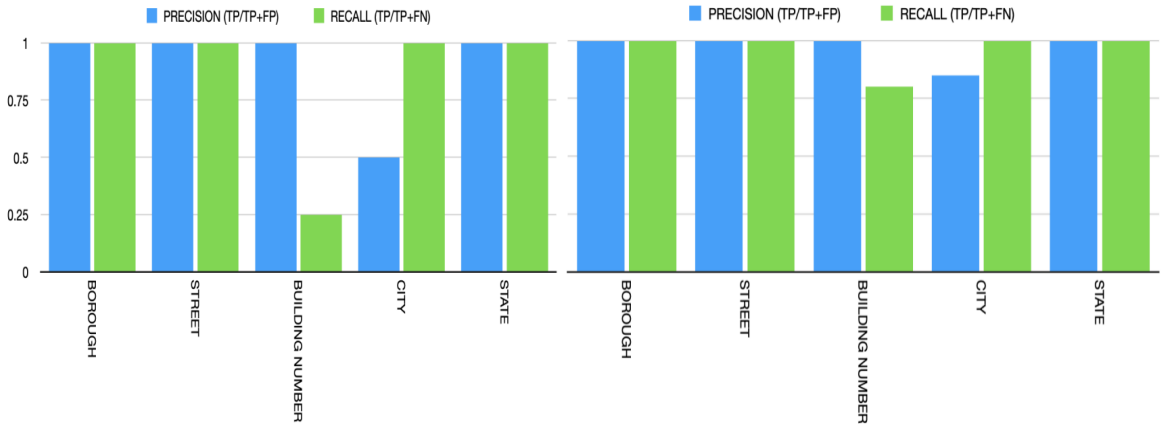
Improved Technique

9. Trade Waste Licenses.



Initial Technique Improved Technique

10. Inspections.



Initial Technique Improved Technique

References

[1] Rahm, Erhard, and Hong Hai Do. “Data Cleaning: Problems and Current Approaches.” *IEEE Data Eng. Bull*, Dec. 2000.

[2] Abedjan, Ziawasch, et al. “Profiling Relational Data: A Survey.” *The VLDB Journal*, vol. 24, no. 4, 2015, pp. 557–581.

[3] “Precision and Recall.” *Wikipedia*, Wikimedia Foundation, 18 Oct. 2021, https://en.wikipedia.org/wiki/Precision_and_recall.

[4] “Historical Dob Permit Issuance: NYC Open Data.” *Historical DOB Permit Issuance | NYC Open Data*, 7 Aug. 2018, https://data.cityofnewyork.us/Housing-Development/Historical-DOB-Permit-Issuance/bty7-2jhb.

[5] “DOB Certificate of Occupancy: NYC Open Data.” *DOB Certificate Of Occupancy | NYC Open Data*, 11 Dec. 2021, https://data.cityofnewyork.us/Housing-Development/DOB-Certificate-Of-Occupancy/bs8b-p36w.

[6] Department of Health and Mental Hygiene (DOHMH). “Dohmh New York City Restaurant Inspection Results: NYC Open Data.” *DOHMH New York City Restaurant Inspection Results | NYC Open Data*, 11 Dec. 2021, https://data.cityofnewyork.us/Health/DOHMH-New-York-City-Restaurant-Inspection-Results/43nn-pn8j.

[7] (DOB), Department of Buildings. “DOB Cellular Antenna Filings: NYC Open Data.” *DOB Cellular Antenna Filings | NYC Open Data*, 11 Dec. 2021, https://data.cityofnewyork.us/Housing-Development/DOB-Cellular-Antenna-Filings/iz2q-9x8d.

[8] (DCA), Department of Consumer Affairs. “Inspections: NYC Open Data.” *Inspections | NYC Open Data*, 10 Dec. 2021, https://data.cityofnewyork.us/Business/Inspections/jzhd-m6uv.

[9] Commission, Business Integrity. “Trade Waste Hauler Licensees: NYC Open Data.” *Trade Waste Hauler Licensees | NYC Open Data*, 10 Dec. 2021, https://data.cityofnewyork.us/City-Government/Trade-Waste-Hauler-Licensees/867j-5pgi.

[10] (DCA), Department of Consumer Affairs. “Charges: NYC Open Data.” *Charges | NYC Open Data*, 10 Dec. 2021, https://data.cityofnewyork.us/Business/Charges/5fn4-dr26.

[11] (NYCHA), NYC Housing Authority. “NYCHA RESIDENTIAL ADDRESSES: NYC Open Data.” *NYCHA Residential Addresses | NYC Open Data*, 15 Feb. 2021, https://data.cityofnewyork.us/Housing-Development/NYCHA-Residential-Addresses/3ub5-4ph8.

[12] (DOB), Department of Buildings. “DOB Stalled Construction Sites: NYC Open Data.” *DOB Stalled Construction Sites | NYC Open Data*, 11 Dec. 2021, https://data.cityofnewyork.us/Housing-Development/DOB-Stalled-Construction-Sites/i296-73x5.

[13] Commission, Business Integrity. “Self Hauler Registrants: NYC Open Data.” *Self Hauler Registrants | NYC Open Data*, 10 Dec. 2021, https://data.cityofnewyork.us/City-Government/Self-Hauler-Registrants/a8wp-rerh.

[14] (DOB), Department of Buildings. “DOB Now: Electrical Permit Applications: NYC Open Data.” *DOB NOW: Electrical Permit Applications | NYC Open Data*, 11 Dec. 2021, https://data.cityofnewyork.us/City-Government/DOB-NOW-Electrical-Permit-Applications/dm9a-ab7w.

[15] T. F. Kusumasari and Fitria, "Data profiling for data quality improvement with OpenRefine," 2016 International Conference on Information Technology Systems and Innovation (ICITSI), 2016, pp. 1-6, doi: 10.1109/ICITSI.2016.7858197.

[16] “Sample Size Calculator.” *Sample Size Calculator - Confidence Level, Confidence Interval, Sample Size, Population Size, Relevant Population - Creative Research Systems*, https://www.surveysystem.com/sscalc.htm.