POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Data and Information Quality Project (Dataset 8)

Author(s): **Alessandro Cavallo 10734387**
Allegra Chiavacci 10733258

Group Number: **6**

Academic Year: 2024-2025

Contents

1 | SETUP CHOICES

In this chapter, the goal of the project and the setup choices are described.

1.1. Goal of the project

The goal of the project consists in performing the complete Data Preparation Pipeline on a dirty dataset. We were assigned a dataset regarding the "Attività commerciali di media e grande distribuzione" of the city of Milan. After the data profiling and the data quality assessment, we performed the data cleaning and then we exported the resulting cleaned dataset.

1.2. Setup Choices

The project is written in Python using Colab as an IDE. Some useful libraries, which were presented during practical lectures, were used during the different phases of the data preparation pipeline.

- Pandas
- Numpy
- Jaro
- Seaborn (heatmap, boxplot)
- matplotlib.pyplot
- skew (from scipy.stats)
- Data profiling phase:
  - ProfileReport (from ydata_profiling)
- Error Detection and correction phase:
  - ensemble (from sklearn) : random forest regressor

  - scripts_for_E5 (scripts given by the professor during classes for encoding categorical variables used for random forest algorithm)
  - NearestNeighbors (from sklearn.neighbors) for outlier detection

- Data Deduplication
  - Recordlinkage (SortedNeighbourhood)

2 | PIPELINE IMPLEMENTATION

2.1. Colab Setup

In this first phase we set up the Google Colab environment used for the project.

1. Import of useful libraries
2. Mount of Google drive account to access the .csv file
3. Definition of Z-Score technique used for outlier detection
4. Definition of functions used to evaluate typos in column 'Insegna' with Jaro Winkler metric (to be used during data wrangling phase)

2.2. Data exploration and profiling

After importing the .csv file using Pandas into Colab, we perform some basic operations to inspect data. We display columns, the number of tuples and columns of the dataset, the type of each attribute, the different properties of numerical attributes. We evaluate correlation based on Pearson coefficient and we can notice that "Superficie totale" is strongly correlated to "Superficie vendita" and "Superficie altri usi" (and viceversa) since the value exceeds the 0.7 threshold. Results are plotted using heatmaps (seaborn library and matplotlib.pyplot). The profile report is created using ProfileReport library (ydata_profiling). Thanks to this report we discover a correlation between "Codice Via" and "ZD". Other problems that we can detect from the report are the presence of 1 duplicate row and the presence of missing values in "Settore Merceologico", "Insegna", "Civico", "Superficie altri usi". The goal is to solve these problems, along with the standardization of attributes, such as "Ubicazione".

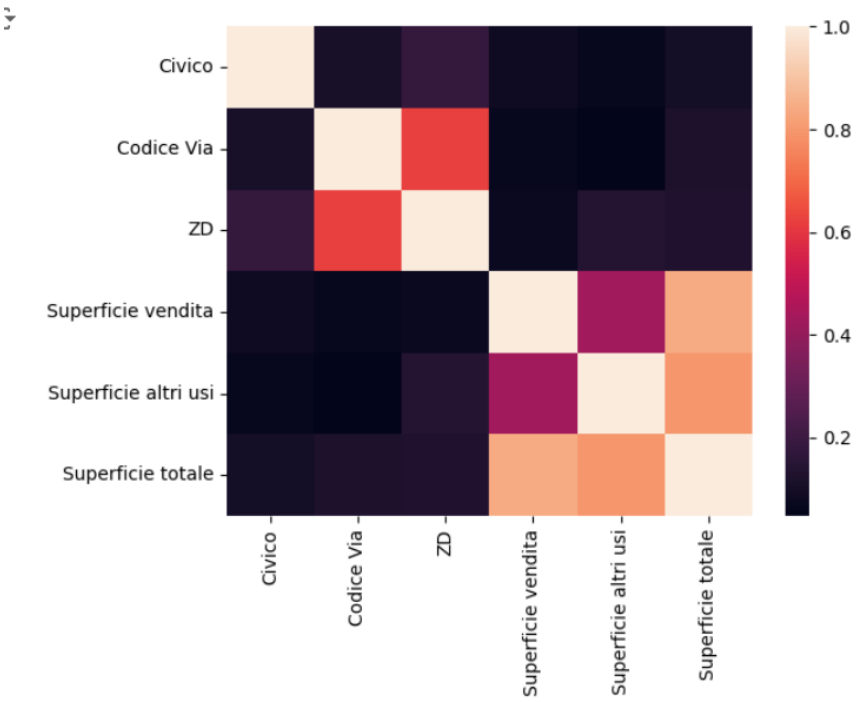

Figure 2.1: Correlation evaluation heatmap

2.3. Data Quality Assessment

In this phase, Data Quality dimensions are evaluated and these metrics will be compared with the ones of the cleaned dataset at the end of the process. Timeliness can't be evaluated since the dataset doesn't contain any timestamp reference. As far as accuracy is concerned, it's possible to evaluate it only for 'ZD' since it's known that the Zones of the city of Milan are values between 1 and 9. We also evaluate the uniqueness, completeness, constancy and distinctness of all attributes. As noticed in the Data Profiling Report, "Insegna" (completeness: 61.58%), "Civico" (completeness: 92.09%), "Superficie altri usi" (completeness: 17.01%) columns have a lot of missing values. Completeness can also be evaluated at table level, showing a value of 88.1% for the whole dataset. In order to evaluate consistency we define a rule stating that the sum of "Superficie vendita" and "Superficie altri usi" should be equal to "Superficie totale" and we can notice that only the 15.80% of the tuples satisfy this rule. We also discover that there are exact matching duplicates which will be dealt with during the duplicate detection phase.

2.4. Data cleaning (Data Tranformation and Standardization)

The first operation we perform is renaming columns removing blank spaces between words such as "Superficie totale" -> "SuperficieTotale" and so on. Then we standardize the values of "TipoVia" according to toponyms, such as "PZA" -> "Piazza" and so on, avoiding abbreviations. As far as values in "SettoreMerceologico" column are concerned, we choose to transform "alimentare;non alimentare" -> "Non specializzato", since it's used without a specific criteria. For example, it's used even for toys shops or clothes shops, which don't sell food. Moreover, we standardize the values of "SettoreMerceologico" avoiding the wording "tabella speciale". All values are in the form "Settore" followed by the corresponding economic sector, such as "farmaceutico", "carburanti", "alimentare" and so on. In addition to that, we try to detect typos in 'Insegna' values using the Jaro-Winkler metrics with a threshold of 0.87. Analyzing results, we manually standardize some of them, such as 'di per di', 'di perdi', 'di x di' becoming all 'di per di', since they are are just misspelled or have useless blank spaces between words. Then we deal with the column "Ubicazione" where several operations are performed in order to extract useful additional information. We are able to extract the "ZD2", "TipoAccesso", "PtoStrada", "Isolato" and "Civico2" attributes. Evaluating the number of null values in each column, we decide to drop the "PtoStrada" column, since only few values are not null. As far as "ZD2" is concerned, it's useless since every tuple already has a "ZD" value and we consequently drop it as well. We can also notice that "Civico" column has some missing values, which may be filled using "Civico2" during the "Missing Values Handling" phase. We decide to drop the remaining content of the "Ubicazione" column as it contains values already captured in other columns or highly specific ones tied to individual tuples, which could not be meaningfully abstracted into a separate column. "SuperficieVendita" and "SuperficieTotale" are cast to float. The content of "Civico2" needs to be standardized, as its format appears to be ambiguous due to multiple meanings of the /, being followed by a number ("22/25", that may indicate an interval of civic numbers, or "22/3", that may indicate the 3th floor of a specific building) or by a letter ("22/a"). We decide to drop everything that follows the /, since we couldn't find a standard convention for the civic number format. "Civico 2" column is then cast to int64, as well as "Isolato" and "Civico" (cast to Int64 to allow also null values). As far as the standardization of values in "TipoAccesso" is concerned, we try to solve the abbreviation of "area ap.rec." with "area aperta recintata", which seems a feasible option, and remove useless blank spaces.

2.5. Data Cleaning (Error detection and correction Missing Values)

Null values in "SettoreMerceologico" and "TipoAccesso" are filled with "Non specificato" while "Insegna" is filled with "Non presente". Null values in "Civico" are filled with the corresponding value present in "Civico2". The missing values in "SuperficieTotale", "SuperficieVendita" and "SuperficieAltriUsi" are filled according to the following rule: "SuperficieVendita" + "SuperficieAltriUsi" = "SuperficieTotale". As far as "Isolato" is concerned, the imputation of missing values is performed using Random Forest algorithm. The concept of "Isolato" is not extremely clear in the dataset since tuples with the same street and civic number may have different corresponding "Isolato" values. Consequently, not even "CodiceVia", "Via" and "Civico" appears to be enough to define the correct value of "Isolato". Anyway, we choose to handle missing values for this column with a Machine Learning based imputation technique (Random Forest) based on those 3 parameters. The main reason why we choose this algorithm is that a rule-based approach (assign the most frequent block for a given street) may fail for long streets with mixed blocks. Anyway, Random Forest may fail due to inconsistencies in the original dataset (in "Isolato" column) since the model may learn incorrect patterns. "Civico", "CodiceVia" and "ZD" are used as parameters, since a shop is likely to have the same isolato of another shop with the same CodiceVia and so on. Since the resulting values may be float, they are rounded and then cast to int64.

2.6. Data Cleaning (Error detection and correction Outliers)

First of all we solve inconsistencies related to the "SuperficieTotale" according to the already stated consistency rule ("SuperficieVendita" + "SuperficieAltriUsi" = "SuperficieTotale"). The outlier detection phase is critical. We try to detect outliers in "SuperficieVendita", "SuperficieAltriUsi", "SuperficieTotale" and "Civico" using the Z-Score method and the Box Plots to have a visual representation. In order to consider the three features "SuperficieVendita", "SuperficieAltriUsi", "SuperficieTotale" simultaneously, we also decide to use KNN method. KNN doesn't rely on normality assumptions, unlike Z-Score, and it may be useful for the outlier detection in this case, since the values of shop floor areas are highly skewed (skewness -> "SuperficieVendita" = 8.96, "SuperficieAltriUsi" = 9.05, "SuperficieTotale" = 5.30). Points detected as outliers at the same time by both methods are likely to be real outliers. Anyway we decide not to drop them,

since it would exclude all shops with a large floor area (such as shopping mall or supermarkets) and shops that are located on long streets with high civic numbers.

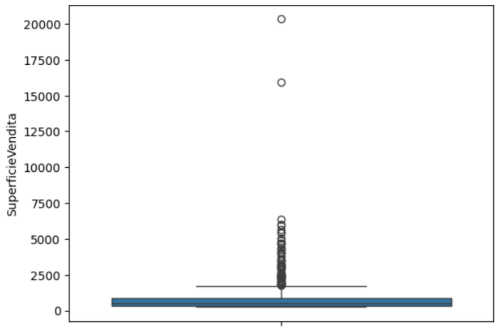

Figure 2.2: "SuperficieVendita" Boxplot

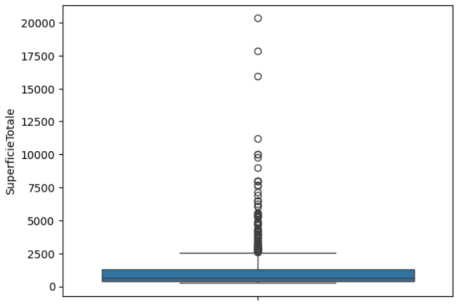

Figure 2.3: "SuperficieTotale" Boxplot

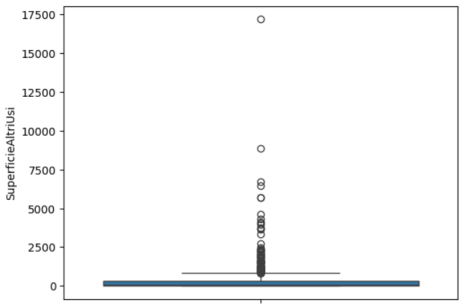

Figure 2.4: "SuperficieAltriUsi" Boxplot

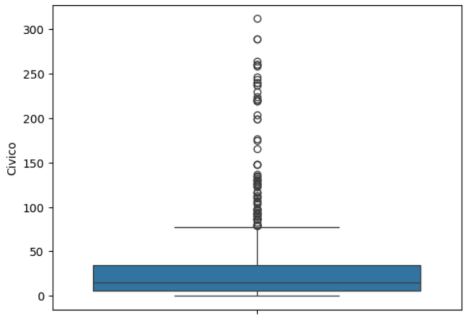

Figure 2.5: "Civico" Boxplot

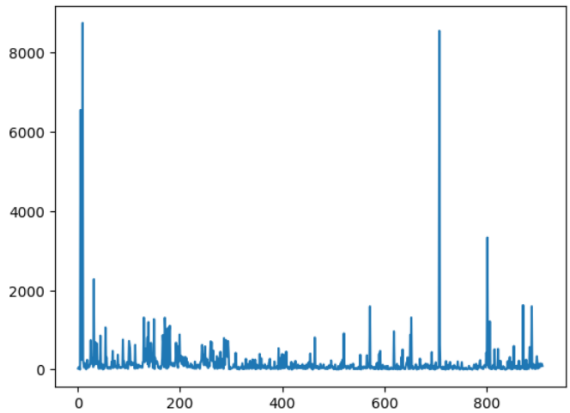

Figure 2.6: KNN Plot (mean of k-distances)

2.7. Data Cleaning - Data Deduplication

First of all, we drop exact matching duplicates. Then, we perform 2 runs of record linkage in order to find non-exact duplicates. We find the candidate pairs with Sorted Neighbourhood on the basis of "CodiceVia", defining a window of size = 9. We choose to sort according to "CodiceVia", since it ensures that shops in the same street are grouped together. It also minimizes the possibility of having typos, since the code is numeric. During the first record linkage, we define 12 rules, including checks on all attributes. We choose to compare 'Insegna', 'Via', 'SettoreMerceologico' and 'TipoAccesso' using Jaro-Winkler similarity measure with a 0.9 threshold, while other attributes have to match exactly, in order for the tuples to be considered duplicates. We choose to perform a second run of record linkage, using the same key "CodiceVia", but relaxing conditions according to the shop floor area. A shop owner may have increased or decreased the shop floor area over time due to renovations or expansions. Consequently, the dataset may include two different entries referring to the same shop, one with the old and one with the new shop floor area details. We are able to detect a duplicate during the first record linkage and 8 during the second one. All the duplicates are dropped, keeping the first element of the pair.

2.8. Data Quality Assessment

At the end we evaluate the Data Quality metrics on the cleaned dataset. We can notice that the completeness is 1.00 for each column and there are no inconsistencies, as expected. The attribute "TipoVia" has a low uniqueness and distinctness, but it is understandable since it contains a small set of repeated values (i.e. "Via", "Piazza", "Corso" and so on). The same happens for columns "SettoreMerceologico" and "ZD". We can notice that the distinctness of "Insegna" has decreased from 0.65 to 0.387. This is likely to occur due to the various operations performed on the column (i.e. missing values are substituted with "Non specificato" and some values are standardized, such as 'di per di', 'di perdi', 'di x di' have become 'di per di'). The value of uniqueness (for each attribute) coincides with the value of distinctness because we have filled all the missing values. Consequently, the number of non-null values corresponds to the number of rows.