# Clinical

**No. of Observations:** 284717087

**Features:**

|  |  |
| --- | --- |
| PERSONID | Unique ID for Patient |
| ENCNTRID |  |
| ORDERID |  |
| CLINICALEVENTID |  |
| EVENTDATETIME |  |
| EVENTNAME |  |
| EVENTRESULT |  |
| RESULTUNIT |  |
| TASKASSAY |  |
| POMPE | Target: True if Patient has IOPD. |

**High-Level Goal:** Our objective is to structure the data such that each patient (PATIENTID) is uniquely represented by a single row, with distinct events (EVENTNAME) delineated as individual columns. For each patient-event combination, we capture the corresponding result (EVENTRESULT) from the latest entry (EVENTDATETIME) as the value within the column.

**Preprocessing Steps:**

1. Filter patients by unique patient IDs (this refers to patients available in the demographic table).
2. Encode POMPE variable: YES to 1, NO to 0.
3. Perform data type conversion: POMPE to int8, EVENTRESULT to str.
4. Drop columns: ORDERID, CLINICALEVENTID, TASKASSAY.
5. Exclude entry where EVENTDATETIME equals 4557.
6. Remove leading white spaces: EVENTDATETIME.
7. Convert EVENTDATETIME to datetime type.
8. Sort the dataframe by EVENTDATETIME in descending order.
9. Replace “nan”, “None”, None with np.nan to represent missing values.
10. Remove rows where EVENTRESULT is null or empty.
11. Strip any leading or trailing white spaces in EVENTRESULT.
12. Filter out entries in EVENTRESULT that are not numeric.
13. Remove trailing periods in EVENTNAME.
14. Group the data by PERSONID and EVENTNAME, keeping the first occurrence of each unique combination.
15. Pivot the grouped data, setting PERSONID as the index, EVENTNAME as columns, and EVENTRESULT as values.
16. Merge the pivoted dataframe with the unique PERSONID and POMPE values from the original dataframe.

A screenshot of a computer

Description automatically generated

Figure 1. Original Data

A close-up of a number

Description automatically generated

Figure 2. Preprocessed Data

**Post Preprocessing:**

**No. of Observations:** 76985

**No. of Features:** 3264

# Demographic

**No. of Observations:** 94521

**Features:**

|  |  |
| --- | --- |
| PERSONID |  |
| EPI |  |
| GENDER |  |
| NATIONALITY |  |
| DOB |  |
| DOE |  |
| POMPE |  |

**High-Level Goal:** Our aim is to streamline and enhance the demographic data. For each individual, identified by their unique PERSONID, we intend to represent accurate, deduplicated information regarding their gender, age, nationality, and status on specific attributes like POMPE and mortality (DEATH). This process will involve refining date values, calculating age based on a reference date, updating categorical encodings, and deriving geographical insights from the nationality data. Ultimately, we seek to transform the dataset such that it is both richer in content and better structured for downstream analytics or modeling.

**Preprocessing Steps:**

1. Date Conversion
   * Strip any white spaces from the DOB (Date of Birth) column and convert its format to a datetime object with the format %d/%b/%Y.
   * Similarly, strip any white spaces from the DOE (Date of Event/Death, I presume) column and convert its format to a datetime object with the format %d/%b/%Y.
2. Deduplication
   * Check for duplicated rows based on the combination of PERSONID, GENDER, and NATIONALITY.
   * If duplicates are found, log the number of duplicates and remove them from the dataframe.
3. Age Calculation
   * Using the reference date "2023-04-24", compute the AGE for each row based on the difference between this reference date and the DOB. Set the age to None if it's calculated to be negative or if the DOB is missing.
4. Death Indicator
   * Create a new column named DEATH to indicate whether a DOE is present or not. If DOE is not null, set the value to 1 (indicating death), otherwise 0.
5. Data Value Replacement
   * Encode the POMPE column: "YES" to 1, "NO" to 0, and "UNKNOWN" to None.
   * Encode the GENDER column: "Male" to 1, "Female" to 2, and "Unknown" to None.
6. Data Type Conversion
   * Convert the GENDER column's data type to float32.
   * Convert the POMPE column's data type to int8.
   * Convert the AGE column's data type to float32.
7. Mapping Nationality to Country, Continent, and Region
   * Map the values in the NATIONALITY column to their corresponding country using the nationality\_to\_country dictionary and store the result in the COUNTRY column.
   * Map the values in the NATIONALITY column to their corresponding continent and region using the nationality\_to\_continent\_and\_region dictionary and store the results in the CONTINENT and REGION columns respectively.

A table with black text

Description automatically generated

Figure 3. Original Data

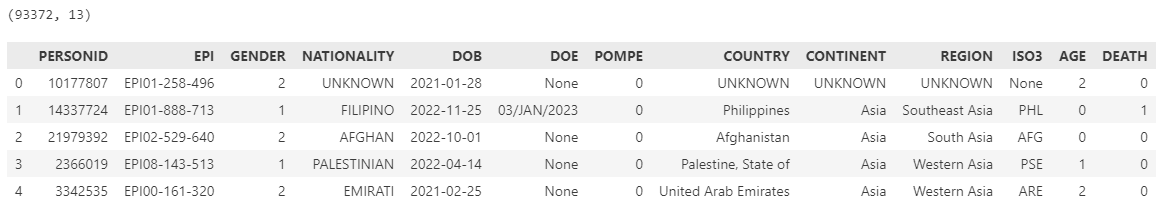


Figure 4. Preprocessed Data

**Post Preprocessing:**

**No. of Observations:** 93372

**No. of Features:** 13

# Drugs

**No. of Observations:** 4121979

**Features:**

|  |  |
| --- | --- |
| PERSONID |  |
| ENCNTRID |  |
| ORDERID |  |
| ORDERMNEMONIC |  |
| ORDERDETAIL |  |
| ORDERDATE |  |
| POMPE |  |

**High-Level Goal:** Our goal is to process the drug-related data such that for each patient, identified by their unique PERSONID, we consolidate the various drugs (or orders) they have been associated with. Each patient will be uniquely represented by a single row, with distinct drugs (or orders) represented as individual columns. In these columns, a value of 1 will indicate the association of that drug with the patient. Additionally, the patient's POMPE status will be attached to this matrix, providing a holistic view of each patient's drug history and their status on the specific attribute POMPE.

**Preprocessing Steps:**

1. Filtering Based on Unique IDs
   * Retain only those rows in the dataframe where the PERSONID matches an ID present in the unique\_ids\_list.
2. Date Conversion and Cleaning
   * Strip any white spaces from the ORDERDATE column and convert its format to a datetime object using the format %d/%b/%Y.
3. Cleaning and Encoding
   * Strip any white spaces from the ORDERMNEMONIC column.
   * Encode the POMPE column: Convert "YES" to 1 and "NO" to 0.
4. Sorting and Grouping
   * Sort the dataframe by ORDERDATE in descending order.
   * Group the dataframe by PERSONID, ENCNTRID, and ORDERMNEMONIC, and retain the last (most recent) entry for each group.
5. Pivoting Data
   * Create a new column VALUE in the grouped dataframe and set all its values to 1.
   * Pivot the grouped dataframe such that each PERSONID becomes a unique row, each distinct ORDERMNEMONIC becomes a column, and the values in these columns are filled with the corresponding VALUE. This will result in a matrix indicating which drugs (or orders) are associated with which person.
6. Merging with POMPE Information
   * Create a mapping (pompe\_mapping) that links each unique PERSONID to their POMPE status.
   * Merge the pivoted drug dataframe with this mapping based on the PERSONID to ensure that each person's drug data also has their POMPE status attached.

A screenshot of a computer

Description automatically generated

Figure 5. Original Data

A close-up of a document

Description automatically generated

Figure 6. Preprocessed Data

**Post Preprocessing:**

**No. of Observations:** 82921

**No. of Features:** 25692

# Labs

**No. of Observations:** 7769611

**Features:**

|  |  |
| --- | --- |
| PERSONID |  |
| ENCNTRID |  |
| ORDERID |  |
| ORDERCATALOG |  |
| ORDERDATE |  |
| RESULTID |  |
| RESULTVALUEALPHA |  |
| RESULTVALUE |  |
| TASKASSAY |  |
| POMPE |  |

**High-Level Goal:** Our objective is to restructure the labs-related data to ensure that each patient, identifiable by their unique PERSONID, has a comprehensive representation of their latest lab results. Each patient will be represented by a single row, with various lab order catalogs denoted as individual columns. The values within these columns will reflect the latest lab result value for that specific lab order. In addition to this, the patient's POMPE status will be appended to provide a full snapshot of each patient's labs in conjunction with their POMPE status.

**Preprocessing Steps:**

1. Filtering Based on Unique IDs
   * Retain only those rows in the dataframe where the PERSONID matches an ID present in the unique\_ids\_list.
2. Data Type Conversion
   * Convert the data type of ORDERCATALOG column to string based on the provided configuration.
3. Date Conversion and Sorting
   * Strip any white spaces from the ORDERDATE column and convert its format to a datetime object using the format %d/%b/%Y %H:%M:%S.
   * Sort the dataframe by ORDERDATE in descending order.
4. Encoding and Cleaning
   * Encode the POMPE column: Convert "YES" to 1 and "NO" to 0.
   * Retain only the first (most recent, given the sorting) unique entry for each combination of PERSONID and ORDERCATALOG.
5. Pivoting Data
   * Pivot the dataframe so that each PERSONID becomes a unique row, each distinct ORDERCATALOG becomes a column, and the values in these columns are filled with the corresponding RESULTVALUE. This transformation results in a matrix that indicates the latest lab result value for each patient for every lab order catalog.
6. Merging with POMPE Information
   * Merge the pivoted labs dataframe with the POMPE status of each patient. Ensure that the final dataframe has a POMPE status for every PERSONID.

A table with numbers and letters

Description automatically generated with medium confidence

Figure 7. Original Data

A white paper with black text

Description automatically generated

Figure 8. Preprocessed Data

**Post Preprocessing:**

**No. of Observations:** 64247

**No. of Features:** 821

# Diagnosis

**No. of Observations:** 2508846

**Features:**

|  |  |
| --- | --- |
| PERSONID |  |
| ENCNTRID |  |
| DIAGNOSISID |  |
| NOMENCLATUREID ICDCODE |  |
| ICDDESCRIPTION |  |
| DIAGNOSISDATETIME |  |
| DIAGNOSISTYPE |  |
| POMPE |  |

**High-Level Goal:** The primary aim is to achieve a structured representation of each patient's diagnosis data, with individual ICD codes transformed into distinct columns. Each patient, represented by their PERSONID, will have a row where the presence or absence of a specific ICD code during their last encounter is indicated by 1 or 0, respectively. Additionally, the patient's POMPE status is appended for a comprehensive overview. A consistent mapping between ICD codes and their descriptions is also established and saved externally for future reference.

**Preprocessing Steps:**

1. Filtering and Column Selection
   * Retain only those rows in the dataframe where the PERSONID matches an ID present in the unique\_ids\_list.
   * Keep only the columns of interest: PERSONID, ENCNTRID, ICDCODE, ICDDESCRIPTION, and POMPE.
2. Data Type Conversion
   * Convert the data types of PERSONID and ENCNTRID columns to int32.
   * Convert the data types of ICDCODE and ICDDESCRIPTION columns to string based on the provided configuration.
3. Grouping and Mapping Descriptions
   * Group the dataframe by PERSONID, ENCNTRID, and ICDCODE to retain the last encounter for every unique combination.
   * Create a mapping of each ICDCODE to its respective ICDDESCRIPTION. This mapping ensures that each ICD code has only one associated description, even if there might be slight variations in the descriptions in the raw data.
   * Update the ICDDESCRIPTION in the main dataframe using this mapping to ensure consistency.
   * Save this mapping to an external icdcodes.json file.
4. Transforming Data to Binary Matrix
   * Create a new column VALUE and assign it a constant value of 1. This will be used to fill the pivot table.
   * Pivot the dataframe so that each PERSONID becomes a unique row, each distinct ICDCODE becomes a column, and the values in these columns are filled with the VALUE column. This transformation results in a binary matrix indicating the presence (1) or absence (0) of each ICD code for every patient.
5. Encoding and Merging with POMPE Information
   * For each unique PERSONID, extract their POMPE status.
   * Encode the POMPE column: Convert "YES" to 1, "NO" to 0, and "UNKNOWN" to None.
   * Merge the pivoted ICD codes dataframe with the POMPE status of each patient. Ensure that the final dataframe has a POMPE status for every PERSONID.

A screenshot of a computer

Description automatically generated

Figure 9. Original Data

A screenshot of a computer

Description automatically generated

Figure 10. Preprocessed Data

**Post Preprocessing:**

**No. of Observations:** 89379

**No. of Features:** 7509

# Encounter



Figure 11. Original Data

**Justification for Exclusion from Data Analysis and Modelling:**

* The Encounter table typically captures specific instances when a patient interacts with the healthcare system. While these interactions may contain a wealth of granular data, they might not be directly relevant to the overarching objectives of certain analyses. For instance, specific timestamps or encounter locations may add noise without significant value for predictive modeling. Furthermore, if the model's purpose is to evaluate long-term health outcomes or patient characteristics, individual encounters might not be as informative as aggregated or high-level data.

# Orders

A screenshot of a computer

Description automatically generated

Figure 12. Original Data

**Justification for Exclusion from Data Analysis and Modelling:**

* The Orders table usually includes details about tests, medications, or interventions ordered by healthcare providers. If the primary focus of the analysis is on outcomes, conditions, or patient demographics, the granular details about individual orders might be superfluous. Moreover, the presence of numerous orders does not necessarily imply they were all fulfilled or that they directly influenced the patient's health status. Including them could complicate the modeling process without tangible benefits.

# Problems

A screenshot of a computer

Description automatically generated

Figure 13. Original Data

**Justification for Exclusion from Data Analysis and Modelling:**

* The Problems table often lists health issues, concerns, or symptoms raised during medical visits. However, if the dataset already incorporates definitive diagnosis or lab results, the problems table may introduce redundancy. Additionally, such tables might have entries based on patients' subjective reports, leading to potential inconsistencies or inaccuracies. If the goal is to work with confirmed and objective data, it might make sense to prioritize diagnoses over reported problems.

# Procedures

A screenshot of a computer screen

Description automatically generated

Figure 14. Original Data

**Justification for Exclusion from Data Analysis and Modelling:**

* The Procedures table typically captures interventions, surgeries, or procedures performed on patients. While it can offer insights into the type and intensity of care a patient received, it might not be directly pertinent to certain analyses. For instance, if the primary objective is understanding the natural progression of a disease without interventions, or if modeling the direct outcomes of specific treatments, then including all procedures might add unnecessary complexity. Additionally, without appropriate contextual data (like success rate, post-procedure complications, etc.), the raw procedure data may not be very informative.