Owkin Use Case

Thomas Le Naour

First impressions on dataset

The dataset is a structured dataset containing information about patients and their medical history.
Each row looks to be attributed to a patient, each column represents a variable. There are quantitative and qualitative variables, and temporal data since there is some event represented by a date.
With a quick observation we can see there is missing data (empty and marked with several keywords) and some columns have various mixed types.
Column naming is not really standardized. There are typos in column names (biobsy, treatmetn …)
There are some inconsistencies in rows like upper / lower strings, typo errors, inconsistent formatting.

Step 1 Dataset description

1.1 Necessary informations to understand the structure of the dataset

Information about data source and collections:
It can be useful to know from where the data is and how it was collected. It can help figuring out some inconsistency, for example, if a column comes from free text entry it can be a source of inconsistency. On the other hand, if we know it comes from a database, it can help us collect other data with the partner such as data referential for joining with our dataset.
It is also important to know how the data has already been processed.

Information about columns:
For each column we need a description of the meaning. It helps to understand data and figure out how to deal with data quality. Especially here where 3 columns are ambiguous. ab Value 1, Label Value 2 and Lab Value 3 are difficult to imagine what they represent.
It can help connect relationships between variables.

Information about types:
For each column we need to know the data type. It will help with insights for storage, loading and processing

Information about ranges and scales:
For quantitative variables, it can be very useful to have insights about ranges and scales, it can help understand data and catch outliers.

Information about how missing data:
We need to know how missing data are handled to avoid ambiguity about empty data

Informations about cardinality:
Knowing data cardinality can help detect problems about duplicates, missing data, and poor data quality. It also impacts design and storage of data, and performances.

Informations about Categorical values:
For categorical variables, we need to know unique values. It helps finding data inconsistency

1.3 Quality assessment file
For the data description, I decided to apply 3 types of descriptions:

- Date => Min and Max
- Quantitative variable => Min, Max, Quartile, Mean, standard deviation and variance
- Qualitative variable => Count by values, top value, count of top value

For all types i compute number of records, number of missing records, number of unique values
Here is an example of generated json


```
{
  "Center": {
    "count": 39,
    "missing": 0,
    "unique": 2,
    "value_counts": {
      "Owkin": 33,
      "Externe": 6
    },
    "top_value": "Owkin",
    "top_value_count": 33
  },
  "Date of Birth": {
    "count": 33,
    "missing": 6,
    "unique": 33,
    "min": "1973-03-24 00:00:00",
    "max": "2012-05-19 00:00:00"
  },
  "Pack years (PA)": {
    "count": 26,
    "missing": 13,
    "unique": 16,
    "min": 0,
    "25%": 15,
    "50%": 34.5,
    "75%": 43.75,
    "max": 90,
    "mean": 34.07692307692308,
    "std": 21.03316063157998
  }
}
```

Step 2 Dataset quality

2.1 Quality defects spotting

Date of birth:
Invalid Dates: /08/1996, 21/003/1974, NA/02/1986
Missing Datas: One value is empty
Format: Format looks to be DD/MM/YYYY, it can be ambiguous with the format MM/DD/YYYY for some records

date of diagnosis:
Invalid Dates: 28/08/01998, .
Format: Format looks to be DD/MM/YYYY, it can be ambiguous with the format MM/DD/YYYY for some records
Inconsistency: For some patients, there is an inconsistency between Date of birth and date of diagnosis, ie the diagnosis happened before the birth.
For example patient 31 is borned on 17/11/1993 and diagnosed on 01/06/1974. Some date of diagnosis and date of birth are very close, a good variable can be age at diagnosis to detect potential outliers

Smoking history:
Inconsistent values: Values differs for some records (heavy, former, 1,2)

Pack years
Inconsistent values: Mixed int and strings. There is 0 value for a former smoker
Missing Data: unk means unknown. We have a zero value*

name of treatment:
Data format: Some records has a cardinality > 1 and are stored within a string based on “+” separator.
Two records (VRD and Rd) look like they are acronym of treatments (ie VRD can equal Vincristine + Rituximab + Daratumumab for example)

Line of Treatment:
Invalid values: There is a typo for a value “/1”
Missing Values: Flagged with NA value

date of Psa level:
Invalid Dates: 28/10/20020,
Missing data: Empty record, unknow, N/A
Format: Format looks to be DD/MM/YYYY, it can be ambiguous with the format MM/DD/YYYY for some records

PSA level:
Invalid Values: juil-79, this record has nothing to do there and suggests this patient has poor quality data
Missing data: Empty record, NA, Uk (unknow ?)

ab Value,Lab Value 2, Lab Value 3
Invalid data: ., value not formatted as the other records (10:02 patient id 34)
Missi data: Empty records, None, NA
Consistency: There is an outlier in Lab Value 3 with value = 100 (patient 19), there is a record with different type (15 for patient 39)

date of biopsy:
Format: Format looks to be DD/MM/YYYY, it can be ambiguous with the format MM/DD/YYYY for some records
Inconsistency: For some patients, there is an inconsistency between Date of birth and date of diagnosis, ie the diagnosis happened before the birth.
For example patient 27 is borned on 14/09/1997 and biopsy happened on 09/05/1973

Biopsy:
Inconsistent values: Several data representations (Bone / bone, Lymph, Lymp nodes)

Status:
Inconsistent values: Several data representations (Alive, Death / Dead)
Missing datas: Empty, N/A

2.2 - 2.3
I decided to consider a representative substract of the variables of the dataset. So for next step my dataset is reduced to columns:

- Patient ID
- Date of Birth
- date of diagnosis
- Smoking history
- Pack years (PA)

We have a mix of temporale, quantitative and qualitative variables. So handling quality for those columns will help for generalization later

My pipeline is built in 3 steps
1 => Missing values identification
I flagged empty records for each column. A new column is generated with conversion {column_name}-is_empty. It contains a boolean value that is True if the record for the column is empty

2 => Malformatted records
I transform raw values to expect data type. If the transformation fails, a boolean value that is True is stored for the record in a column following name convention {column_name}-is_format_not_compliant

3 => Functional compliance
I added 3 checks that implement functional compliance
date of birth should be before date of diagnosis
smoking status should equal a value in [‘heavy’, ‘former’]

packet years should be in a predefined interval of value
For each rule, I store as boolean the result of the functional rule in a column following the previous implementation

2.4 quality assessment file
I serialize the quality assessment in a json file
The json contains macro metrics like the list of patients with defects, the count of defects by variables, the detailed variables defects by patient.
There is also metadata about quality checks, like quality checks values used.

Step 3 dashboards

I reused quality assessment files to compute macros indicators about data quality
number of patients with defects
count of defects by variables

Step 4 Curation

I transformed **Smoking history** and **Pack years** non-compliant values to compliant values

Smoking status:
I built a smoking history referential

```
old_value,new_value
heavy,heavy
heavy smoker,heavy
2,heavy
1,former
former,former
```

The purpose is to standardize the content of smoking history by transforming low population labels to standard labels. With this referential, I just need to join the dataset with and keep the new_value from the referential.

Pack years:
To improve compliance for the Pack years column, I wrote a transformation trying to catch only the integer value of the record and storing it as integer in the column. I also standardized the missing value with **-1**. It's not a relevant value for the variable so i t is a good candidate to track missing data.

Step 5 Scaling

5.1
ETL pipeline:

- Extract raw datas
- Measure defects
- Fix defects
- Measure fixed defects / Measure data quality

- Produce and store fixed dataset

Pipeline Orchestration:

- Automate the execution of the pipeline using workflow management tools or frameworks.
- Schedule regular updates or reruns of the pipeline to accommodate new data arrivals or changes in requirements.

Monitoring and alerting

- Define warning / error threshold about data quality
- Automatic notifications on threshold

Traceability:

- We need to track the origin of the data, ie which center has produced the data

5.2

POO paradigm:

Define classes for different components of the pipeline, such as data loader, variables checker, quality measurement, storing data …

Functional programming:

- Immutability
- Pure function without side effects

5.3

- Python Package
    - Dependency management
    - Versioning
- Containerization / Virtualisation
    - The artifact is a vm or a container
    - It helps ensuring the environment inside the container / vm is always the same
    - Containerization / Virtualisation component must be available in the center
- Portability / self containment of python virtual environment
    - Full python virtual environment containing dependencies / code can be shipped to the center
- Automate deployment with scripting tools like bash / ansible

5.4

- Good unit test coverage and continuous integration
    - Unit test is a good documentation
    - Unit test prevents new bugs
    - Continuous integration makes easier work integration of several developers on the same code base
- Development practice
    - Code review
    - Pair programming
    - Mob programming
- Documentation
    - Code
    - Builds
    - Deployment
    - Data