Case Study: Assessing Health Status Using Prescriptions and Diagnoses

Data Overview

In this case study, we will be looking at methods to define a patient's health status using diagnosis data, and then try to predict a patient's health status using prescription data only. This is a problem relevant in healthcare, for example as we try to understand a patient's future therapeutic needs.

For this task, we have available three datasets simulating data for about 85,000 patients over a period of 3 years:

1. **Patient Diagnosis.** Each row of this dataset contains a patient's diagnosis provided on a specific date. The diagnosis codes are presented in a standard called ICD-10.

2. **Patient Prescriptions.** Each row of this dataset contains a patient's prescription filled on a specific date. The prescriptions contain drug category, drug group, and drug class.

3. **ICD-to-Clinical Categories Map (CCS).** Each row in this file contains an ICD-10 diagnosis code (with a slightly different formatting than in the Patient Diagnosis table) and diagnosis descriptions as explained here: *https://www.hcup-us.ahrq.gov/toolssoftware/ccs10/ccs10.jsp*. Not all diagnosis codes have a CSS code, so you may have to work around this.

D&A Case Study: Example Tasks

1. **Defining a Patient's Health Status.** For this task we will focus on defining a patient's health status based on their diagnostic data. The CCS map will be useful for this. Hint: Use whatever information is appropriate in the given data sets to define a robust characterization of each patient's health status. Ideally, the characterization should be useful for establishing something like "Patient 0123 has anemia and skin infection."

2. **Predicting Heath Status using Prescription Data alone.** In this task, we will infer a patient's health status using only their prescription data. In particular, we will build a model that would allow us to potentially predict the health status of patients outside the ones provided.

Example Solution: Version 0

The solutions to both questions are well-documented in a step-by-step manner in the Jupyter Notebook (.ipynb) titled "Merck Case Study - Prescriptions vs Health Status." I'd recommend following the order of the demo in the notebook due to the potential file dependency (e.g. training data set needs to generated before model training and evaluation).

In this document, we shall highlight the key ideas and observations.

Task #1

So for first task, how do we represent a patient's health status using the diagnosis data? First, I'd like to establish the notion of temporally ordered sequence of diagnoses and treatments, where treatments in this case are the prescription drugs. The temporal sequence of diagnoses is called d-sequences for simplicity, and similarly, the temporal sequende of prescriptions are refered to as p-sequences.

A patient's health status is a time-varying state that progresses over time (aka disease progression), which can be conveniently and efficienity captured by a temporal sequence representation. Further, a patient's

health status is highly related to the recency or temoperal locality of the diagnoses, meaning that more recent the diagnoses the more pertinent to the patient's current health status.

To use diagnosis codes as a proxy for health status, we also need to consider the frequency with which a code is documented in the patient data as well as the specificity of the code. Altogether we can express the degree of importance of a diagnosis code as a function of i) recency ii) frequency and iii) specificity. For this, representing patients in terms of diagnosis sequences has an advantage of extracting these signals. I'll use TF-IDF as an example measure for the importance of a diagnosis code. In particular, a code that has a high term frequency in a d-sequence would indicate a recurrent diagnosis, thereby increasing its importance degree. However, a frequent diagnosis could be due to the fact that it is a common comorbid condition to various diseases (e.g. hypertension for CKD, diabetes, asthma patients), or it can be a common medical procedure for a primary diagnosis i.e. this patient's main health issue, which may not be documented as as often in the d-sequence. Consquenctly, we want to consider as diagnosis (i.e. ICD10) as "primary" by ensuring that it has a high specificity i.e. disease-specific and truly representative in characterizing a patient's health status. A high inverse document frequence or IDF has this property by ensuring that the candidate diagnosis has its individuality, pertaining to a particular patient and not occurring often across all patients.

In the notebook, there's more detailed discussion on this. Essentially a patient's health status can be represented by the top-n most important diagnoses (or ICD10 codes) in the patient's history i.e. d-sequence. Note that by truncating the d-sequency to the most recent n days (say n=180), we also take care of the "recency" criterion.

Similar to how d-sequences capture disease progression, the p-sequence captures the notion of treatment pathway and by the same token, we can also find the top-n most important prescriptions.

Task #2

Now, for the second task, using prescription data to predict the health status, it's important to establish the notion of "temporal alignment." Certainly, we want to try to use the most relevant prescriptions to predict a patient's health status respresented by the top-n diagnosis codes (as determined by running TF-IDF on the most recent segment of the patient's d-sequence)

In particuarl, how do we sort out the prescription relevancy? The answer is temporal alignment. Assuming that prescriptions are given to patients to treat one or more diseases as documented by one or more ICD10 codes, then for each ICD10 code and its diagnosis date (d), we can define a time window W centered on d, such that the W encloses d and is not too "wide." In a typical healthcare process, a prescription drug could be given prior to a diagnosis, say 60 days (e.g. the patient already had the health issue prior to the visit). On the other hand, prescription drugs are usually given following a diagnosis up to say 30 days, and then the patient has to schedule a follow-up visit in order to get another set of prescriptions. For instance, using the observation given above, we can set W = (60, 30), which allows us to focus only on the prescriptions as early as 60 days prior to the target diagnosis and as late as 30 days following the diagnosis.

Given the temperal alignment criterion, we will then match each diagnosis date in the d-sequence with relevant prescriptions given the window W to identify candidate prescriptions matching a given diagnosis code. In particular, we loop through all the diagnosis dates in a d-sequence, each of which is associated with a set of relevant prescriptions, qualified via W and the given diagnosis date (d). Note also that it's possible that no prescriptions can be found at all given W and a particular diagnosis date (d).

The temporal matching allows us to collect a training data for each primary diagnosis (i.e. the most important ICD10 code). For the purpose of this demo, I'll use the simpliest assumption by predicting each primary diagnosis individually and independently. Please see the notebook for more details on how to create and train the relevant classifiers.

Exploratory Data Analysis (EDA)

The first step to the analysis starts with exploratory data analysis. In particular, it's important to find out if there are missing values or errors in the data (e.g. ill-formatted ICD10 codes) that may introduce unnecessary biases to the anlaysis and modeling.

Key Observations

1. There are missing values in the diagnosis table (but not in the treatment and CCS tables)
2. Approximately 80% of patients appear in both diagnosis and treatment table; i.e. ~20% went untreated (or not documented in the treatment table)
3. For those who were treated, can we identify all of them in the diagnosis table? No, some patients appear in the treatment table but not in the diagnosis table
4. Some of the CSS-missed codes (or simply "misses") are actually documented quite often (e.g. 999.99 has 591 mentions, where 999.99 could be an mis-coded ICD-9 code)
5. Some of the misses are legit even though they are not found in the CCS table (i.e. df_res). E.g. J45.90: unspecified asthma
6. Some high-frequency ICD10 codes do not necessarily represent a health status per se but represent external factors (e.g. lab tests, symptoms) linked to the health status or disease state. E.g. Z00.00 (Encntr for general adult medical exam w/o abnormal findings) has over 20K mentions in the data set (df_diag) but it is not a direct description of a health status and therefore by itself does not say much about a patient's clinical conditions. E.g. R05 (Cough), is a symptom
7. ICD10 codes that are alphanumerically larger than O00 are often not disease-specific coding according to their standard categorization
8. As we can observe from the data, most patients tend to have more than one ICD10 code documented in their records, which can be **a recurrent health condition, the same condition with a different severity stage (hence a different code), comorbic conditions of a primary diagnosis -- or simply a new diagnosis reflecting a new episode of pathologically different heath condition than those from the past. The new diagnosis may or may not be relevant to his/her old conditions.**
9. There are 604 drug classes, 415 drug groups and 89 drug categories. Drugs are not coded.

Jupyter Notebook

Below is a highlight for the notebook covering topics ranging from EDA, analysis of the diagnosis and prescription data, "mini-sequencing pipeline" to, the matching between prescriptions and (primary) diagnoses (including how they are determined using a given measure like TF-IDF), training data creation (predicting health status using prescription data) and last but not least — model training and evaluation. Please refer to the notebook for more details. Please ignore the hyperlinks here.

Note that EDA plots and model performance plots can be generated from within the notebook. Since the data generated may take a lot of space, they are not tentatievely included as part of the solution (I will observe the size and decide on that).

Note also that each module in the solution code has demo functions for important features of the modules in addition to comments; these demo functions all have names prefixed by "demo" (e.g. demo_make_classification())

EDA

- Load data sets
- Explore data sets
- Understand the patient cohort
- Diagnosis and prescription dates
- Analyze diagnoses in terms of ICD10 codes

Defining health status

- From diagnosis codes to defining health status
- Observing patients and diagnoses as temporal sequences
- Sequencing diagnosis data
- Diagnosis sequence, disease progression and health status
- Using primary diagnoses (plural) characterize health status
    - Identifying most important diagnoses using TF-IDF measure as an example

Predicting health status using prescriptions

- Analyzing prescription data
- Encoding prescriptions
- Sequencing prescriptios (Optional)
- Identifying important prescriptions (Optional)
- Using (temporally-aligned) prescriptions to predict health status
- Model formulation
    - Matching prescriptions with diagnoses via temporal alignment
- Creating training examples
- Model training and evaluation
- Other potential methods