Case Study: Prescriptions vs Health Status

1. Defining a Patient's Health Status. Ideally, your characterization would be useful for establishing something like "Patient 0123 has anemia and skin infection."
2. Predicting Heath Status using Prescription Data alone.

EDA

Defining health status

Predicting health status using prescriptions

Import libraries

In [3]:

```
import sys, os
import re
import collections
import numpy as np
import pandas as pd
from pandas import DataFrame, Series
import matplotlib.pyplot as plt
from matplotlib.ticker import MaxNLocator # used to force yticks to assume integer values
plt.style.use('ggplot')

# user-defined libraries
# import utils
from data_pipeline import load_data, fraction_rows_missing
import feature_extractor as fext

# Import relevant libraries, classes and utilty functions
from utils import * # length_of_history, time_window, transform_diagnosis, etc.
from utils import Diagnosis, Treatment
from icd_utils import encode, decode, is_disease_specific, is_valid_icd

# set random state seed for reproducibility
np.random.seed(53)

print(f"The following analysis was performed under:\n- Python v{sys.version}\n\n- Pandas version {pd.__versi
print(f"\n- Numpy version: {np.__version__}")
# print(f"Detailed system versioning of the development stack:\n{pd.show_versions()}")
```

```
The following analysis was performed under:
- Python v3.7.1 (default, Dec 14 2018, 13:28:58)
[Clang 4.0.1 (tags/RELEASE_401/final)]

- Pandas version 1.2.0

- Numpy version: 1.19.2
```

Load data sets

- All the data processing functions are organized in the module `data_pipeline`
- We could choose to load the entire dataset -- or only a subset of the data for the ease of prototyping and testing
  - When subsetting the patient data from the diagnosis and treatment table, it's convenient as a default to subset only patients (Patient_id) that appear in both tables. Later on, when predicting the health status by prescriptions, we want to ensure that each patient has (some) records in both table for the purpose of training statistical models.
  - It's important to always check missing values (this can be achieved via `fraction_rows_missing`)

In [4]:

```
# from data_pipeline import load_data, fraction_rows_missing

# data sets: Diagnosis.csv Prescriptions.csv  ccs.csv
input_dir = os.getcwd()
diagnosis_file = "Diagnosis.csv"
treatment_file = "Prescriptions.csv"
resource_file = "ccs.csv"
diagnosis_path = os.path.join(input_dir, diagnosis_file)
treatment_path = os.path.join(input_dir, treatment_file)
resource_path = os.path.join(input_dir, resource_file)
col_key = 'Patient_id'
col_date = 'Diag_date'
col_code = 'ICD10'

n_samples = 1000

print("> Load a subset of the patient data ...") # by default, only those that appear in both diagnosis and
df_diag, df_treat, df_res = load_data(input_dir=os.getcwd(), n=n_samples, dropna=False, verbose=False)
assert len(df_diag[col_key].unique()) == n_samples
assert set(df_treat[col_key].unique()) == set(df_diag[col_key].unique())
print(f"> Num of patients in diagnosis table: {len(df_diag[col_key].unique())}")

print()
print("> Load the entire data set ...")
df_diag, df_treat, df_res = load_data(input_dir=os.getcwd(), subset=False, dropna=False, verbose=True)
print(f"> Num of patients in diagnosis table: {len(df_diag[col_key].unique())}")
print(f"> Num of patients in treatment table: {len(df_treat[col_key].unique())}")
```

```
> Load a subset of the patient data ...
> Num of patients in diagnosis table: 1000

> Load the entire data set ...
> size of diagnosis table: 660092
> size of treatment table: 1024029
> size of ccs lookup     : 72167
> Num of patients in diagnosis table: 84059
> Num of patients in treatment table: 82546
```

Explore data sets

```
- Examine each patient attribute using appropriate visualization methods
- Answer a few patient-specific queries to better understand the data
```

In [5]:

```
df_diag, df_treat, df_res = load_data(input_dir=os.getcwd(), subset=False, dropna=False, verbose=True)
Nd0, Nt0, Nr0 = df_diag.shape[0], df_treat.shape[0], df_res.shape[0]
print()

# Check missing values
r_missing = fraction_rows_missing(df_diag, verbose=True)
print(f"> Ratio of rows with nulls in diagnosis: {r_missing:.3g}, N={int(r_missing * Nd0)}") # 9.09e-06
r_missing = fraction_rows_missing(df_treat, verbose=True)
print(f"> Ratio of rows with nulls in treatment table: {r_missing:.3g}, N={int(r_missing * Nt0)}") # 0
r_missing = fraction_rows_missing(df_res, verbose=True)
print(f"> Ratio of rows with nulls in CCS table: {r_missing:.3g}, N={int(r_missing * Nt0)}") # 0

print()

# Now load only rows without missing values (this is a default behavior)
df_diag, df_treat, df_res = load_data(input_dir=os.getcwd(), verbose=True)
r_missing = fraction_rows_missing(df_treat, verbose=True)

# only diagnosis table has missing values
for name, df in [('diagnosis', df_diag), ('treatment', df_treat), ('CCS', df_res), ]:
    assert fraction_rows_missing(df) == 0, \
        "Found missing values in {} table. R(missing)={}".format(name, r_missing)

print(f"> Sample size drops from {Nd0} to {df_diag.shape[0]}")
```

```
> size of diagnosis table: 660092
> size of treatment table: 1024029
> size of ccs lookup     : 72167

(fraction_rows_missing) Rows with nulls (n=6):
Patient_id Diag_date    ICD10
    P02662       NaN    Z04.1
    P11855       NaN   Z12.11
    P13678       NaN  L03.818
    P24482       NaN    K76.0
    P27885       NaN  M05.731
    P58187       NaN T78.1XXA

> Ratio of rows with nulls in diagnosis: 9.09e-06, N=6
> Ratio of rows with nulls in treatment table: 0, N=0
> Ratio of rows with nulls in CCS table: 0, N=0

> size(df_diag): 660092 -> 660086; dropped 6 rows
> size(df_treat): 1024029 -> 1024029; dropped 0 rows
> size of diagnosis table: 660086
> size of treatment table: 1024029
> size of ccs lookup     : 72167
> Sample size drops from 660092 to 660086
```

In [6]:

```
df_diag.sample(n=5)
```

Out[6]:

	Patient_id	Diag_date	ICD10
39935	P06014	2018-03-30	M25.569
50168	P07441	2018-03-22	Z96.1
353643	P53291	2017-08-22	S51.851A
245448	P36622	2017-10-28	I10
185858	P27611	2016-04-04	I10

In [7]:

```
# Check missing data
# r_missing = fraction_rows_missing(df_treat)
# assert r_missing == 0, "Found missing values. R(missing)={}".format(r_missing)

df_treat.sample(n=10)
```

Out[7]:

	Patient_id	Prescription_date	drug_category	drug_group	drug_class	drug_code
370570	P44710	2016-08-19	Calcium Channel Blockers	Calcium Channel Blockers	Calcium Channel Blockers	D-183
183549	P12356	2015-02-17	Anticonvulsants	Anticonvulsants - Misc.	Anticonvulsants - Misc.	D-69
224226	P61683	2018-05-14	Antineoplastics and Adjunctive Therapies	Antimetabolites	Antimetabolites	D-92
872159	P47530	2016-10-09	Antidepressants	Serotonin Modulators	Serotonin Modulators	D-523
79821	P22844	2016-12-26	Adhd/Anti-narcolepsy/Anti-obesity/Anorexiants	Amphetamines	Amphetamine Mixtures	D-27
143705	P73970	2016-10-12	Antihypertensives	Antiadrenergic Antihypertensives	Antiadrenergics - Peripherally Acting	D-56
331157	P03695	2016-09-23	Ophthalmic Agents	Beta-Blockers - Ophthalmic	Beta-Blockers - Ophthalmic Combinations	D-161
799532	P16036	2015-07-28	Ulcer Drugs	Proton Pump Inhibitors	Proton Pump Inhibitors	D-489
859408	P40360	2017-05-02	Antidepressants	Selective Serotonin Reuptake Inhibitors (SSRIs)	Selective Serotonin Reuptake Inhibitors (SSRIs)	D-519
749622	P28485	2017-01-07	Analgesics - Opioid	Opioid Combinations	Opioid Combinations	D-432

In [8]:

```
# Check missing data
# r_missing = fraction_rows_missing(df_res)
# assert r_missing == 0, "Found missing values. R(missing)={}".format(r_missing)

df_res.sample(n=5)
```

Out[8]:

	diag	diag_desc	ccs_1_desc	ccs_2_desc	ccs_3_desc
59193	T458X2D	Poisn by oth prim sys and hematolog agents, sl...	Mental Illness	Suicide and intentional self-inflicted injury ...	Suicide and intentional self-inflicted injury
699	B2701	Gammaherpesviral mononucleosis with polyneurop...	Diseases of the nervous system and sense organs	Other nervous system disorders [95.]	Other nervous system disorders
32349	S4411XA	Injury of median nerve at upper arm level, rig...	Injury and poisoning	Other injuries and conditions due to external ...	Other injuries and conditions due to external ...
27582	S270XXA	Traumatic pneumothorax, initial encounter	Injury and poisoning	Crushing injury or internal injury [234.]	Crushing injury or internal injury
5331	H018	Other specified inflammations of eyelid	Diseases of the nervous system and sense organs	Eye disorders	Inflammation; infection of eye (except that ca...

Understand the patient cohort

- Unique patients
  - Is the set of patient IDs in the treatment/prescription table a subset of those in the diagosis table? No
  - how many patients went untreated (or at least treatment not observed in the data set)?
    - Number of patients in both diagnosis and treatment: 66608
    - Approximately 80% (0.7924) of the patients in diagnosis table also have treatment data
- Statistics on diagnosis dates and treatment dates
- Statistics on other diagnosis and treatment attributes
  - unique ICD codes
  - any ill-formatted codes? Yes

In [9]:

```
col_id = 'Patient_id'
uniq_patient_ids_diag = df_diag[col_id].unique()
uniq_patient_ids_treat = df_treat[col_id].unique()
n_uniq_pids_diag, n_uniq_pids_treat = len(uniq_patient_ids_diag), len(uniq_patient_ids_treat)
print(f"> Total unique patitent IDs in diagnosis table: {len(uniq_patient_ids_diag)}")
print(f"> Total unique patient IDs in treatment table: {len(uniq_patient_ids_treat)}")

# the set of(unique) patients in diagnosis table who also appear in treatment table
set_patients_diag_treat = list(set(uniq_patient_ids_diag).intersection(uniq_patient_ids_treat))
print(f"> n(overlap): {len(set_patients_diag_treat)}, ratio wrt diag: {len(set_patients_diag_treat)/n_uniq_p

# For those who were treated, can we identify all of them in the diagnosis table?
is_treated_found_in_diag = set(uniq_patient_ids_treat) <= set(uniq_patient_ids_diag)
print(f"> PatientIDs in treatment is a subset of those in diagnosis? {'Yes' if is_treated_found_in_diag else

# Some patients with diagnoses did not get treated (or at least not recorded in the treatment table)
patients_diagnosed_untreated = set(uniq_patient_ids_diag) - set(uniq_patient_ids_treat)
print(f"> Num of patients diagnosed but untreated: {len(patients_diagnosed_untreated)}")

patients_diagnosed_undiagnosed = set(uniq_patient_ids_treat) - set(uniq_patient_ids_diag)
print(f"> Num of patients treated but undiagnosed: {len(patients_diagnosed_undiagnosed)}")
```

```
> Total unique patitent IDs in diagnosis table: 84059
> Total unique patient IDs in treatment table: 82546
> n(overlap): 66608, ratio wrt diag: 0.7923958172236167
> PatientIDs in treatment is a subset of those in diagnosis? No
> Num of patients diagnosed but untreated: 17451
> Num of patients treated but undiagnosed: 15938
```

Now, let's look at the date attributes: `Diag_date` and `Prescription_date`

- A reasonable assumption is that a treatment generally comes after (some) diagnoses, but does it hold true in this dataset?
- We'll begin by defining some useful utility functions

Observe the relationship between patients and their first diagnois dates vs treatment/prescription dates

The relationship between the diagnosis date and the prescription date will be important later on when we try to predict health status (or disease states) via prescriptions; in particular, we may want to ensure that we only use **relevant prescriptions to predict a particular diagnosis**

- How do we sort out the prescription relevancy? The answer is **temporal alignment**. Assuming that prescriptions are given to treat one or more diseases as represented by one or more ICD10 codes, then for each ICD10 and its diagnosis date (d), we can define the 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 do a follow-up visit in order to get another set of prescriptions

Definition: Temporal Alignment

Denote a prescription by *p* and a diagnosis by *d*. Further let the date of a prescription be given by *t(p)* and the diagnosis date given by *t(d)*. Then, a prescription (p) is temporally aligned with a diagnosis (d) with respect to a time window *W = [w1, w2]* iff t(p) >= t(d)-w1 and t(p) <= t(d)+w2 (i.e. t(p) falls within the interval W)

- I am abusing the notation. *t(d)* can be viewed as function of *d*, or simply t subscript d
- Using the observation given above, we can then set W = (60, 60), which allows us to focus only on the prescriptions as early as 60 days prior to the **target diagnosis** and as late as 60 days following the diagnosis.
  - Certainly, the best interval depends both on the clinical condition and the treatment

Observations

- As can be seen from the example J06.9 (Acute upper respiratory infection, unspecified), we can find a subset of patients (approximately ~119 out of the 1500 patients) who had prescriptions matching the diagnosis given by W = (60, 60)

In [10]:

```
# from utils import match
from data_pipeline import intersection

dp = Diagnosis.properties
col_key = dp['id']
col_date = dp['date']
col_code = dp['code']

tp = Treatment.properties
col_date_t = tp['date']
col_class_t = tp['class']  # drug class
col_code_t = tp['code']

n_samples = 1500
# Get diagnosis and treatment table (subsampling if n_samples is given)
#####################################################
df_diag, df_treat, df_res = load_data(input_dir=os.getcwd(), n=n_samples, verbose=True)
if n_samples is None:  # need to ensure that we take only patients that appear in both tables
    set_patients = intersection(df_diag, df_treat, on='Patient_id')
    n_samples = len(set_patients)
    df_diag = df_diag[df_diag[col_key].isin(set_patients)]
    df_treat = df_treat[df_treat[col_key].isin(set_patients)]
else:
    # if n_samples is not None, it's guaranteed that patients in set_patients appear in both table
    set_patients = df_diag[col_key].unique()

N_diag = len(df_diag[col_key].unique())
N_treat = len(df_treat[col_key].unique())
assert N_diag == len(set_patients) <= n_samples, f"N_diag: {N_diag} =?= {len(set_patients)} <=? {n_samples}"
assert N_treat == len(set_patients) <= n_samples, f"N_treat: {N_treat} =?= {len(set_patients)} <=? {n_sample
#####################################################

is_prescription_encoded = col_code_t in df_treat.columns
# print(f"> col(df_treat): {df_treat.columns}")

target_codes = ['J06.9', ]
W = (60, 60)  # select a window
for target_code in target_codes:
    df_match = match(df_diag, df_treat, target_code, window=W, verbose=1)

    # pick a random n patients and observe their respective diagnosis and prescription dates (which should s
    # the date constraints)
    n_examples = 10
    pids = np.random.choice(df_match[col_key].unique(), n_examples)
    for pid, dfi in df_match[df_match[col_key].isin(pids)].groupby(col_key):
        print(f"> Patient ID: {pid}")
        assert not dfi.empty
        if is_prescription_encoded:
            print(dfi[[col_date, col_date_t, col_code_t]])
        else:
            print(dfi[[col_date, col_date_t, col_class_t]])
```

```
> size(df_diag): 660092 -> 660086; dropped 6 rows
> size(df_treat): 1024029 -> 1024029; dropped 0 rows
> size of diagnosis table: 13598
> size of treatment table: 21106
> size of ccs lookup     : 72167
(match) Total number of patients: 1500
(match) Number of patients with diagnosis J06.9: 131
(match) Number of patients who had been given prescriptions for diagnosis J06.9: 131
... n(diagnosed): 131 >=? n(treated): 131
(match) diagnosis table with the window of dates determined (W=(60, 60)):
Index(['Patient_id', 'Diag_date', 'ICD10', 'Diag_min', 'Diag_max'], dtype='object')

... sampled dataframe:
+--------------+-------------+---------+------------+------------+
| Patient_id   | Diag_date   | ICD10   | Diag_min   | Diag_max   |
|--------------+-------------+---------+------------+------------|
| P01734       | 2017-05-31  | J06.9   | 2017-04-01 | 2017-07-30 |
| P01734       | 2018-05-15  | J06.9   | 2018-03-16 | 2018-05-15 |
| P01769       | 2017-03-04  | J06.9   | 2017-01-03 | 2017-05-03 |
| P01776       | 2018-01-16  | J06.9   | 2017-11-17 | 2018-03-17 |
| P02131       | 2018-03-28  | J06.9   | 2018-01-27 | 2018-05-15 |
| P02131       | 2018-04-01  | J06.9   | 2018-01-31 | 2018-05-15 |
| P02880       | 2016-03-11  | J06.9   | 2016-01-11 | 2016-05-10 |
| P04841       | 2016-09-01  | J06.9   | 2016-07-03 | 2016-10-31 |
| P05720       | 2015-12-15  | J06.9   | 2015-10-16 | 2016-02-13 |
| P06881       | 2016-09-04  | J06.9   | 2016-07-06 | 2016-11-03 |
+--------------+-------------+---------+------------+------------+

/Users/pleiades/Documents/work/interview/Merck/utils.py:335: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df[col] = s.apply(lambda x: x[i])
(match) Found n=120 patients with matching prescription records
> Patient ID: P02131
    Diag_date Prescription_date drug_code
23  2018-03-28        2018-03-28     D-134
24  2018-03-28        2018-04-01     D-146
25  2018-03-28        2018-02-19     D-303
26  2018-03-28        2018-02-19     D-393
27  2018-03-28        2018-04-01     D-399
28  2018-03-28        2018-03-07     D-430
29  2018-04-01        2018-03-28     D-134
30  2018-04-01        2018-04-01     D-146
31  2018-04-01        2018-02-19     D-303
32  2018-04-01        2018-02-19     D-393
33  2018-04-01        2018-04-01     D-399
34  2018-04-01        2018-03-07     D-430
> Patient ID: P06881
    Diag_date Prescription_date drug_code
60  2016-09-04        2016-09-04      D-25
61  2016-09-04        2016-10-16     D-156
62  2016-09-04        2016-10-16     D-287
63  2016-09-04        2016-10-22     D-536
> Patient ID: P13228
     Diag_date Prescription_date drug_code
118  2017-02-14        2017-02-14     D-134
119  2017-02-14        2017-02-14     D-146
120  2017-02-14        2017-02-28     D-159
> Patient ID: P19603
     Diag_date Prescription_date drug_code
203  2018-03-16        2018-04-10      D-45
204  2018-03-16        2018-03-16     D-146
205  2018-03-16        2018-03-16     D-206
206  2018-03-16        2018-04-07     D-206
207  2018-03-16        2018-04-01     D-303
208  2018-03-16        2018-04-27     D-303
209  2018-03-16        2018-03-02     D-303
> Patient ID: P21859
     Diag_date Prescription_date drug_code
219  2017-12-12        2018-01-30      D-25
220  2017-12-12        2017-10-31      D-91
221  2017-12-12        2017-12-24      D-91
222  2017-12-12        2018-01-13      D-91
223  2017-12-12        2017-11-26      D-91
224  2017-12-12        2018-01-13     D-153
225  2017-12-12        2017-11-26     D-153
226  2017-12-12        2017-10-31     D-153
227  2017-12-12        2017-10-31     D-159
228  2017-12-12        2018-01-13     D-159
229  2017-12-12        2017-11-26     D-159
230  2017-12-12        2017-12-24     D-159
> Patient ID: P40535
     Diag_date Prescription_date drug_code
326  2016-05-18        2016-05-18     D-136
327  2016-05-18        2016-05-18     D-146
328  2016-05-18        2016-05-16     D-163
329  2016-05-18        2016-04-12     D-163
330  2016-05-18        2016-05-05     D-229
331  2016-05-18        2016-04-12     D-275
332  2016-05-18        2016-05-18     D-287
> Patient ID: P60363
     Diag_date Prescription_date drug_code
527  2017-04-16        2017-04-22     D-146
528  2017-04-16        2017-02-15     D-155
529  2017-04-16        2017-06-03     D-155
530  2017-04-16        2017-06-01     D-287
531  2017-04-16        2017-03-20     D-394
532  2017-04-16        2017-04-02     D-519
> Patient ID: P60412
     Diag_date Prescription_date drug_code
537  2017-09-21        2017-09-21     D-349
> Patient ID: P82887
     Diag_date Prescription_date drug_code
723  2016-02-17        2016-02-17      D-25
724  2016-02-17        2016-02-17     D-287
725  2016-02-17        2016-02-17     D-410
> Patient ID: P84382
     Diag_date Prescription_date drug_code
728  2018-02-27        2018-03-11      D-27
729  2018-02-27        2018-02-27     D-399
730  2018-02-27        2018-04-15     D-519
731  2018-02-27        2018-02-21     D-519
```

`Patient_id` vs number of sessions in diagosis table and treatment table

- Number of sessions are defined by counting the distinct values of diagosis dates and treatment dates
- Sort patients according to the number of sessions in decending order and with the same patient ordering, find their respective numbers of treatment sessions (i.e. number of prescription dates)

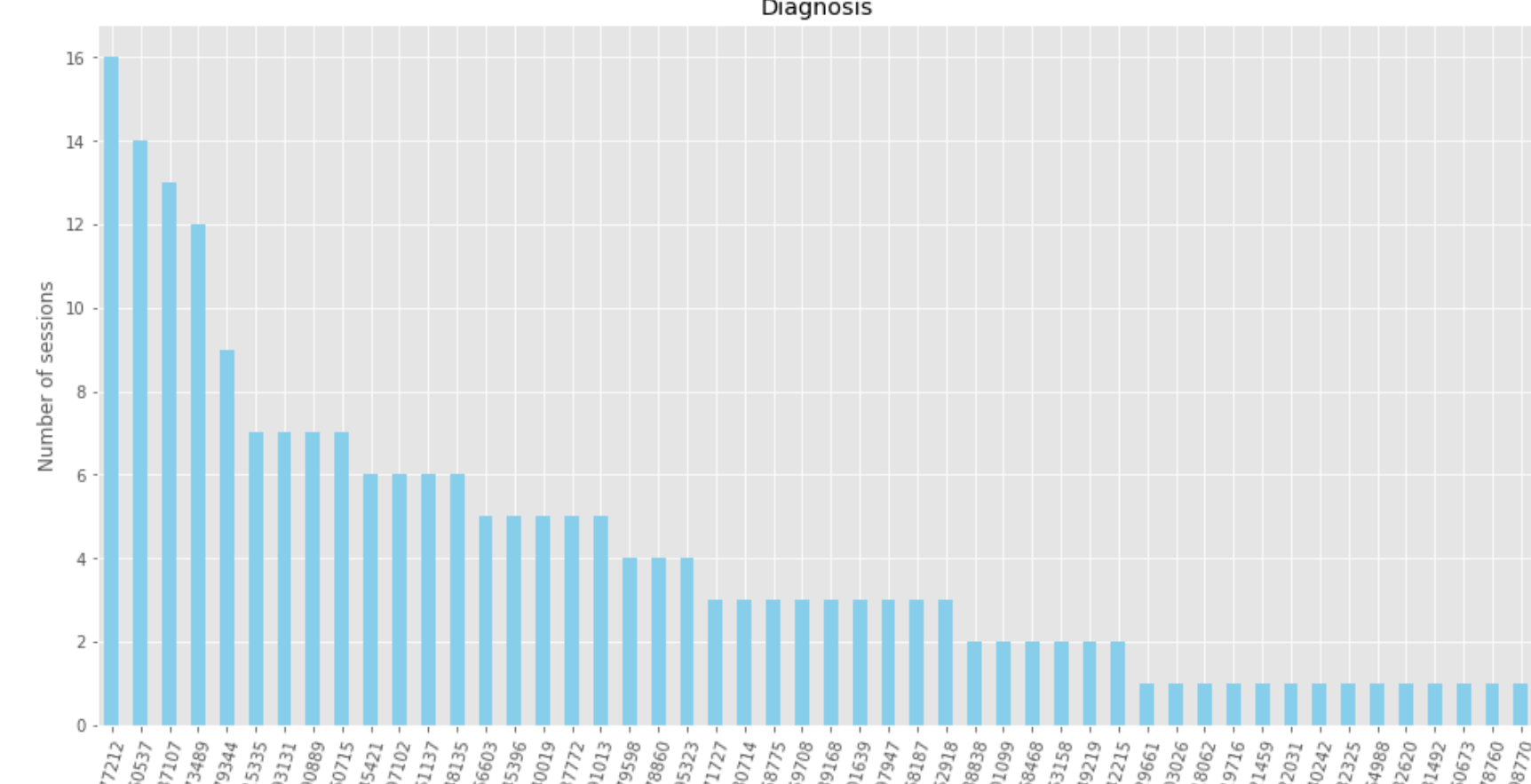
In [11]:

```
from plot_utils import plot_sessions

ax = plt.figure().gca()
ax.yaxis.set_major_locator(MaxNLocator(integer=True)) # force yticks to use integers

# np.random.seed(53)
df_diag_sub, df_treat_sub, _ = load_data(input_dir=os.getcwd(), n=50)
deidentify = True
pids = plot_sessions(df_diag_sub, show_key=deidentify, col='Diag_date', title='Diagnosis')

# plot patients (IDs) vs num of sessions according to the order given by the diagnosis table above
_ = plot_sessions(df_treat_sub, show_key=deidentify, col='Prescription_date', title='Treatment', pids=pids)
```



Analyze diagnoses in terms of ICDs across different time stamps (or sessions)

Utilities of analyzing ICD10 codes are given the module `icd_utils`

- Below is an example EDA session to verify ICD10 codes given in the diagnosis table
  - Number of unique codes: 14257
  - Are there any ill-formatted codes? Yes, n=14
  - Number of codes not found in CCS? n=83

Observations

- There are totally 83 codes not found in CCS. Let's refer to each of these codes as a "miss" for simplicity.
  - Among these misses, a large fraction of them seem to be valid ICD10 codes; we could verify them individually and choose to keep them in the analysis. However, for the purpose of a prototype, we could also ignore them temporarily because their ratio is low.
  - Some of the misses may be ICD-9, which can be converted to ICD-10 with appropriate software.

In [12]:

```
# Import relevant libraries, classes and utilty functions
from data_pipeline import analyze_diagnosis

df_diag, codebook = analyze_diagnosis(verbose=1)
```

```
(1) Number of unique codes: 14257
(2) Invalid codes (n=14):
['999.99', '999.9', '036.8130', 'KHD.X', '000', '309.28', '281', '411', '742.3', 'MHD.X', '065.5', '327.23',
'167.1', '000.0']

(3) Num of codes not found in CCS (n=83):
['I83.89', 'Z12.1', 'S30.0', 'C44.5', 'S02.612', 'S93.401', 'S92.11A', 'S69.91', 'S80.211', '999.99', 'V50.
9', 'K52.80', 'S92.352', '000.0', 'R20', 'Z80.401', '167.1', '742.3', 'M25.11', 'H40.113', 'S99', 'R51.67',
'T42.0', 'S83.501', 'E07.8', 'V70.0', 'N04.59', 'H10.42', 'Q33.5XX1', 'M75.12', 'I83.8', '999.9', 'N30.4',
'M97.645', 'O47.88', 'H66.3', 'M21.61', 'S83.512', '000', 'K11.2', 'Z00.0', 'O29', 'S00.06', 'H10.2', 'C47.0
3', 'D37.0', 'M75.11', 'B08', '065.5', 'O26.2', 'R29.70', 'S73', 'M20.1', 'N64.80', 'Z00.1', 'J45.90', 'S90.
112', 'KHD.X', 'M35.0', 'R88.82', '327.23', 'V22.1', 'Z01.41', '411', 'Z48.81', 'MHD.X', 'F04.2', 'S27.00X
X', 'C50.91', '281', 'S86.011', 'H40.133', 'S76.111', '309.28', 'H90.1', 'M11.16', 'F78.4', 'S00', 'S42.41
3', 'J45.4', '036.8130', 'K59.0', 'J01.1']

(3.1) Codes missed by CCS must be invalid? True
(5) List example ICDs across sessions ...

... PID=P00001
... date: Diag_date
... dfi:
   Patient_id   Diag_date    ICD10
0      P00001  2015-12-06    N92.6
1      P00001  2015-12-06  O26.842
2      P00001  2015-12-13  O26.842
3      P00001  2016-02-29  O26.843
4      P00001  2016-03-27  O26.843
5      P00001  2016-04-03  O26.843
6      P00001  2016-04-10  O26.843
7      P00001  2016-04-11  O26.843
8      P00001  2016-04-13   O34.21
9      P00001  2016-04-13      O80
10     P00001  2016-04-13   Z38.01
11     P00001  2016-04-14   Z38.01
12     P00001  2016-04-15   Z38.01
13     P00001  2016-04-16   Z38.01
14     P00001  2016-04-17   Z38.01

... PID=P00002
... date: Diag_date
... dfi:
   Patient_id   Diag_date    ICD10
15     P00002  2016-02-23   Z82.49
16     P00002  2017-01-31   F43.20
17     P00002  2017-01-31   Z00.00
18     P00002  2017-05-19    R07.9
19     P00002  2017-05-20   R07.89
20     P00002  2017-06-05  I25.118

... PID=P00003
... date: Diag_date
... dfi:
   Patient_id   Diag_date   ICD10
21     P00003  2017-07-26  M20.22
22     P00003  2017-08-16  M20.22

... PID=P00004
... date: Diag_date
... dfi:
   Patient_id   Diag_date   ICD10
23     P00004  2018-03-16  Z00.00
24     P00004  2018-03-19   Z11.1

... PID=P00005
... date: Diag_date
... dfi:
   Patient_id   Diag_date    ICD10
25     P00005  2018-02-13  G43.909
26     P00005  2018-02-14  G43.001

... PID=P00006
... date: Diag_date
... dfi:
   Patient_id   Diag_date   ICD10
27     P00006  2016-03-29  R07.89
28     P00006  2016-04-02  Z00.00
29     P00006  2016-04-25  R06.02
30     P00006  2016-04-26  R09.82
31     P00006  2016-06-05   E03.9
32     P00006  2016-06-05   E78.2
```

In [13]:

```
codes_missed = codebook['missed']
code_counts = collections.Counter(dict(df_diag[col_code].value_counts()))

# Among all the codes that are missed by CCS, list their documentation frequencies:
# Are they simply errors, systematic anomelies, or perhaps "valid" but not in ICD10?
for code in sorted(codes_missed, key=lambda x: code_counts[x], reverse=True)[:20]:
    code_eff = decode(code)
    assert code_eff in code_counts, "code {} not found".format(code_eff)
    print(f"> Code missed {code_eff}: {code_counts[code_eff]}")
```

```
> Code missed 999.99: 591
> Code missed KHD.X: 35
> Code missed 000: 21
> Code missed S00.06: 12
> Code missed K59.0: 6
> Code missed S86.011: 4
> Code missed Z01.41: 3
> Code missed Z00.0: 2
> Code missed Z00.1: 2
> Code missed J45.90: 2
> Code missed V22.1: 2
> Code missed MHD.X: 2
> Code missed S27.00XX: 2
> Code missed 309.28: 2
> Code missed I83.89: 1
> Code missed Z12.1: 1
> Code missed S30.0: 1
> Code missed C44.5: 1
> Code missed S02.612: 1
> Code missed S93.401: 1
```

From diagnosis codes to defining health status

Observations

- 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)
- 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
- 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
  - ICD10 codes that are alphanumerically larger than **O00** are often not disease-specific coding according to their standard categorization
- Some disease-specific ICD10 codes are mentioned very frequently because they are often concomitant relative to something more critical. E.g. I10 (hypertension) has 16315 mentions but it may be a comorbid condition to diabetes, CKD, etc.
  - As will be discussed shortly, we can sort the ICD10 codes according to their associated time stamps and thereby form a diagnosis-specific temporal sequence, which can give us a quick overview of how the patient's disease state progresses over time. This will serve as a basis for computing signficance of each diagnosis later on.
  - E.g. Patient (ID=*P52427*) has the diagnosis sequence `R05 I10 J45.40 K76.0 R10.13 I10 K57.92 K57.92 K57.92 R10.30`, where **J45.40 (asthma)** could be the primary diagnosis, for which **I10** is a correlated condition and thus tends to co-occur often with J45.40, i.e. patients with asthma are more likely to develop hypertension (see this paper for more details)

In [14]:

```
# Example codes that are (probably) not disease-specific but relevant external factors
# (e.g. medical procedures) linked to the patient's health conditions.
codes_aux = ['R0609', 'Z4732', 'Z1211', 'Z23']
df_res[df_res['diag'].isin(codes_aux)]
```

Out[14]:

	diag	diag_desc	ccs_1_desc	ccs_2_desc	ccs_3_desc
22550	R0609	Other forms of dyspnea	Diseases of the respiratory system	Other lower respiratory disease [133.]	Other lower respiratory disease
71018	Z1211	Encounter for screening for malignant neoplasm...	Symptoms; signs; and ill-defined conditions an...	Factors influencing health care	Other screening for suspected conditions (not ...
71141	Z23	Encounter for immunization	Infectious and parasitic diseases	Immunizations and screening for infectious dis...	Immunizations and screening for infectious dis...
71408	Z4732	Aftercare following explantation of hip joint ...	Symptoms; signs; and ill-defined conditions an...	Factors influencing health care	Other aftercare

In [15]:

```
# Example codes that are likely to be comorbid conditions (rather than the main diagnosis)
_, _, df_res = load_data(input_dir=os.getcwd(), subset=False, verbose=0)
codes_secondary = ['I10', 'M542']
df_res[df_res['diag'].isin(codes_secondary)]
```

Out[15]:

	diag	diag_desc	ccs_1_desc	ccs_2_desc	ccs_3_desc
8532	I10	Essential (primary) hypertension	Diseases of the circulatory system	Hypertension	Essential hypertension
14772	M542	Cervicalgia	Diseases of the musculoskeletal system and con...	Spondylosis; intervertebral disc disorders; ot...	Spondylosis; intervertebral disc disorders; ot...

In [16]:

```
# top 10 diagnosis codes
codes_valid = codebook['valid'] # set(codes) - set(codes_invalid)
codes_disease = codebook['disease']
print(f"> Found {len(codes_valid)} valid ICD10 codes.")
print(f"> Found {len(codes_disease)} disease-specific codes (ratio: {len(codes_disease)/len(codes_valid)})")
print(f"> examples:\n{list(codes_valid)[:10]}\n")
print()

topn = 15
print(f"> 1. Most popular diagnosis codes ...") # remember to call encode() before table lookup!!!
for code in sorted(codes_valid, key=lambda x: code_counts[x], reverse=True)[:topn]:
    print(f"> {code} (freq={code_counts[code]}): {df_res[df_res['diag']==encode(code)]['diag_desc'].values}"

print()
print(f"> 2. Most popular disease-specific diagnosis codes ...")
for code in sorted(codes_disease, key=lambda x: code_counts[x], reverse=True)[:topn]:
    print(f"> {code} (freq={code_counts[code]}): {df_res[df_res['diag']==encode(code)]['diag_desc'].values}"

# Among all the patients with I10 (hypertension), can we find
# a reasonable primary diagnosis for which hypertension could be just a concomitant condition (e.g. CKD, dia
print()
codes_secondary = ['I10', 'M54.2']
n_samples = 5
pids = df_diag[df_diag[col_code].isin(codes_secondary)][col_key].unique()
pids = np.random.choice(pids, min(n_samples, len(pids)))
df_seq = transform_diagnosis(df_diag[df_diag[col_key].isin(pids)])
col_key, col_date, col_code, *_ = df_diag.columns
col_intv = 'History'
for pid, row in df_seq.iterrows():
    print(f"> ID:       {row[col_key]}")
    print(f"> history: {row[col_intv]} days")
    print(f"> sequence:\n{row[col_code]}\n")
```

```
> Found 14243 valid ICD10 codes.
> Found 7852 disease-specific codes (ratio: 0.5512883521729972)
> examples:
['C81.74', 'M71.579', 'N44.1', 'T43.011A', 'N82.3', 'G91.0', 'T45.1X5D', 'D80.2', 'T38.4X5S', 'F40.11']


> 1. Most popular diagnosis codes ...
> Z00.00 (freq=23987): ['Encntr for general adult medical exam w/o abnormal findings']
> I10 (freq=16315): ['Essential (primary) hypertension']
> Z01.419 (freq=12624): ['Encntr for gyn exam (general) (routine) w/o abn findings']
> Z12.31 (freq=8870): ['Encntr screen mammogram for malignant neoplasm of breast']
> M54.5 (freq=8410): ['Low back pain']
> Z00.129 (freq=7805): ['Encntr for routine child health exam w/o abnormal findings']
> J06.9 (freq=7766): ['Acute upper respiratory infection, unspecified']
> Z23 (freq=6889): ['Encounter for immunization']
> J02.9 (freq=6174): ['Acute pharyngitis, unspecified']
> E11.9 (freq=6043): ['Type 2 diabetes mellitus without complications']
> R05 (freq=5567): ['Cough']
> R07.9 (freq=4361): ['Chest pain, unspecified']
> Z00.01 (freq=3960): ['Encounter for general adult medical exam w abnormal findings']
> M54.2 (freq=3880): ['Cervicalgia']
> J30.1 (freq=3767): ['Allergic rhinitis due to pollen']

> 2. Most popular disease-specific diagnosis codes ...
> I10 (freq=16315): ['Essential (primary) hypertension']
> M54.5 (freq=8410): ['Low back pain']
> J06.9 (freq=7766): ['Acute upper respiratory infection, unspecified']
> J02.9 (freq=6174): ['Acute pharyngitis, unspecified']
> E11.9 (freq=6043): ['Type 2 diabetes mellitus without complications']
> M54.2 (freq=3880): ['Cervicalgia']
> J30.1 (freq=3767): ['Allergic rhinitis due to pollen']
> G47.33 (freq=3733): ['Obstructive sleep apnea (adult) (pediatric)']
> N39.0 (freq=3592): ['Urinary tract infection, site not specified']
> J01.90 (freq=3477): ['Acute sinusitis, unspecified']
> E11.65 (freq=3407): ['Type 2 diabetes mellitus with hyperglycemia']
> E78.5 (freq=3373): ['Hyperlipidemia, unspecified']
> E03.9 (freq=3180): ['Hypothyroidism, unspecified']
> L70.0 (freq=3006): ['Acne vulgaris']
> J20.9 (freq=3000): ['Acute bronchitis, unspecified']

> ID:       P13075
> history: 175 days
> sequence:
K57.30 K57.32 K57.90 R19.8 Z45.2 K57.32 K57.32 K57.32 K57.32 K57.32 M10.00 K57.32

> ID:       P40128
> history: 67 days
> sequence:
I10 I10

> ID:       P43465
> history: 177 days
> sequence:
H25.043 I62.9

> ID:       P62527
> history: 82 days
> sequence:
```

```
L65.9 Z00.00 I10 Z12.31 I10 N85.8 N92.1 I10

> ID:      P74779
> history: 147 days
> sequence:
K29.70 K86.2 I10 R22.42 K74.60 K74.60 K74.69 E11.29
```

Observing patients and their diagnoses as temporal sequences

- How long are their medical histories?
- List all the diagnoses and their dates from the first session to the last

Observations

- The longest medical history is 1339 days long
- (Most) patients tend to have very different diagnoses at different times throughout their medical history. To characterize a patient's health status or disease state, we would want to focus on:
  - 1) Recency or temporal locality (e.g. most recent 180 days): Adjustable depending on the chronicity of the disease.
  - 2) The degree of importance for a given diagnosis (as proxy-represented by one or more ICD10 codes): In particular, the **"importance weight"** of an ICD code is related to their **recency**, **frequency** and **specificity** for an individual's disease state. Assuming that the recency is already being qualified by 1), we can determine the importance weight of an ICD10 code by **some function of a ICD10's documentation frequency** -- and how "special" this diagnosis is to a particular patient. Why?
    - Two patients X and Y may share similar overall diagnoses (e.g. they both have hypertension, or I10, frequently occurred in their records); however, their primary diagnoses may be quite different. E.g. X could be a patient with CKD as the main health issue, whereas Y has been a long-term type 2 diabetes patient, both are at higher risk of experiencing hypertension episodes or even cardiovascular conditions.
  - 3) As mentioned earlier, some ICD codes are not directly indicative of a health status while others may be comorbid conditions (i.e. not primary diagnoses to be informative for defining health status).

Note that the aforementioned criteria (i.e. temperal locality and frequency) are just an approximation to the patient's true disease state. The "true" disease state is actually a **time series** and therefore it changes over time.

To consider a more precise characterization of a patient's health status, we can also consider the temporal sequence of ICD10 codes obtained from the most recent medical history (e.g. codes assigned during the past 180 days) as an even better approximation for a patient's disease state, given that the ordering of diagnoses could serve as a **proxy representation for disease progression**.

As can be observed from the data, most patients tend to have more than one ICD10 code documented in their records, which could be **i) a recurrent health condition ii) the same condition with a different severity stage (hence a different ICD code) iii) comorbid conditions for a primary diagnosis -- or iv) simply a new diagnosis reflecting a new episode of pathologically different heath condition than those from the past. Any new ICD assignment may or may not be relevant to his/her old conditions.**

- E.g. Among those patients with COPD (J44.9), we could very well observe multiple other diagnosis codes (that tend to co-occur with the primary diagnosis) such as i) J42 (unspecified chronic bronchitis) as a comorbid condition ii) tobacco use (Z72.0), as a another external factor that would affect the health status ... //this is my own example, not what's being observed in this data set

Not all diagnosis codes represent diseases but they can be symptoms, abnormal lab findgs, injuries, and other factors that affect health status but nonetheless, they do not represent "heatlh status" by themselves.

In [ ]:

```python
from utils import demo_sequencing, subsample_seq
from IPython.display import clear_output

sample_size = 1000
df_seq = demo_sequencing(dtype='diag', verbose=1)

print()
df_diag, df_treat, df_res = load_data(input_dir=os.getcwd(), subset=False, verbose=False)
pids_anomely = ['P00021', ]
for r, row in df_diag[df_diag[col_key].isin(pids_anomely)].iterrows():
    print(row)
# clear_output(wait=False)
```

Sequencing diagnosis and loading sequenced data

In [ ]:

```python
from utils import subsample_seq

df_diag = Diagnosis.load(dtype='in', subset=False, verbose=0)
Np = len(df_diag[col_key].unique())
tLoad = True

diag = Diagnosis() # create a Diagnosis object
if tLoad:
    df_seq = Diagnosis.load(dtype='seq') # load the pre-computed sequence data, because it's take a minute o
else:
    diag.init(**kargs) # ensure that 'diag' is hooked to the diagnosis table
    df_seq = diag.sequence(tFilterByICD=True, tFilterByLength=True, n_days_lookback=180)
    # note: set tFilterByICD to True to only include valid (well-formatted) ICD10 codes
    #       set tFilterByLength to False to include the entire d-sequence for each patient
    #       Say you want to focus on only the most recent 100 days of diagnoses, then pass
    #       n_days_lookback=100

Ns = len(df_seq[col_key].unique())
print(f"Number of (unique) patients in original data: {Np}, in sequenced file: {Ns} | {Np} >=? {Ns}")
print(f"Longest length of medical records (days): {df_seq[col_intv].max()}")

print(f"\nShow example d-sequences ...")
subsample_seq(df_seq, n=10, verbose=True)
```

Diagonsis sequence, disease progression and health status

- Since each patient can have a long medical history, which equates to a long time series of diagnosis. The diagonsis sequence sheds light on how a disease progresses over time and thereby defining health status, which is time-varying.
- We can simplify the representation of the diagnosis time series by representing them as a temporally ordered sequence of diagnosis codes like so: `L57.8 M25.512 S43.085A S43.085A S43.085A S43.085A S43.085A S43.085A`
- To distinguish between primary and secondary diagnoses as pertaining to comorbid conditions, domain knowledge is often required but we could apply simple heurstics to tease out the main diagnoses

- We will use TF-IDF as an example measure of diagnostic importance via the following steps:
  - First, we represent each patient in terms of a temporally-ordered diagnostic sequence, which we will refer to as simply -- diagnositic sequence, or d-sequence.
  - Treat each d-sequence as a (patient) document
  - Compute a TF-IDF score for each ICD10 associated with a d-sequence
    - Intuitively, ICD10 codes that occur frequently in a d-sequence are important because they have high "term frequencies (TFs)"
    - However, the ICD10 codes that are individually important for a patient should be clinically more important than those that tend to occur in different patients' diagnoses (common comorbid conditions, seasonal flu, etc.); this is measured by ICD10's inverse document frequencies (IDFs)

Using the primary diagnoses (plural) to characterize health status

- We could pick the top-n ICD codes with the highest importance score (e.g. TF-IDF score) as the primary diagnoses
  - Note that this should be considered as **an approximation to a patient's true health status** (although diagnosis codes themselves are merely a proxy to a patient's health status because they are not always accurate)
  - D-sequence contains more information about a patient's health status but in order to use prescriptions to predict health status, d-sequence is not as simple to model as a "label."
  - Later on, by representing prescriptions as temperal sequences like a d-sequence, it's also possible to apply sequence models (e.g. seq2seq, Siamese networks) to predict the health status encoded by d-sequences; however, in this prototype, we shall consider simpler methods

Representing patients by their d-sequences

- As mentioned before, recency is an important factor for qualifying a patient's health status; from the EDA, we know that the longest medical history in this dataset is over 3 years (1300+ days), which is perhaps too long to be factored in a patient's health status.
- In the following demo, we shall consider the **truncated d-sequence** for each patient (180 days), out of which we then compute the importance score for each ICD code.
  - Alternatively, pass different values to `n_days_lookback` parameter (as an argument for `Diagnosis.sequence()`) to truncate the older segment of the d-sequence at varying degrees. For instance, if you want to focus on only the most recent 60 days of diagnoses, then specify `n_days_lookback=60` also in `Diagnosis.sequence()` . Patients with predominently acute conditions would be good use cases for a more restrictive constraint.
  - Diagnosis dates are lost in d-sequence by default (only the ordering is preserved). If you wish to include time stamps in the d-sequence, please set `tIncludeTime` to True.
  - The `History` attribute in the resulting d-sequence dataframe (`df_seq`) is measured in **days**.

Estimating the most important diagnoses

- Compute the top-n most important diagnoses according to a given measure (e.g. TF-IDF)
- We'll apply TF-IDF on d-sequences as a measure of importance
  - The top-n ICD10 codes, where n=3 in the following demo, are considered as the most representative diagonses that collectively describe a patient's health status

In [17]:

```python
from utils import demo_topn_diagnoses
demo_topn_diagnoses(topn=20)
```

```
[Diagnosis] Loading sequenced diagnosis file from:
/Users/pleiades/Documents/work/interview/Merck/Diagnosis-sequenced.csv

(eval_topn_features) Nd: 84051, dim(Xtr): (84051, 11863), size(vocab): 11863
... example feature names (n=11863=?=11863):
['A02.0', 'A03.8', 'A04.0', 'A04.1', 'A04.4', 'A04.5', 'A04.7', 'A04.71', 'A04.72', 'A04.8', 'A04.9', 'A05.8', 'A05.9', 'A06.0', 'A06.9', 'A07.1', 'A07.3', 'A07.8', 'A08.0', 'A08.11', 'A08.19', 'A08.2', 'A08.39', 'A08.4', 'A08.8', 'A09', 'A15.0', 'A15.6', 'A15.7', 'A15.9', 'A150', 'A17.81', 'A18.4', 'A26.0', 'A28.1', 'A31.0', 'A31.9', 'A35', 'A37.00', 'A37.90', 'A38.8', 'A38.9', 'A40.1', 'A40.8', 'A40.9', 'A41.01', 'A41.02', 'A41.1', 'A41.50', 'A41.51']
['Z95.4', 'Z95.5', 'Z95.810', 'Z95.811', 'Z95.818', 'Z95.820', 'Z95.828', 'Z95.9', 'Z96.0', 'Z96.1', 'Z96.22', 'Z96.41', 'Z96.5', 'Z96.60', 'Z96.611', 'Z96.612', 'Z96.622', 'Z96.641', 'Z96.642', 'Z96.643', 'Z96.649', 'Z96.651', 'Z96.652', 'Z96.653', 'Z96.659', 'Z96.7', 'Z96.81', 'Z96.9', 'Z97.10', 'Z97.3', 'Z97.5', 'Z97.8', 'Z98.0', 'Z98.1', 'Z98.2', 'Z98.41', 'Z98.51', 'Z98.52', 'Z98.61', 'Z98.62', 'Z98.811', 'Z98.818', 'Z98.82', 'Z98.84', 'Z98.86', 'Z98.89', 'Z98.890', 'Z98.891', 'Z99.11', 'Z99.81']

--- doc #11109 ---
  feature  score
0  Z01.419    1.0
...... top N(3) features: Z01.419

--- doc #23794 ---
  feature  score
0  Z00.00    1.0
...... top N(3) features: Z00.00

--- doc #24446 ---
  feature  score
0     I10    1.0
...... top N(3) features: I10

--- doc #26997 ---
  feature     score
0   D59.4  0.542917
1   D58.0  0.524015
2   D59.9  0.510604
...... top N(3) features: D59.4 D58.0 D59.9

--- doc #37658 ---
  feature  score
0     Z23    1.0
...... top N(3) features: Z23

--- doc #55292 ---
  feature     score
0   E29.9  0.645367
1  H00.11  0.519681
2  D22.30  0.481650
...... top N(3) features: E29.9 H00.11 D22.30

--- doc #57472 ---
  feature  score
0   E03.9    1.0
...... top N(3) features: E03.9

--- doc #69369 ---
  feature  score
0  Z00.129    1.0
...... top N(3) features: Z00.129

--- doc #75296 ---
  feature     score
0  M67.911  0.583284
1   K70.30  0.524527
2  D61.818  0.308749
...... top N(3) features: M67.911 K70.30 D61.818

--- doc #81402 ---
  feature  score
0     C73    1.0
...... top N(3) features: C73

(eval_topn_features) top N features overall across all docs ...
... doc(mean):
  feature     score
0   Z00.00  0.070945
1  Z01.419  0.036782
2      I10  0.033008
3  Z00.129  0.028271
4    J06.9  0.024895
5      Z23  0.021805
6    J02.9  0.020129
7   Z12.31  0.019765
8      R05  0.015207
9   Z00.01  0.014022

> There are n=10678 (unique) primary diagnosis codes in total.
> Most common primary diagnosis: Z00.00, n=8834
> Least common primary diagnosis: O65.5, n=1
> size(df_diag): 660092 -> 660086; dropped 6 rows
> size(df_treat): 1024029 -> 1024029; dropped 0 rows
> size of diagnosis table: 660086
> size of treatment table: 1024029
> size of ccs lookup     : 72167
> Most most common conditions:
['I10' 'J06.9' 'J02.9' 'E11.9' 'M54.5' 'J01.90' 'J20.9' 'E78.5' 'N39.0'
 'E03.9' 'L70.0' 'E11.65' 'E78.2' 'J02.0' 'J30.9' 'J11.1' 'F41.9' 'J01.00'
 'K21.9' 'M54.2']

+--------+---------+-------------------------------------------------------------+
| code   |   count | code_desc                                                   |
|--------+---------+-------------------------------------------------------------|
| I10    |    3766 | Essential (primary) hypertension                            |
| J06.9  |    2795 | Acute upper respiratory infection, unspecified              |
| J02.9  |    2257 | Acute pharyngitis, unspecified                              |
| E11.9  |    1446 | Type 2 diabetes mellitus without complications              |
| M54.5  |    1282 | Low back pain                                               |
| J01.90 |    1252 | Acute sinusitis, unspecified                                |
| J20.9  |    1076 | Acute bronchitis, unspecified                               |
| E78.5  |    1041 | Hyperlipidemia, unspecified                                 |
| N39.0  |     954 | Urinary tract infection, site not specified                 |
| E03.9  |     887 | Hypothyroidism, unspecified                                 |
| L70.0  |     883 | Acne vulgaris                                               |
| E11.65 |     784 | Type 2 diabetes mellitus with hyperglycemia                 |
| E78.2  |     732 | Mixed hyperlipidemia                                        |
| J02.0  |     717 | Streptococcal pharyngitis                                   |
| J30.9  |     691 | Allergic rhinitis, unspecified                              |
| J11.1  |     627 | Flu due to unidentified influenza virus w oth resp manifest |
| F41.9  |     618 | Anxiety disorder, unspecified                               |
| J01.00 |     597 | Acute maxillary sinusitis, unspecified                      |
| K21.9  |     591 | Gastro-esophageal reflux disease without esophagitis        |
| M54.2  |     588 | Cervicalgia                                                 |
+--------+---------+-------------------------------------------------------------+
> Least common conditions:
['H95.89' 'F19.24' 'D47.9' 'H02.004' 'F12.90' 'G47.11' 'M08.00' 'E34.4'
 'F10.29' 'G40.822' 'F19.17' 'I31.8' 'H35.30' 'M05.59' 'H68.101' 'M10.271'
 'H20.13' 'G40.804' 'B25.1' 'M21061']

+---------+---------+---------------------------------------------------------------+
| code    |   count | code_desc                                                     |
|---------+---------+---------------------------------------------------------------|
| H95.89  |       1 | Oth postproc comp and disorders of the ear/mastd, NEC         |
| F19.24  |       1 | Oth psychoactive substance dependence w mood disorder         |
| D47.9   |       1 | Neoplm of uncrt behav of lymphoid,hematpoetc & rel tiss,unsp  |
| H02.004 |       1 | Unspecified entropion of left upper eyelid                    |
| F12.90  |       1 | Cannabis use, unspecified, uncomplicated                      |
| G47.11  |       1 | Idiopathic hypersomnia with long sleep time                   |
| M08.00  |       1 | Unsp juvenile rheumatoid arthritis of unspecified site        |
| E34.4   |       1 | Constitutional tall stature                                   |
| F10.29  |       1 | Alcohol dependence with unspecified alcohol-induced disorder  |
| G40.822 |       1 | Epileptic spasms, not intractable, w/o status epilepticus     |
| F19.17  |       1 | Oth psychoactive substance abuse w persisting dementia        |
| I31.8   |       1 | Other specified diseases of pericardium                       |
| H35.30  |       1 | Unspecified macular degeneration                              |
| M05.59  |       1 | Rheumatoid polyneuropathy w rheumatoid arthritis mult site    |
| H68.101 |       1 | Unspecified obstruction of Eustachian tube, right ear         |
| M10.271 |       1 | Drug-induced gout, right ankle and foot                       |
| H20.13  |       1 | Chronic iridocyclitis, bilateral                              |
| G40.804 |       1 | Other epilepsy, intractable, without status epilepticus       |
| B25.1   |       1 | Cytomegaloviral hepatitis                                     |
| M21061  |       1 | n/a                                                           |
+---------+---------+---------------------------------------------------------------+
```

Out[17]:

```
(       code  count                                          code_desc
 15       I10   3766                   Essential (primary) hypertension
 24     J06.9   2795     Acute upper respiratory infection, unspecified
 33     J02.9   2257                     Acute pharyngitis, unspecified
 18     E11.9   1446     Type 2 diabetes mellitus without complications
 91     M54.5   1282                                      Low back pain
 73    J01.90   1252                       Acute sinusitis, unspecified
 136    J20.9   1076                      Acute bronchitis, unspecified
 108    E78.5   1041                        Hyperlipidemia, unspecified
 47     N39.0    954        Urinary tract infection, site not specified
 156    E03.9    887                        Hypothyroidism, unspecified
 102    L70.0    883                                      Acne vulgaris
 344   E11.65    784        Type 2 diabetes mellitus with hyperglycemia
 25     E78.2    732                               Mixed hyperlipidemia
 332    J02.0    717                          Streptococcal pharyngitis
 27     J30.9    691                     Allergic rhinitis, unspecified
 379    J11.1    627  Flu due to unidentified influenza virus w oth ...
 149    F41.9    618                      Anxiety disorder, unspecified
 158   J01.00    597             Acute maxillary sinusitis, unspecified
 394    K21.9    591  Gastro-esophageal reflux disease without esoph...
 370    M54.2    588                                        Cervicalgia,
          code  count                                          code_desc
 4861   H95.89      1  Oth postproc comp and disorders of the ear/mas...
 6965   F19.24      1  Oth psychoactive substance dependence w mood d...
 6966    D47.9      1  Neoplm of uncrt behav of lymphoid,hematpoetc &...
 6967  H02.004      1         Unspecified entropion of left upper eyelid
 6977   F12.90      1           Cannabis use, unspecified, uncomplicated
 6980   G47.11      1        Idiopathic hypersomnia with long sleep time
 6989   M08.00      1  Unsp juvenile rheumatoid arthritis of unspecif...
 6998    E34.4      1                        Constitutional tall stature
 7006   F10.29      1  Alcohol dependence with unspecified alcohol-in...
 7014  G40.822      1  Epileptic spasms, not intractable, w/o status ...
 7017   F19.17      1  Oth psychoactive substance abuse w persisting ...
 7023    I31.8      1            Other specified diseases of pericardium
 7027   H35.30      1                   Unspecified macular degeneration
 7030   M05.59      1  Rheumatoid polyneuropathy w rheumatoid arthrit...
 7031  H68.101      1  Unspecified obstruction of Eustachian tube, ri...
 7032  M10.271      1            Drug-induced gout, right ankle and foot
 7034   H20.13      1                   Chronic iridocyclitis, bilateral
 7038  G40.804      1  Other epilepsy, intractable, without status ep...
 7043    B25.1      1                          Cytomegaloviral hepatitis
 7047   M21061      1                                               n/a)
```

Analyzing prescription data

- There are 604 drug classes, 415 drug groups and 89 drug categories.
- Similar to how the diagnosis progression can be captured by the *d-sequence*, we can also define the prescription sequence (or treatment pathway) as the **temporally ordered sequence of prescriptions (or simply denoted as the p-sequence)**.
  - The analysis for the treatment/prescription data is similar to the diagnosis table give earlier except that drugs are not coded like ICD10 for diagnosis.
  - For the ease of analysis and modeling, we will encode the prescription based on the drug attribute with the most specificity, which is the `drug_class` (|drug_class|=604)
  - Given this, we can then define prescription sequence (or *p-sequence*) as the temporally ordered drug codes.
- To determine the most important prescription drugs for each patient, we can again apply TF-IDF over the p-sequences to heuristically determine the importance weight for each drug prescription in each patient historical record.
  - Doing so could help us potentially reduce the number of prescription variables when building statistical models to predict the health status.
  - It's helpful sometimes to consider reducing the number of variables because we may not have sufficient sample sizes in some cases to train a reasonably accurate model with 604 variables.
- For the purpose of this demo, I'll build baseline models (e.g. logistic regression) for a few ICD10 codes (n=10) selected from the most common primary diagnoses, where 'primary' is defined by couting from within the top-n ICD10 codes (n=3) used to represent the health status for each patient. See `utils.demo_topn_diagnoses()` for more details.
  - In this demo, I will not choose non-disease-specific ICD10 codes (i.e. most ICD10 codes alphanumerically greater than O00) as targets; intuitively, prescriptions are more correlated to disease-related diagnoses rather than, say, lab tests and misc medical procedures but of course, this hypothesis needs to be verified.
  - But in general, we could build a predictive model for all primary diagnoses.
  - We can then observe the predictive performances (in F1 score) in these cases.

In [1]:

```python
from data_pipeline import analyze_treatment
analyze_treatment()
```

```
(1.1) Found n=89 drug categories, and their counts are as follows:
Antidepressants                                 92834
Antihypertensives                               62022
Antihyperlipidemics                             58264
Contraceptives                                  51732
Antidiabetics                                   49251
Dermatologicals                                 40185
Anticonvulsants                                 37644
Analgesics - Opioid                             34163
Thyroid Agents                                  31785
Antiasthmatic and Bronchodilator Agents         30977
Analgesics - Anti-Inflammatory                  29641
Adhd/Anti-narcolepsy/Anti-obesity/Anorexiants   28434
Antianxiety Agents                              28018
Beta Blockers                                   27910
Ulcer Drugs                                     27055
Penicillins                                     25007
Antivirals                                      24577
Calcium Channel Blockers                        20434
Diuretics                                       20089
Corticosteroids                                 18398
Name: drug_category, dtype: int64

(1.2) Found n=415 drug groups, and their counts are as follows:
Hmg CoA Reductase Inhibitors                      50639
Selective Serotonin Reuptake Inhibitors (SSRIs)   47576
Combination Contraceptives - Oral                 45042
Thyroid Hormones                                  31138
Nonsteroidal Anti-Inflammatory Agents (NSAIDs)    26995
Anticonvulsants - Misc.                           24836
Ace Inhibitors                                    24464
Benzodiazepines                                   23120
Amphetamines                                      21911
Biguanides                                        21436
Calcium Channel Blockers                          20434
Opioid Combinations                               20302
Beta Blockers Cardio-Selective                    20204
Proton Pump Inhibitors                            19237
Glucocorticosteroids                              18238
Antidepressants - Misc.                           17404
Angiotensin Ii Receptor Antagonists               17278
Sympathomimetics                                  17226
Azithromycin                                      16729
Antihypertensive Combinations                     16367
Name: drug_group, dtype: int64

(1.3) Found n=604 drug classes, and their counts are as follows:
Hmg CoA Reductase Inhibitors                          50639
Selective Serotonin Reuptake Inhibitors (SSRIs)       47576
Thyroid Hormones                                      31138
Combination Contraceptives - Oral                     30632
Nonsteroidal Anti-Inflammatory Agents (NSAIDs)        25367
Anticonvulsants - Misc.                               24836
Ace Inhibitors                                        24464
Benzodiazepines                                       23120
Biguanides                                            21436
Calcium Channel Blockers                              20434
Beta Blockers Cardio-Selective                        20204
Proton Pump Inhibitors                                19237
Glucocorticosteroids                                  18238
Antidepressants - Misc.                               17404
Angiotensin Ii Receptor Antagonists                   17278
Amphetamine Mixtures                                  16978
Azithromycin                                          16729
Serotonin-Norepinephrine Reuptake Inhibitors (SNRIs)  15059
Aminopenicillins                                      14077
Nasal Steroids                                        14013
Name: drug_class, dtype: int64
```

Out[1]:

```
(        Patient_id Prescription_date                           drug_category  \
 0           P04447        2015-07-22    Genitourinary Agents - Miscellaneous
 1           P20764        2016-12-19    Genitourinary Agents - Miscellaneous
 2           P34917        2016-12-17    Genitourinary Agents - Miscellaneous
 3           P18892        2018-02-12    Genitourinary Agents - Miscellaneous
 4           P33131        2018-02-09    Genitourinary Agents - Miscellaneous
 ...            ...               ...                                     ...
 1024024     P09160        2016-08-21  Antiasthmatic and Bronchodilator Agents
 1024025     P08970        2016-07-09  Antiasthmatic and Bronchodilator Agents
 1024026     P16202        2017-04-15  Antiasthmatic and Bronchodilator Agents
 1024027     P06670        2016-04-21  Antiasthmatic and Bronchodilator Agents
 1024028     P53711        2018-04-12  Antiasthmatic and Bronchodilator Agents

                        drug_group                   drug_class drug_code
 0        Prostatic Hypertrophy Agents  5-Alpha Reductase Inhibitors       D-0
 1        Prostatic Hypertrophy Agents  5-Alpha Reductase Inhibitors       D-0
 2        Prostatic Hypertrophy Agents  5-Alpha Reductase Inhibitors       D-0
 3        Prostatic Hypertrophy Agents  5-Alpha Reductase Inhibitors       D-0
 4        Prostatic Hypertrophy Agents  5-Alpha Reductase Inhibitors       D-0
 ...                               ...                          ...       ...
 1024024                     Xanthines                    Xanthines     D-603
 1024025                     Xanthines                    Xanthines     D-603
 1024026                     Xanthines                    Xanthines     D-603
 1024027                     Xanthines                    Xanthines     D-603
 1024028                     Xanthines                    Xanthines     D-603

 [1024029 rows x 6 columns],
 {'5-Alpha Reductase Inhibitors': 'D-0',
  '5-HT3 Receptor Antagonists': 'D-1',
  '5-Lipoxygenase Inhibitors': 'D-2',
  'Ace Inhibitor & Calcium Channel Blocker Combinations': 'D-3',
  'Ace Inhibitors': 'D-4',
  'Ace Inhibitors & Thiazide/Thiazide-Like': 'D-5',
  'Acne Antibiotics': 'D-6',
  'Acne Combinations': 'D-7',
  'Acne Products': 'D-8',
  'Adhd Agent - Selective Alpha Adrenergic Agonists': 'D-9',
  'Adhd Agent - Selective Norepinephrine Reuptake Inhibitor': 'D-10',
  'Adrenergic Combinations': 'D-11',
  'Agents for External Genital and Perianal Warts': 'D-12',
  'Agents for Facial Wrinkles - Retinoids': 'D-13',
  'Agents for Gaucher Disease': 'D-14',
  'Agents for Pheochromocytoma': 'D-15',
  'Alcohol Deterrents': 'D-16',
  'Alkylating Agents': 'D-17',
  'Alpha 1-Adrenoceptor Antagonists': 'D-18',
  'Alpha Adrenergic Agonist & Carbonic Anhydrase Inhib Comb': 'D-19',
  'Alpha-2 Receptor Antagonists (Tetracyclics)': 'D-20',
  'Alpha-Beta Blockers': 'D-21',
  'Alpha-Glucosidase Inhibitors': 'D-22',
  'Alpha-Proteinase Inhibitor (Human)': 'D-23',
  'Aminoglycosides': 'D-24',
  'Aminopenicillins': 'D-25',
  'Ampa Glutamate Receptor Antagonists': 'D-26',
  'Amphetamine Mixtures': 'D-27',
  'Amphetamines': 'D-28',
  'Analgesics Other': 'D-29',
  'Analgesics-Sedatives': 'D-30',
  'Anaphylaxis Therapy Agents': 'D-31',
  'Androgen Biosynthesis Inhibitors': 'D-32',
  'Androgens': 'D-33',
  'Anesthetics Topical Oral': 'D-34',
  'Anesthetics Topical Oral - Combinations': 'D-35',
  'Angiotensin Ii Receptor Ant-Ca Channel Blocker-Thiazides': 'D-36',
  'Angiotensin Ii Receptor Antag & Ca Channel Blocker Comb': 'D-37',
  'Angiotensin Ii Receptor Antag & Thiazide/Thiazide-Like': 'D-38',
  'Angiotensin Ii Receptor Antagonists': 'D-39',
  'Anorexiant Combinations': 'D-40',
  'Anorexiants Non-Amphetamine': 'D-41',
  'Anthelmintics': 'D-42',
  'Anti-Cataplectic Agents': 'D-43',
  'Anti-IgE Monoclonal Antibodies': 'D-44',
  'Anti-Infective Agents - Misc.': 'D-45',
  'Anti-Infective Genitourinary Irrigants': 'D-46',
  'Anti-Infective Misc. - Combinations': 'D-47',
  'Anti-Infectives - Throat': 'D-48',
  'Anti-Inflammatory Agents': 'D-49',
  'Anti-Inflammatory Agents - Topical': 'D-50',
  'Anti-Inflammatory Combinations - Topical': 'D-51',
  'Anti-Obesity - GLP-1 Receptor Agonists': 'D-52',
  'Anti-Obesity Agent Combinations': 'D-53',
  'Anti-TNF-Alpha - Monoclonal Antibodies': 'D-54',
  'Antiadrenergics - Centrally Acting': 'D-55',
  'Antiadrenergics - Peripherally Acting': 'D-56',
  'Antiandrogens': 'D-57',
  'Antianginals-Other': 'D-58',
  'Antianxiety Agents - Misc.': 'D-59',
  'Antiarrhythmics Type I-A': 'D-60',
  'Antiarrhythmics Type I-B': 'D-61',
  'Antiarrhythmics Type I-C': 'D-62',
  'Antiarrhythmics Type Iii': 'D-63',
  'Antibiotic Mixtures Topical': 'D-64',
  'Antibiotic Steroid Combinations - Topical': 'D-65',
  'Antibiotics - Topical': 'D-66',
  'Anticholinergic Combinations': 'D-67',
  'Anticonvulsants - Benzodiazepines': 'D-68',
  'Anticonvulsants - Misc.': 'D-69',
  'Antidepressants - Misc.': 'D-70',
  'Antidiarrheal/Probiotic Agents - Misc.': 'D-71',
  'Antidotes - Chelating Agents': 'D-72',
  'Antidotes and Specific Antagonists': 'D-73',
  'Antiemetic Combinations': 'D-74',
  'Antiemetics - Anticholinergic': 'D-75',
  'Antiemetics - Miscellaneous': 'D-76',
  'Antiestrogens': 'D-77',
  'Antifungals': 'D-78',
  'Antifungals - Topical': 'D-79',
  'Antifungals - Topical Combinations': 'D-80',
  'Antihemophilic Products': 'D-81',
  'Antihistamine-Steroid': 'D-82',
  'Antihistamines - Ethanolamines': 'D-83',
  'Antihistamines - Non-Sedating': 'D-84',
  'Antihistamines - Phenothiazines': 'D-85',
  'Antihistamines - Piperidines': 'D-86',
  'Antihyperlipidemics - Misc.': 'D-87',
  'Antihyperlipidemics Misc. Combinations': 'D-88',
  'Antimalarial Combinations': 'D-89',
  'Antimalarials': 'D-90',
  'Antimanic Agents': 'D-91',
  'Antimetabolites': 'D-92',
  'Antimyasthenic/Cholinergic Agents': 'D-93',
  'Antimycobacterial Agents': 'D-94',
  'Antineoplastic - Braf Kinase Inhibitors': 'D-95',
  'Antineoplastic - Immunomodulators': 'D-96',
  'Antineoplastic - Mek Inhibitors': 'D-97',
  'Antineoplastic - Monoclonal Antibodies': 'D-98',
  'Antineoplastic - Multikinase Inhibitors': 'D-99',
  'Antineoplastic - Proteasome Inhibitors': 'D-100',
  'Antineoplastic - Tyrosine Kinase Inhibitors': 'D-101',
  'Antineoplastic - mTOR Kinase Inhibitors': 'D-102',
  'Antineoplastic Antibiotics': 'D-103',
  'Antineoplastic Antimetabolites - Topical': 'D-104',
  'Antineoplastic Combinations': 'D-105',
  'Antineoplastic or Premalignant Lesions - Topical Misc.': 'D-106',
  "Antineoplastic or Premalignant Lesions - Topical NSAID's": 'D-107',
  'Antineoplastics Misc.': 'D-108',
  'Antiparkinson Anticholinergics': 'D-109',
  'Antiparkinson Dopaminergics': 'D-110',
  'Antiparkinson Monoamine Oxidase Inhibitors': 'D-111',
  'Antiperistaltic Agents': 'D-112',
  'Antiprotozoal Agents': 'D-113',
  'Antipruritics - Topical': 'D-114',
  'Antipsoriatics': 'D-115',
  'Antipsoriatics - Systemic': 'D-116',
  'Antipsychotics - Misc.': 'D-117',
  'Antiretroviral Combinations': 'D-118',
  'Antiretrovirals - CCR5 Antagonists (Entry Inhibitor)': 'D-119',
  'Antiretrovirals - Integrase Inhibitors': 'D-120',
  'Antiretrovirals - Protease Inhibitors': 'D-121',
  'Antiretrovirals - RTI-Non-Nucleoside Analogues': 'D-122',
  'Antiretrovirals - RTI-Nucleoside Analogues-Purines': 'D-123',
  'Antiretrovirals - RTI-Nucleoside Analogues-Pyrimidines': 'D-124',
  'Antiretrovirals - RTI-Nucleoside Analogues-Thymidines': 'D-125',
  'Antiretrovirals - RTI-Nucleotide Analogues': 'D-126',
  'Antirheumatic - Janus Kinase (Jak) Inhibitors': 'D-127',
  'Antirheumatic Antimetabolites': 'D-128',
  'Antiseborrheic Combinations': 'D-129',
  'Antiseborrheic Products': 'D-130',
  'Antiseptics - Mouth/Throat': 'D-131',
  'Antispasmodics': 'D-132',
  'Antithyroid Agents': 'D-133',
  'Antitussive - Nonnarcotic': 'D-134',
  'Antitussive - Opioid': 'D-135',
  'Antitussive-Expectorant': 'D-136',
  'Antitussive-Expectorants-Decongestant': 'D-137',
  'Antiviral Monoclonal Antibodies': 'D-138',
  'Antivirals - Topical': 'D-139',
  'Applicators,Cotton Balls,etc': 'D-140',
  'Aromatase Inhibitors': 'D-141',
  'Artificial Tear Inserts': 'D-142',
  'Artificial Tear and Lubricant Combinations': 'D-143',
  'Astringents': 'D-144',
  'Atopic Dermatitis - Monoclonal Antibodies': 'D-145',
  'Azithromycin': 'D-146',
  'B-Complex W/ C & Folic Acid': 'D-147',
  'B-Complex W/ C-Zn & Folic Acid': 'D-148',
  'Bacterial Vaccines': 'D-149',
  'Barbiturate Hypnotics': 'D-150',
  'Belladonna Alkaloids': 'D-151',
  'Benzathiazoles': 'D-152',
  'Benzisoxazoles': 'D-153',
  'Benzodiazepine Hypnotics': 'D-154',
  'Benzodiazepines': 'D-155',
  'Beta Adrenergics': 'D-156',
  'Beta Blocker & Diuretic Combinations': 'D-157',
  'Beta Blockers Cardio-Selective': 'D-158',
  'Beta Blockers Non-Selective': 'D-159',
  'Beta-Blockers - Ophthalmic': 'D-160',
  'Beta-Blockers - Ophthalmic Combinations': 'D-161',
  'Bicarbonates': 'D-162',
  'Biguanides': 'D-163',
  'Bile Acid Sequestrants': 'D-164',
  'Biphasic Contraceptives - Oral': 'D-165',
  'Bisphosphonates': 'D-166',
  'Bowel Evacuant Combinations': 'D-167',
  'Bradykinin B2 Receptor Antagonists': 'D-168',
  'Bronchodilators - Anticholinergics': 'D-169',
  "Bulk Chemicals - Ga's": 'D-170',
  "Bulk Chemicals - Hy's": 'D-171',
  "Bulk Chemicals - Me's": 'D-172',
  "Bulk Chemicals - Ri's": 'D-173',
  'Bulk Chemicals - Ta': 'D-174',
  'Burn Products': 'D-175',
  'Butyrophenones': 'D-176',
  'C1 Inhibitors': 'D-177',
  'CXCR4 Receptor Antagonist': 'D-178',
  'Calcimimetic Agents': 'D-179',
  'Calcitonins': 'D-180',
  'Calcium': 'D-181',
  'Calcium Channel Blocker & Hmg CoA Reductase Inhibit Comb': 'D-182',
  'Calcium Channel Blockers': 'D-183',
  'Calcium Combinations': 'D-184',
  'Carbamates': 'D-185',
  'Carbapenems': 'D-186',
  'Carbohydrates': 'D-187',
  'Carbonic Anhydrase Inhibitors': 'D-188',
  'Cardiac Glycosides': 'D-189',
  'Carnitine Replenisher - Agents': 'D-190',
  'Central Muscle Relaxants': 'D-191',
  'Cephalosporins - 1st Generation': 'D-192',
  'Cephalosporins - 2nd Generation': 'D-193',
  'Cephalosporins - 3rd Generation': 'D-194',
  'Cephalosporins - 4th Generation': 'D-195',
  'Chelating Agents': 'D-196',
  'Chemotherapy Adjuncts - Hyperuricemia Agents': 'D-197',
  'Chlorine Antiseptics': 'D-198',
  'Cholinomimetics - Ache Inhibitors': 'D-199',
  'Cic Agents - Guanylate Cyclase-C (Gc-C) Agonists': 'D-200',
  'Citrates': 'D-201',
  'Clarithromycin': 'D-202',
  'Cmv Agents': 'D-203',
  'Cobalamins': 'D-204',
  'Codeine Combinations': 'D-205',
  'Combination Contraceptives - Oral': 'D-206',
  'Combination Contraceptives - Transdermal': 'D-207',
  'Combination Contraceptives - Vaginal': 'D-208',
  'Condoms - Female': 'D-209',
  'Continuous Contraceptives - Oral': 'D-210',
  'Copper Contraceptives - Iud': 'D-211',
  'Corticosteroids - Topical': 'D-212',
  'Corticotropin': 'D-213',
  'Coumarin Anticoagulants': 'D-214',
  'Cyclin-Dependent Kinases (Cdk) Inhibitors': 'D-215',
  'Cyclooxygenase 2 (COX-2) Inhibitors': 'D-216',
  'Cycloplegic Mydriatics': 'D-217',
  'Cyclosporine Analogs': 'D-218',
  'Cystic Fibrosis Agent - Combinations': 'D-219',
  'Cytotoxic Agents': 'D-220',
  'DPP-4 Inhibitor-Thiazolidinedione Combinations': 'D-221',
  'Decarboxylase Inhibitors': 'D-222',
  'Decongestant & Antihistamine': 'D-223',
  'Dental Products - Combinations': 'D-224',
  'Depigmenting Agents': 'D-225',
  'Depigmenting Combinations': 'D-226',
  'Diabetic Other': 'D-227',
  'Diagnostic Drugs': 'D-228',
  'Diagnostic Tests': 'D-229',
  'Diaphragms': 'D-230',
  'Dibenzo-Oxepino Pyrroles': 'D-231',
  'Dibenzodiazepines': 'D-232',
  'Dibenzothiazepines': 'D-233',
  'Dibenzoxazepines': 'D-234',
  'Dietary Management Product Combinations': 'D-235',
  'Dietary Management Products': 'D-236',
  'Digestive Enzymes': 'D-237',
  'Dipeptidyl Peptidase-4 (DPP-4) Inhibitors': 'D-238',
  'Dipeptidyl Peptidase-4 Inhibitor-Biguanide Combinations': 'D-239',
  'Direct Factor Xa Inhibitors': 'D-240',
  'Direct Muscle Relaxants': 'D-241',
  'Direct Renin Inhibitors': 'D-242',
  'Direct Renin Inhibitors & Thiazide/Thiazide-Like Comb': 'D-243',
  'Direct-Acting P2Y12 Inhibitors': 'D-244',
  'Diuretic Combinations': 'D-245',
  'Dopamine Receptor Agonists': 'D-246',
  'Dopamine Receptor Agonists - Ergot Derivatives': 'D-247',
  'Dry Mouth Agents and Artificial Saliva': 'D-248',
  'Electrolytes & Dextrose': 'D-249',
  'Emergency Contraceptives': 'D-250',
  'Emollient/Keratolytic Agents': 'D-251',
  'Emollient/Keratolytic Combinations': 'D-252',
  'Emollients': 'D-253',
  'Enzymes': 'D-254',
  'Enzymes - Topical': 'D-255',
  'Ergot Combinations': 'D-256',
  'Erythromycins': 'D-257',
  'Erythropoiesis-Stimulating Agents (ESAs)': 'D-258',
  'Estrogen & Androgen': 'D-259',
  'Estrogen & Progestin': 'D-260',
  'Estrogen-Selective Estrogen Receptor Modulator Comb': 'D-261',
  'Estrogens': 'D-262',
  'Expectorants': 'D-263',
  'Extended-Cycle Contraceptives - Oral': 'D-264',
  'External Vehicle Ingredients': 'D-265',
  'Eyelid Cleansers & Lubricants': 'D-266',
  'Fabry Disease - Agents': 'D-267',
  'Fibric Acid Derivatives': 'D-268',
  'Fibromyalgia Agent - SNRIs': 'D-269',
  'Fidaxomicin': 'D-270',
  'Flavoring Agents': 'D-271',
  'Fluoride': 'D-272',
  'Fluoride Combinations': 'D-273',
  'Fluoride Dental Products': 'D-274',
  'Fluoroquinolones': 'D-275',
  'Folic Acid Antagonists Rescue Agents': 'D-276',
  'Folic Acid/Folate Combinations': 'D-277',
  'Folic Acid/Folates': 'D-278',
  'Four Phase Contraceptives - Oral': 'D-279',
  'Gaba Modulators': 'D-280',
  'Gallstone Solubilizing Agents': 'D-281',
  'Gastrointestinal Antiallergy Agents': 'D-282',
  'Gastrointestinal Chloride Channel Activators': 'D-283',
  'Gastrointestinal Stimulants': 'D-284',
  'Genitourinary Irrigants': 'D-285',
  'Glucagon-Like Peptide-2 (GLP-2) Analogs': 'D-286',
  'Glucocorticosteroids': 'D-287',
  'Glucose Monitoring Test Supplies': 'D-288',
  'GnRH/LHRH Antagonists': 'D-289',
  'Gonadotropin Releasing Hormone (GnRH) Antagonists': 'D-290',
  'Gout Agent Combinations': 'D-291',
  'Gout Agents': 'D-292',
  'Granulocyte Colony-Stimulating Factors (G-Csf)': 'D-293',
  'Granulocyte/Macrophage Colony-Stimulating Factor(GM-CSF)': 'D-294',
  'Growth Hormones': 'D-295',
  'H-2 Antagonists': 'D-296',
  'Hematorheologic Agents': 'D-297',
  'Hemostatics - Systemic': 'D-298',
  'Heparins and Heparinoid-Like Agents': 'D-299',
  'Hepatitis B Agents': 'D-300',
  'Hepatitis C Agent - Combinations': 'D-301',
  'Hepatitis C Agents': 'D-302',
  'Herpes Agents - Purine Analogues': 'D-303',
  'Herpes Agents - Thymidine Analogues': 'D-304',
  'Hmg CoA Reductase Inhibitors': 'D-305',
  'Human Insulin': 'D-306',
  'Hydantoins': 'D-307',
  'Hydrocodone Combinations': 'D-308',
  'Hydrolytic Enzymes': 'D-309',
  'Hyperparathyroid Treatment - Vitamin D Analogs': 'D-310',
  'Hypnotics - Tricyclic Agents': 'D-311',
  'Ibs Agent - Guanylate Cyclase-C (Gc-C) Agonists': 'D-312',
  'Ibs Agent - Mu-Opioid Receptor Agonists': 'D-313',
  'Ibs Agent - Selective 5-HT3 Receptor Antagonists': 'D-314',
  'Imidazole-Related Antifungals': 'D-315',
  'Imidazole-Related Antifungals - Topical': 'D-316',
  'Imidazoles': 'D-317',
  'Imidazotetrazines': 'D-318',
  'Immune Serums': 'D-319',
  'Immunomodulators Imidazoquinolinamines - Topical': 'D-320',
  'Immunomodulators for Myelodysplastic Syndromes': 'D-321',
  'Incretin Mimetic Agents (GLP-1 Receptor Agonists)': 'D-322',
  'Infant Foods': 'D-323',
  'Inflammatory Bowel Agents': 'D-324',
  'Influenza Agents': 'D-325',
  'Inosine Monophosphate Dehydrogenase Inhibitors': 'D-326',
  'Insulin Administration Supplies': 'D-327',
  'Insulin-Incretin Mimetic Combinations': 'D-328',
  'Integrin Receptor Antagonists': 'D-329',
  'Interleukin Antagonists': 'D-330',
  'Interleukin-1 Blockers': 'D-331',
  'Interleukin-1beta Blockers': 'D-332',
  'Interleukin-5 Antagonists (IgG1 Kappa)': 'D-333',
  'Interleukin-6 Receptor Inhibitors': 'D-334',
  'Interstitial Cystitis Agents': 'D-335',
  'Intest Cholest Absorp Inhib-HMG CoA Reductase Inhib Comb': 'D-336',
  'Intestinal Acidifiers': 'D-337',
  'Intestinal Cholesterol Absorption Inhibitors': 'D-338',
  'Intrarectal Steroids': 'D-339',
  'Iodine Antiseptics': 'D-340',
  'Iron': 'D-341',
  'Iron Combinations': 'D-342',
  'Iron W/ Vitamins': 'D-343',
  'Iron-B12-Folate': 'D-344',
  'Janus Associated Kinase (Jak) Inhibitors': 'D-345',
  'Keratolytic/Antimitotic Agents': 'D-346',
  'Laxatives - Miscellaneous': 'D-347',
  'Leprostatics': 'D-348',
  'Leukotriene Receptor Antagonists': 'D-349',
  'Levodopa Combinations': 'D-350',
  'Lhrh Analogs': 'D-351',
  'Lincosamides': 'D-352',
  'Lipase Inhibitors': 'D-353',
  'Local Anesthetics - Amides': 'D-354',
  'Local Anesthetics - Topical': 'D-355',
  'Loop Diuretics': 'D-356',
  'Low Molecular Weight Heparins': 'D-357',
  'Lymphocyte Function-Associated Antigen-1 (LFA-1) Antag': 'D-358',
  'Macrolide Immunosuppressants': 'D-359',
  'Macrolide Immunosuppressants - Topical': 'D-360',
  'Magnesium': 'D-361',
  'Meglitinide Analogues': 'D-362',
  'Migraine Products': 'D-363',
  'Migraine Products - NSAIDs': 'D-364',
  'Mineralocorticoids': 'D-365',
  'Miotics - Direct Acting': 'D-366',
  'Misc. Anti-Ulcer': 'D-367',
  'Misc. Dermatological Products': 'D-368',
  'Misc. Devices': 'D-369',
  'Misc. Nutritional Substances': 'D-370',
  'Misc. Respiratory Inhalants': 'D-371',
  'Misc. Topical': 'D-372',
  'Misc. Topical Combinations': 'D-373',
  'Mitotic Inhibitors': 'D-374',
  'Mixed Allergenic Extracts': 'D-375',
  'Monoamine Oxidase Inhibitors (MAOIs)': 'D-376',
  'Monobactams': 'D-377',
  'Ms Agents - Pyrimidine Synthesis Inhibitors': 'D-378',
  'Mucolytics': 'D-379',
  'Multiple Sclerosis Agents': 'D-380',
  'Multiple Sclerosis Agents - Interferons': 'D-381',
  'Multiple Sclerosis Agents - Monoclonal Antibodies': 'D-382',
  'Multiple Sclerosis Agents - Nrf2 Pathway Activators': 'D-383',
  'Multiple Sclerosis Agents - Potassium Channel Blockers': 'D-384',
  'Multiple Vitamins W/ Minerals': 'D-385',
  'Multiple Vitamins W/ Minerals & Calcium-Folic Acid': 'D-386',
  'Multiple Vitamins W/ Minerals & Folic Acid': 'D-387',
  'Multivitamins': 'D-388',
  'N-Methyl-D-Aspartate (Nmda) Receptor Antagonists': 'D-389',
  'Nasal Agents - Misc.': 'D-390',
  'Nasal Antibiotics': 'D-391',
  'Nasal Anticholinergics': 'D-392',
  'Nasal Antihistamines': 'D-393',
  'Nasal Steroids': 'D-394',
  'Natural Penicillins': 'D-395',
  'Nebulizers': 'D-396',
  'Needles & Syringes': 'D-397',
  'Neprilysin Inhib (ARNI)-Angiotensin Ii Recept Antag Comb': 'D-398',
  'Neuraminidase Inhibitors': 'D-399',
  'Neuromuscular Blocking Agent - Neurotoxins': 'D-400',
  'Niacinamide W/ Zinc-Copper & Folic Acid': 'D-401',
  'Nicotinic Acid Derivatives': 'D-402',
  'Nitrate & Vasodilator Combinations': 'D-403',
  'Nitrate Vasodilating Agents': 'D-404',
  'Nitrates': 'D-405',
  'Nitrogen Mustards': 'D-406',
  'Nitrosoureas': 'D-407',
  'Non-Benzodiazepine - GABA-Receptor Modulators': 'D-408',
  'Non-Narc Antitussive-Antihistamine': 'D-409',
  'Non-Narc Antitussive-Decongestant-Antihistamine': 'D-410',
  'Nonergoline Dopamine Receptor Agonists': 'D-411',
  'Nonsteroidal Anti-Inflammatory Agent Combinations': 'D-412',
  'Nonsteroidal Anti-Inflammatory Agents (NSAIDs)': 'D-413',
  'Nutritional Supplements': 'D-414',
  'Ophthalmic Anti-Infective Combinations': 'D-415',
  'Ophthalmic Antiallergic': 'D-416',
  'Ophthalmic Antibiotics': 'D-417',
  'Ophthalmic Antivirals': 'D-418',
  'Ophthalmic Carbonic Anhydrase Inhibitors': 'D-419',
  'Ophthalmic Decongestants': 'D-420',
  'Ophthalmic Immunomodulators': 'D-421',
  'Ophthalmic Local Anesthetics': 'D-422',
  'Ophthalmic Nonsteroidal Anti-Inflammatory Agents': 'D-423',
  'Ophthalmic Selective Alpha Adrenergic Agonists': 'D-424',
  'Ophthalmic Steroid Combinations': 'D-425',
  'Ophthalmic Steroids': 'D-426',
  'Ophthalmic Sulfonamides': 'D-427',
```

```
 'Opioid Agonists': 'D-428',
 'Opioid Antagonists': 'D-429',
 'Opioid Antitussive-Antihistamine': 'D-430',
 'Opioid Antitussive-Decongestant-Antihistamine': 'D-431',
 'Opioid Combinations': 'D-432',
 'Opioid Partial Agonists': 'D-433',
 'Oral Vehicles': 'D-434',
 'Orexin Receptor Antagonists': 'D-435',
 'Ornithine Decarboxylase (Odc) Inhibitors - Topical': 'D-436',
 'Otic Agents - Miscellaneous': 'D-437',
 'Otic Analgesic Combinations': 'D-438',
 'Otic Anti-Infectives': 'D-439',
 'Otic Steroid-Anti-Infective Combinations': 'D-440',
 'Otic Steroids': 'D-441',
 'Ovulation Stimulants-Gonadotropins': 'D-442',
 'Ovulation Stimulants-Synthetic': 'D-443',
 'Oxaborole-Related Antifungals - Topical': 'D-444',
 'Oxazolidinones': 'D-445',
 'Oxytocics': 'D-446',
 'PCSK9 Inhibitors': 'D-447',
 'Parathyroid Hormone and Derivatives': 'D-448',
 'Parenteral Vehicles': 'D-449',
 'Ped Multi Vitamins W/Fl & Fe': 'D-450',
 'Ped Mv W/ Fluoride': 'D-451',
 'Ped Vitamins Acd Fluoride & Iron': 'D-452',
 'Ped Vitamins Acd W/ Fluoride': 'D-453',
 'Penicillin Combinations': 'D-454',
 'Penicillinase-Resistant Penicillins': 'D-455',
 'Periodontal Anti-Infectives': 'D-456',
 'Peripheral Opioid Receptor Antagonists': 'D-457',
 'Phenothiazines': 'D-458',
 'Phenylketonuria Treatment - Agents': 'D-459',
 'Phosphate': 'D-460',
 'Phosphate Binder Agents': 'D-461',
 'Phosphodiesterase 4 (PDE4) Inhibitors': 'D-462',
 'Phosphodiesterase 4 (PDE4) Inhibitors - Topical': 'D-463',
 'Phosphodiesterase Iii Inhibitors': 'D-464',
 'Platelet Aggregation Inhibitor Combinations': 'D-465',
 'Platelet Aggregation Inhibitors': 'D-466',
 'Poly (ADP-ribose) Polymerase (Parp) Inhibitors': 'D-467',
 'Postherpetic Neuralgia (PHN)/Neuropathic Pain Agents': 'D-468',
 'Potassium': 'D-469',
 'Potassium Removing Agents': 'D-470',
 'Potassium Sparing Diuretics': 'D-471',
 'Prenatal Mv & Min W/FE-FA': 'D-472',
 'Prenatal Mv & Min W/FE-FA-CA-Omega 3 Fish Oil': 'D-473',
 'Prenatal Mv & Min W/FE-FA-DHA': 'D-474',
 'Progestin Contraceptives - Implants': 'D-475',
 'Progestin Contraceptives - Injectable': 'D-476',
 'Progestin Contraceptives - Iud': 'D-477',
 'Progestin Contraceptives - Oral': 'D-478',
 'Progestins': 'D-479',
 'Progestins-Antineoplastic': 'D-480',
 'Prostaglandin - Impotence Agents': 'D-481',
 'Prostaglandin Vasodilators': 'D-482',
 'Prostaglandins - Ophthalmic': 'D-483',
 'Prostaglandins - Topical': 'D-484',
 'Prostatic Hypertrophy Agent Combinations': 'D-485',
 'Protease-Activated Receptor-1 (PAR-1) Antagonists': 'D-486',
 'Protectants - Mouth/Throat': 'D-487',
 'Proton Pump Inhibitor-Antacid Combinations': 'D-488',
 'Proton Pump Inhibitors': 'D-489',
 'Pseudobulbar Affect Agent Combinations': 'D-490',
 'Psychotherapeutic and Neurological Agents - Misc.': 'D-491',
 'Pulmonary Fibrosis Agents': 'D-492',
 'Pulmonary Hypertension - Endothelin Receptor Antagonists': 'D-493',
 'Pulmonary Hypertension - Phosphodiesterase Inhibitors': 'D-494',
 'Pulmonary Hypertension - Prostacyclin Receptor Agonist': 'D-495',
 'Purine Analogs': 'D-496',
 'Pyrimidine Synthesis Inhibitors': 'D-497',
 'Quaternary Anticholinergics': 'D-498',
 'Quinazoline Agents': 'D-499',
 'Quinolinone Derivatives': 'D-500',
 'Rank Ligand (Rankl) Inhibitors': 'D-501',
 'Rectal Anesthetic/Steroids': 'D-502',
 'Rectal Steroids': 'D-503',
 'Restless Leg Syndrome (Rls) Agents': 'D-504',
 'Rosacea Agents': 'D-505',
 'SGLT2 Inhibitor - DPP-4 Inhibitor Combinations': 'D-506',
 'Salicylates': 'D-507',
 'Saline Laxative Mixtures': 'D-508',
 'Saliva Stimulants': 'D-509',
 'Scabicides & Pediculicides': 'D-510',
 'Scar Treatment Products': 'D-511',
 'Selective Aldosterone Receptor Antagonists (SARAs)': 'D-512',
 'Selective Costimulation Modulators': 'D-513',
 'Selective Estrogen Receptor Modulators (SERMs)': 'D-514',
 'Selective Melatonin Receptor Agonists': 'D-515',
 'Selective Phosphodiesterase 4 (PDE4) Inhibitors': 'D-516',
 'Selective Serotonin Agonist-NSAID Combinations': 'D-517',
 'Selective Serotonin Agonists 5-HT(1)': 'D-518',
 'Selective Serotonin Reuptake Inhibitors (SSRIs)': 'D-519',
 'Selective cGMP Phosphodiesterase Type 5 Inhibitors': 'D-520',
 'Semi Solid Vehicles': 'D-521',
 'Serotonin 2C Receptor Agonists': 'D-522',
 'Serotonin Modulators': 'D-523',
 'Serotonin-Norepinephrine Reuptake Inhibitors (SNRIs)': 'D-524',
 'Sinus Node Inhibitors': 'D-525',
 'Skin Protectants': 'D-526',
 'Smoking Deterrents': 'D-527',
 'Sodium': 'D-528',
 'Sodium-Glucose Co-Transporter 2 (SGLT2) Inhibitors': 'D-529',
 'Sodium-Glucose Co-Transporter 2 Inhibitor-Biguanide Comb': 'D-530',
 'Soluble Tumor Necrosis Factor Receptor Agents': 'D-531',
 'Solvents': 'D-532',
 'Somatostatic Agents': 'D-533',
 'Spacer/Aerosol-Holding Chambers & Supplies': 'D-534',
 'Sphingosine 1-Phosphate (S1P) Receptor Modulators': 'D-535',
 'Steroid Inhalants': 'D-536',
 'Steroid-Local Anesthetic Combinations': 'D-537',
 'Steroids - Mouth/Throat': 'D-538',
 'Stimulants - Misc.': 'D-539',
 'Substance P/Neurokinin 1 (NK1) Receptor Antagonists': 'D-540',
 'Succinimides': 'D-541',
 'Sulfonylurea-Biguanide Combinations': 'D-542',
 'Sulfonylurea-Thiazolidinedione Combinations': 'D-543',
 'Sulfonylureas': 'D-544',
 'Synthetic Heparinoid-Like Agents': 'D-545',
 'Systemic Decongestants': 'D-546',
 'Tetracyclines': 'D-547',
 'Thiazides and Thiazide-Like Diuretics': 'D-548',
 'Thiazolidinedione-Biguanide Combinations': 'D-549',
 'Thiazolidinediones': 'D-550',
 'Thickening Agents': 'D-551',
 'Thienbenzodiazepines': 'D-552',
 'Thienbenzodiazepines & SSRIs': 'D-553',
 'Thienopyridine Derivatives': 'D-554',
 'Thioxanthenes': 'D-555',
 'Thrombin Inhibitors - Selective Direct & Reversible': 'D-556',
 'Thrombopoietin (Tpo) Receptor Agonists': 'D-557',
 'Thyroid Hormones': 'D-558',
 'Topical Anesthetic Combinations': 'D-559',
 'Topical Steroid Combinations': 'D-560',
 'Topoisomerase I Inhibitors': 'D-561',
 'Toxoid Combinations': 'D-562',
 'Tramadol Combinations': 'D-563',
 'Triazoles': 'D-564',
 'Tricyclic Agents': 'D-565',
 'Triphasic Contraceptives - Oral': 'D-566',
 'Tumor Necrosis Factor Alpha Blockers': 'D-567',
 'Type Ii 5-Alpha Reductase Inhibitors': 'D-568',
 'Ulcer Anti-Infective W/ Bismuth Combinations': 'D-569',
 'Ulcer Anti-Infective W/ Proton Pump Inhibitors': 'D-570',
 'Ulcer Drugs - Prostaglandins': 'D-571',
 'Urea Cycle Disorder - Agents': 'D-572',
 'Uricosurics': 'D-573',
 'Urinary Analgesics': 'D-574',
 'Urinary Anti-Infectives': 'D-575',
 'Urinary Antiseptic-Antispasmodic &/Or Analgesics': 'D-576',
 'Urinary Antispasmodic - Antimuscarinic (Anticholinergic)': 'D-577',
 'Urinary Antispasmodics - Beta-3 Adrenergic Agonists': 'D-578',
 'Urinary Antispasmodics - Cholinergic Agonists': 'D-579',
 'Urinary Antispasmodics - Direct Muscle Relaxants': 'D-580',
 'Urinary Tract Protective Agents': 'D-581',
 'Vaginal Anti-Infectives': 'D-582',
 'Vaginal Estrogens': 'D-583',
 'Vaginal Progestins': 'D-584',
 'Valproic Acid': 'D-585',
 'Vascular Endothelial Growth Factor (Vegf) Antagonists': 'D-586',
 'Vascular Endothelial Growth Factor (Vegf) Inhibitors': 'D-587',
 'Vasodilators': 'D-588',
 'Vasomotor Symptom Agents - SSRIs': 'D-589',
 'Vasopressin': 'D-590',
 'Vasopressors': 'D-591',
 'Viral Vaccine Combinations': 'D-592',
 'Viral Vaccines': 'D-593',
 'Viscosupplements': 'D-594',
 'Vitamin B-1': 'D-595',
 'Vitamin B-6': 'D-596',
 'Vitamin C': 'D-597',
 'Vitamin D': 'D-598',
 'Vitamin K': 'D-599',
 'Wound Care - Growth Factor Agents': 'D-600',
 'Wound Care Combinations': 'D-601',
 'Wound Dressings': 'D-602',
 'Xanthines': 'D-603'})
```

Encoding prescriptions

- Each `drug_class` is encoded with a prefix 'D' (by default), following by '-' and an index ranging from 0 to 603 (since there are 604 drug classes in total) i.e. D-0, D-1, ... , D-603

In [ ]:
```python
from utils import demo_encode_treatment
demo_encode_treatment()
```

Sequencing prescriptions

- Generating *p-sequences* helps us identify important prescriptions
- Use TF-IDF to idenify important prescriptions for each patients

- Once we know the most import prescriptions (say top 10) for each patient, we can then take the union of top prescriptions across patients to arrive at the final set of prescriptions, which is likely smaller than the total number of prescriptions. This could help reduce the number of variables when predicting health status.
  - We'll skip this step for the purpose of this demo by using all prescriptions as variables.

Observations

- From the sequencing process, we find that prescription drugs can be duplicated in the same session; this is unlike the case of ICD10, for which we'd expect no duplicates in the same session

In [ ]:
```python
# this takes some time!

from utils import demo_sequencing
demo_sequencing(dtype='treat', verbose=1)
```

Find the most important drug prescriptions

- Given the p-sequences, we can then conveniently use them as inputs to the TF-IDF model to find the most important prescriptions for each patient.
- Note that the sequenced representation is helpful as a format for training seq2seq model (i.e. using prescription sequences to predict likely diagnosis segments); however, we will not use this approach in this demo/prototype

In [2]:
```python
from utils import demo_topn_prescriptions
demo_topn_prescriptions()
```

```
[Treatment] Loading sequenced treatment/prescriptions file from:
/Users/pleiades/Documents/work/interview/Merck/Prescriptions-sequenced.csv

(eval_topn_features) Nd: 82546, dim(Xtr): (82546, 568), size(vocab): 568
... example feature names (n=568=?=568):
['D-0', 'D-1', 'D-10', 'D-100', 'D-101', 'D-102', 'D-103', 'D-104', 'D-105', 'D-106', 'D-107', 'D-108', 'D-1
09', 'D-11', 'D-110', 'D-111', 'D-112', 'D-113', 'D-114', 'D-115', 'D-116', 'D-117', 'D-118', 'D-119', 'D-1
2', 'D-120', 'D-121', 'D-122', 'D-123', 'D-124', 'D-126', 'D-127', 'D-128', 'D-129', 'D-13', 'D-130', 'D-13
1', 'D-132', 'D-133', 'D-134', 'D-135', 'D-136', 'D-138', 'D-139', 'D-14', 'D-140', 'D-141', 'D-142', 'D-14
4', 'D-145']
['D-596', 'D-597', 'D-598', 'D-599', 'D-6', 'D-60', 'D-600', 'D-601', 'D-602', 'D-603', 'D-61', 'D-62', 'D-6
3', 'D-64', 'D-65', 'D-66', 'D-67', 'D-68', 'D-69', 'D-7', 'D-70', 'D-71', 'D-74', 'D-75', 'D-76', 'D-77',
'D-78', 'D-79', 'D-8', 'D-80', 'D-81', 'D-82', 'D-83', 'D-84', 'D-85', 'D-86', 'D-87', 'D-88', 'D-89', 'D-
9', 'D-90', 'D-91', 'D-92', 'D-93', 'D-94', 'D-95', 'D-96', 'D-97', 'D-98', 'D-99']

--- doc #2043 ---
  feature  score
0   D-479    1.0
...... top N(5) features: D-479

--- doc #20867 ---
  feature     score
0   D-564  0.861767
1   D-582  0.383781
2   D-454  0.244778
3   D-593  0.223949
...... top N(5) features: D-564 D-582 D-454 D-593

--- doc #28749 ---
  feature     score
0    D-25  0.800866
1   D-194  0.598844
...... top N(5) features: D-25 D-194

--- doc #31410 ---
  feature  score
0    D-25    1.0
...... top N(5) features: D-25

--- doc #34270 ---
  feature     score
0   D-288  0.733510
1     D-4  0.489277
2   D-229  0.347129
3   D-163  0.251173
4   D-305  0.197445
...... top N(5) features: D-288 D-4 D-229 D-163 D-305

--- doc #44037 ---
  feature     score
0    D-27  0.993926
1   D-191  0.110053
...... top N(5) features: D-27 D-191

--- doc #46481 ---
  feature     score
0   D-544  0.779701
1   D-305  0.468223
2   D-163  0.297816
3     D-4  0.290069
...... top N(5) features: D-544 D-305 D-163 D-4

--- doc #60848 ---
  feature     score
0   D-131  0.793001
1   D-275  0.609220
...... top N(5) features: D-131 D-275

--- doc #66197 ---
  feature  score
0   D-593    1.0
...... top N(5) features: D-593

--- doc #73824 ---
  feature  score
0   D-454    1.0
...... top N(5) features: D-454

(eval_topn_features) top N features overall across all docs ...
... doc(mean):
  feature     score
0   D-519  0.048906
1   D-305  0.047891
2   D-206  0.046719
3   D-413  0.042440
4    D-25  0.042405
5   D-593  0.040971
6   D-146  0.039003
7   D-558  0.037068
8   D-287  0.032760
9     D-4  0.027681

> Most common primary prescritpions: D-305(Hmg CoA Reductase Inhibitors), n=7927
> Least common primary prescritpions: D-576(Urinary Antiseptic-Antispasmodic &/Or Analgesics), n=1
> There are n=560 prescritpion codes in total.
```

Using prescriptions to predict health status

Earlier we used the **top-n primary diagnoses (subject to the contraint of recency) to characterize a patient's health status**. In the demo, we set n=3 and n_days_lookback=180 (as a reminder, this means that we only consider the d-sequences at most 180 days long counting from the most recent visit).

- Certainly, the *n* in top-n, *window size*, *n_days_lookback* are all considered as adjustable hyperparameters to be optimized
  - For simplicity and as a baseline method, we could formulate the health-status predictive problem by predicting a single diagnosis code one at a time.
- In this manner, we will then build a model for each of the primary diagnosis associated with a patient's health status
- However, there are n=10678 models to train (see the following cell), some of which may not even have sufficient sample sizes for a regular statistical model
- Training a large set of models is usually not an issue, which can be put in a loop and trained off-line. But for the purpose of this demo, I'll only focus on 10 ICD10 codes selected from the most popular primary diagnoses to see if any skills that can be learned from the data using prescriptions.

- I'll briefly discuss alternative formulations shortly

In [3]:
```python
from utils import Diagnosis
import collections

dp = Diagnosis.properties
col_key = dp['id']
col_history = dp['history']
col_status = dp['status']

df_seq = Diagnosis.load(dtype='seq', verbose=True)
counter = collections.Counter()
for status in df_seq[col_status].values:
    counter.update(status.split())
print(f"> Found n={len(counter)} unique primary diagnosis codes.")
```

```
[Diagnosis] Loading sequenced diagnosis file from:
/Users/pleiades/Documents/work/interview/Merck/Diagnosis-sequenced.csv

> Found n=10678 unique primary diagnosis codes.
```

Model formulation

- Explanatory variables (or features): All the distinctive prescription drugs will be used as candidate explanatory variables (alternatively you could use top-n prescriptions as a screening mechanism to reduce the variable set; see module `utils`, in particular method `eval_topn_prescriptions()` of the `Treatment` class)
  - For convenience, prescription drugs are coded according to `drug_class` (D-1 ~ D-603)
  - Number of variables: 604
- Response variable (or label): In principle, all the primary diagnosis codes are potential targets; however, I'll demonstrate 10 target ICD10 codes for this demo
  - Training the model for the remaining ICD10 codes is a similar process; however, there are codes with very low sample sizes, for which we will then have to resort to either data augmentation or rule-based approach
  - From the data augmentation perspective, we could consider training **similar ICD10 codes** aggregately as a single model (e.g. all codes with the same first 3 digits); in the interest of time, I'll not address this in this demo
  - The notion of the **temporal alignment** mentioned earlier can be thought of as a rule-based approach. It's much more likely for any prescription drugs with dates close to the diagnosis date to be relevant and predictive; therefore, when the sample size is too low, this alignment rule is a good heurstic.

Matching prescriptions with a given diagnosis date

- The ideal values for W is a hyperparameter to be optimized or can be given by domain experts
- The `match` operation is defined in module `utils`, which identifies all the prescription drugs that fall within the scope of a given W associated with a diagnosis date

In [4]:
```python
from utils import demo_match

target_codes = ['J06.9', ]
W = (60, 60)
demo_match(target_codes, W, verbose=1)
```

```
(match) Total number of patients: 66608
(match) Number of patients with diagnosis J06.9: 5700
(match) Number of patients who had been given prescriptions for diagnosis J06.9: 5700
... n(diagnosed): 5700 >=? n(treated): 5700
/Users/pleiades/Documents/work/interview/Merck/utils.py:335: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df[col] = s.apply(lambda x: x[i])
(match) diagnosis table with the window of dates determined (W=(60, 60)):
Index(['Patient_id', 'Diag_date', 'ICD10', 'Diag_min', 'Diag_max'], dtype='object')

... sampled dataframe:
+-------------+-------------+---------+------------+------------+
| Patient_id  | Diag_date   | ICD10   | Diag_min   | Diag_max   |
|-------------+-------------+---------+------------+------------|
| P00014      | 2018-04-19  | J06.9   | 2018-02-18 | 2018-05-24 |
| P00037      | 2018-01-31  | J06.9   | 2017-12-02 | 2018-04-01 |
| P00054      | 2017-01-05  | J06.9   | 2016-11-06 | 2017-03-06 |
| P00054      | 2017-11-05  | J06.9   | 2017-09-06 | 2018-01-04 |
| P00056      | 2017-01-15  | J06.9   | 2016-11-16 | 2017-03-16 |
| P00075      | 2017-02-01  | J06.9   | 2016-12-03 | 2017-04-02 |
| P00075      | 2018-01-20  | J06.9   | 2017-11-21 | 2018-03-21 |
| P00096      | 2017-12-17  | J06.9   | 2017-10-18 | 2018-02-15 |
| P00109      | 2017-10-26  | J06.9   | 2017-08-27 | 2017-12-25 |
| P00117      | 2016-12-26  | J06.9   | 2016-10-27 | 2017-02-24 |
+-------------+-------------+---------+------------+------------+

(match) Found n=5194 patients with matching prescription records
```

Creating Training examples

- The making of training examples is a direct application of `match()` in module `utils`
- Creating positive examples for a given diagnosis (say J06.9) is straightforward; each patient constitutes a training example. Recall that we've set `n_days_lookback` = 180, which results in *d-sequences* to be at most 180 days long and within this time frame, we'll then loop through all the diagnosis dates, each of which is associated with a set of relevant prescriptions, qualified via W and the given diagnosis date (d). Note that it's possible that no prescriptions can be found at all given W and d.
- Creating negative examples is trickier and is perhaps an open question. What is the most reasonable control group for J06.9 (acute upper respiratory infection)? In principle, any diagnosis code that is not J06.9 can serve as negative examples because after all we are trying to predict 'yes' or 'no' for candidate cases of J06.9. **However**, ...
  - The set of all possible negative examples is too huge and not all of them are meaningful as part of a control group
  - If we include samples whose treatment pathways or prescription histories are too "different" than our target patients of J06.9, then very likely we will be getting a very trial model
  - We may want to incorporate substantial amounts of "similar patients" of varing degrees so that the resulting decision boundary is robust enough to distinguish nuances of minor prescription differences
  - As a trade-off, I'll include **a mixture of both similar and different cases, as negatives, in terms of the ICD hierarchy**. For instance, when predicting positive cases of J06.9, the classifier should learn to distinguish it from codes that **look like J06 (caveat?) but not J06.9 itself** -- as well as J0x and Jxx, i.e. the set of negative examples contain those cases associated with diagnoses of potentially very similar categories.
    - I put a question mark in the first scenario because perhaps that's too "harsh" for the classifier. An alternative way (perhaps a better way) is to exclude all J06 (regardless of its subcategories) because the distinction in subcategories is likely too subtle for prescriptions (a proxy for pharmacists and clinicians) to tell them apart.
    - Any codes that do not fall into the J00-J99 category are considered as substatially different (than our target J06.9), which are proportionally mixed into the control group as well.
- For the purpose of the demo, I did not intend to optimze the control group.

In [ ]:
```python
from utils import demo_make_classification

target_codes = ['J06.9', ] # ['I10', 'N39.0', 'E78.2', 'F41.9', 'K21.9', 'J20.9', 'M54.5', 'L70.0', 'J06.9',
demo_make_classification(target_codes=target_codes, W=(60, 60))
```

Model training and evaluating

- We'll use logistic regression with *l1* regularization (i.e. lasso) in this demo; any model class that deals with sparsity works relatively well in this case becasue a patient had likely only used a few drugs, a small ratio compared to the entire set of candidate drugs (see the output of `demo_make_classification()` for more details on this).
- In this demo, model selection is not addressed E.g. choosing optimial regularization strength, optimal W -- and technically also including length of the d-sequence (recency), and importance weighting schemes on diagnosis and prescriptions, among others.

In [8]:
```python
from sklearn.linear_model import LogisticRegression
from evaluate import demo_evaluate_health_status_prediction

classifier =  LogisticRegression(class_weight='balanced', penalty='l1', solver='saga')
target_codes = ['J06.9', ] # ['I10', 'N39.0', 'E78.2', 'F41.9', 'K21.9', 'J20.9', 'M54.5', 'L70.0',, 'E11.9'
n_folds = 5 # 5-fold cross validation

demo_evaluate_health_status_prediction(target_codes=target_codes, model=classifier, n_folds=n_folds)
```

```
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/utils/validation.py:760: DataConversionWarnin
g: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ),
for example using ravel().
  y = column_or_1d(y, warn=True)
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/_sag.py:330: ConvergenceWarning:
The max_iter was reached which means the coef_ did not converge
  "the coef_ did not converge", ConvergenceWarning)
(saveFig) Saving plot to:
plot/J069-precision_recall_curve.tif
... description: [output] precision recall curve
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/utils/validation.py:760: DataConversionWarnin
g: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ),
for example using ravel().
  y = column_or_1d(y, warn=True)
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/_sag.py:330: ConvergenceWarning:
The max_iter was reached which means the coef_ did not converge
  "the coef_ did not converge", ConvergenceWarning)
(saveFig) Saving plot to:
plot/J069-precision_recall_curve-1.tif
... description: [output] precision recall curve
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/utils/validation.py:760: DataConversionWarnin
g: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ),
for example using ravel().
  y = column_or_1d(y, warn=True)
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/_sag.py:330: ConvergenceWarning:
The max_iter was reached which means the coef_ did not converge
  "the coef_ did not converge", ConvergenceWarning)
/Users/pleiades/Documents/work/interview/Merck/evaluate.py:272: RuntimeWarning: invalid value encountered in
true_divide
  f1 = (1 + beta**2) * (precision * recall) / ((beta**2 * precision) + recall)
(saveFig) Saving plot to:
plot/J069-precision_recall_curve-2.tif
... description: [output] precision recall curve
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/utils/validation.py:760: DataConversionWarnin
g: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ),
for example using ravel().
  y = column_or_1d(y, warn=True)
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/_sag.py:330: ConvergenceWarning:
The max_iter was reached which means the coef_ did not converge
  "the coef_ did not converge", ConvergenceWarning)
/Users/pleiades/Documents/work/interview/Merck/evaluate.py:272: RuntimeWarning: invalid value encountered in
true_divide
  f1 = (1 + beta**2) * (precision * recall) / ((beta**2 * precision) + recall)
(saveFig) Saving plot to:
plot/J069-precision_recall_curve-3.tif
... description: [output] precision recall curve
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/utils/validation.py:760: DataConversionWarnin
g: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ),
for example using ravel().
  y = column_or_1d(y, warn=True)
/Users/pleiades/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/_sag.py:330: ConvergenceWarning:
The max_iter was reached which means the coef_ did not converge
  "the coef_ did not converge", ConvergenceWarning)
/Users/pleiades/Documents/work/interview/Merck/evaluate.py:272: RuntimeWarning: invalid value encountered in
true_divide
  f1 = (1 + beta**2) * (precision * recall) / ((beta**2 * precision) + recall)
(saveFig) Saving plot to:
plot/J069-precision_recall_curve-4.tif
... description: [output] precision recall curve
[plot] Saving ROC curve to:
plot/J069-roc

(saveFig) Saving plot to:
plot/J069-roc.tif
... description: [output] ROC curve
[demo] Fmax average: 0.5111916814747989, std: 0.0048845754081860236
[demo] Fold #[0]:
        [scores] -> {'AUROC': 0.6436326529040176, 'AUPRC': 0.41721365102750957, 'precision': 0.39308176100628
933, 'recall': 0.7218479307025987, 'accuracy': 0.5599148418491484}
[demo] Fold #[1]:
        [scores] -> {'AUROC': 0.641187335532721, 'AUPRC': 0.42111413689392063, 'precision': 0.388284085018144
1, 'recall': 0.7208854667949952, 'accuracy': 0.5529197080291971}
[demo] Fold #[2]:
        [scores] -> {'AUROC': 0.6353246079639289, 'AUPRC': 0.4112703181039957, 'precision': 0.381664175386148
5, 'recall': 0.737247353224254, 'accuracy': 0.5395377128953771}
[demo] Fold #[3]:
        [scores] -> {'AUROC': 0.6234821079714179, 'AUPRC': 0.40758124686122377, 'precision': 0.37359263050153
53, 'recall': 0.7025986525505293, 'accuracy': 0.5337591240875912}
[demo] Fold #[4]:
        [scores] -> {'AUROC': 0.6364341682723186, 'AUPRC': 0.4105583308290158, 'precision': 0.382056451612903
25, 'recall': 0.7302504816955684, 'accuracy': 0.541970802919708}
```

Other potential methods

- As a multilabel classification problem
  - We've used multiple (primary) diagnoses to characterize a patient's health status
  - It's possible that these primary diagnoses are correlated and therefore better to be characterized as a multilabel prediction problem. This is in contrary to what we've done so far, where we've treated each primary diagnosis as independent
- As a seq2seq classification problem
  - Prescriptions and their ordings are captured by p-sequences; similarly, primary diagnoses (as health status) can be ordered according to the diagnosis sequence as well
  - Note that the temporal ordering is operated on the level of sessions (or dates). Prescriptions and ICD10 codes within the same session (date) are usually not ordered.
- As a collaborative filtering problem
  - We could potentially treat prescriptions as "items" and diagnoses as "users" or any two entities that make sense.
  - A prescription is a match to a diagnosis if it's indeed used to treat that diagnosis or they tend to co-occur.
  - Similarly a prescription is not a match to a diagnosis, if they tend not to be related clinically
  - We can then form a "rating" matrix using prescriptions as rows and diagnoses as columns and the cells in the matrix represents "implicit feedback" (i.e. 1 represents preferred/matched whereas 0 represents not preferred/not matched)
  - This can be solved by matrix factorization-related methods using alternate least square for example.

In [ ]: