MedPack Documentation

Release 1.0

Michael Roa

April 29, 2018

Contents

Introduction

Figure 1: Based on openFDA's API

MedPack is a python script used to create and search a JSON file derived of all medications downloaded directly from the United States' OpenFDA.gov website. This does so without using their API is split into two parts. One part is the python script which can be called upon as a module. This script will allow the user to generate lists in python of medications fitting their search criteria. This is useful for a pipelined project. Included as well is a widget function included in a Jupyter-Notebook. This allows the user to explore the contents of the OpenFDA medications without the need for coding.

To run the MedPack Script, it is recommended to use Python version 3. The following libraries were used:

- fuzzywuzzy == 0.16.0
- pandas == 0.20.3
- requests == 2.18.4
- tqdm == 4.19.5
- wget == 3.2

These are also available in the requirements.txt

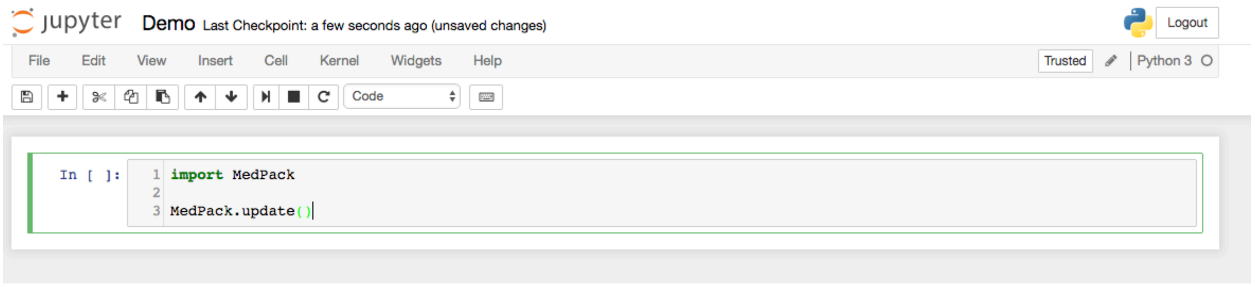
The installation of Anaconda is highly recommended. Anaconda is a free open source project by the Anaconda Inc. Upon installing it, Jupyter Notebook and several basic Python Data Science libraries wil be installed.

For more information on Anaconda, please see their official documentation:
https://media.readthedocs.org/pdf/anaconda-installer/latest/anaconda-installer.pdf

For more information on Jupyter Notebook, please see their official documentation:
https://media.readthedocs.org/pdf/jupyter-notebook/latest/jupyter-notebook.pdf

First Launch and MedPack.update()

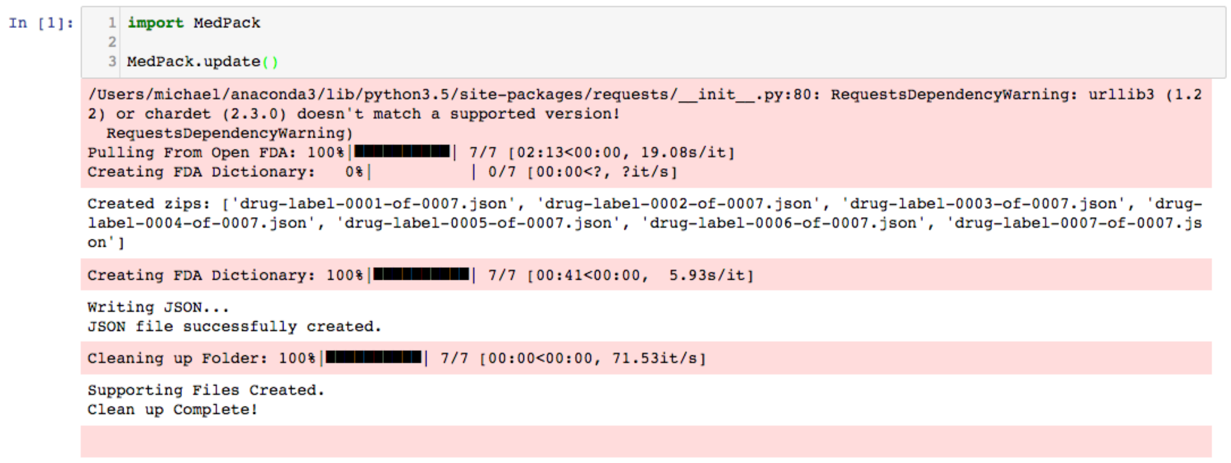
The most important part of this module is the update() function. Running this function is essential if you are beginning with the MedPack.py module. It will set up your environment for you.
Please keep in mind that all work done by the function will be in the same immediate directory. Therefore, please keep MedPack.py and your working Jupyter Notebook or Python folder in the same directory.
For this demonstration, we will be using a Jupyter Notebook:



```
In [ ]: import MedPack

        MedPack.update()
```

Beginning our new Jupyter-Notebook in a blank folder containing the MedPack.py module, we begin with the following commands shown above.

After hitting shift-enter to run this command we should see the output below (Depending on your internet speed and overall system performance, the time to run this may vary from roughly ten minutes):



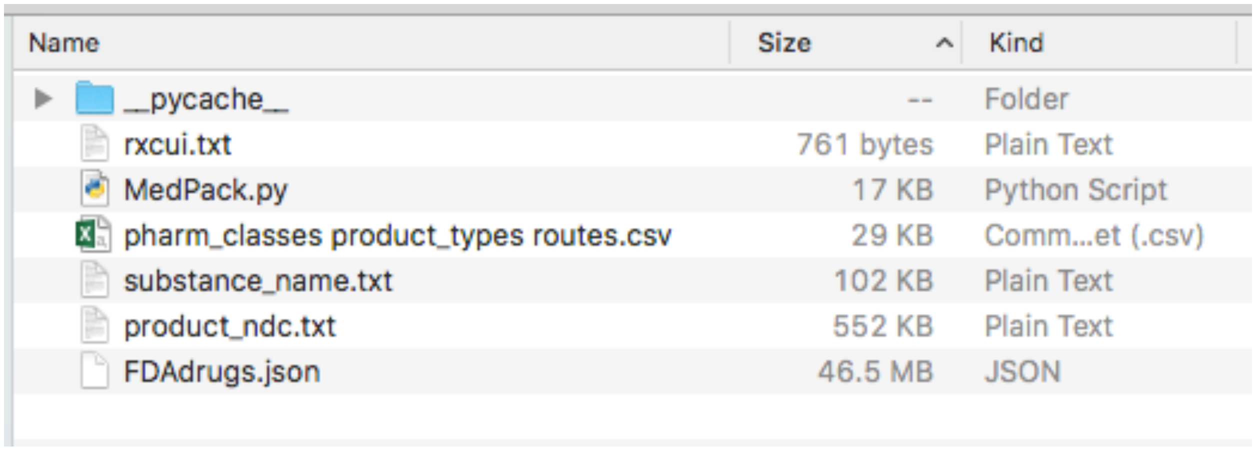
```
In [1]: import MedPack

        MedPack.update()

/Users/michael/anaconda3/lib/python3.5/site-packages/requests/__init__.py:80: RequestsDependencyWarning: urllib3 (1.2
2) or chardet (2.3.0) doesn't match a supported version!
  RequestsDependencyWarning)
Pulling From Open FDA: 100%|██████████| 7/7 [02:13<00:00, 19.08s/it]
Creating FDA Dictionary:   0%|          | 0/7 [00:00<?, ?it/s]

Created zips: ['drug-label-0001-of-0007.json', 'drug-label-0002-of-0007.json', 'drug-label-0003-of-0007.json', 'drug-
label-0004-of-0007.json', 'drug-label-0005-of-0007.json', 'drug-label-0006-of-0007.json', 'drug-label-0007-of-0007.js
on']

Creating FDA Dictionary: 100%|██████████| 7/7 [00:41<00:00,  5.93s/it]

Writing JSON...
JSON file successfully created.

Cleaning up Folder: 100%|██████████| 7/7 [00:00<00:00, 71.53it/s]

Supporting Files Created.
Clean up Complete!
```

If “Clean up Complete” is shown, your environment has successfully been set up in your folder.

During the running of the update() function, MedPack:

1. Calls upon the OpenFDA documentation for available downloads.
2. Creates downloads lists.
3. Downloads Zip files for each of the list.
4. Unzips the JSON files within the zip files and compiles them into one JSON file specific to MedPack.
5. Creates supporting documentation in the form of lists for each categorical variable of the medications.
6. Deletes the original JSON files and relevant zip files downloaded from OpenFDA

The files created from this function are shown here:



Name	Size	Kind
__pycache__	--	Folder
rxcui.txt	761 bytes	Plain Text
MedPack.py	17 KB	Python Script
pharm_classes product_types routes.csv	29 KB	Comm...et (.csv)
substance_name.txt	102 KB	Plain Text
product_ndc.txt	552 KB	Plain Text
FDAdrugs.json	46.5 MB	JSON

Supporting Documents:

- rxcui.txt
- pharm_classes product_types routes.csv
- substance_name.txt
- product_ncd.txt

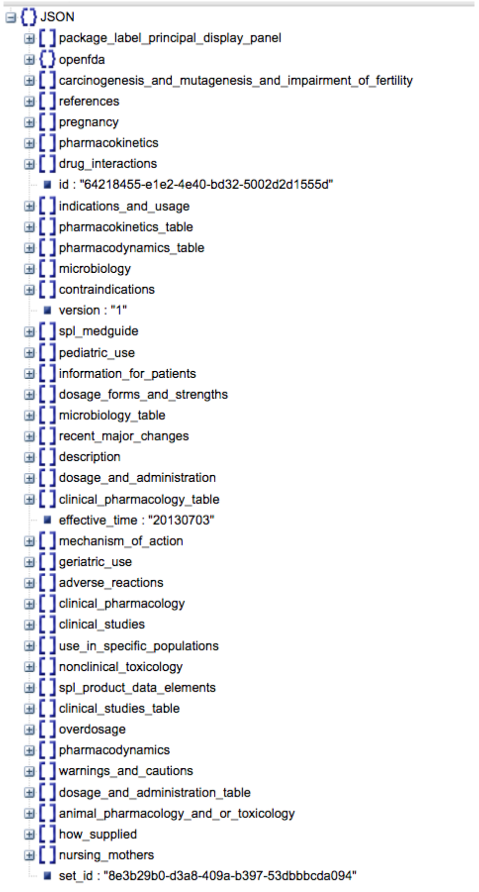
The supporting documents all list out all categorical variables from our FDAdrugs.json file. This file contains all the medication information that will be used later.

Structuring of Drugs JSON

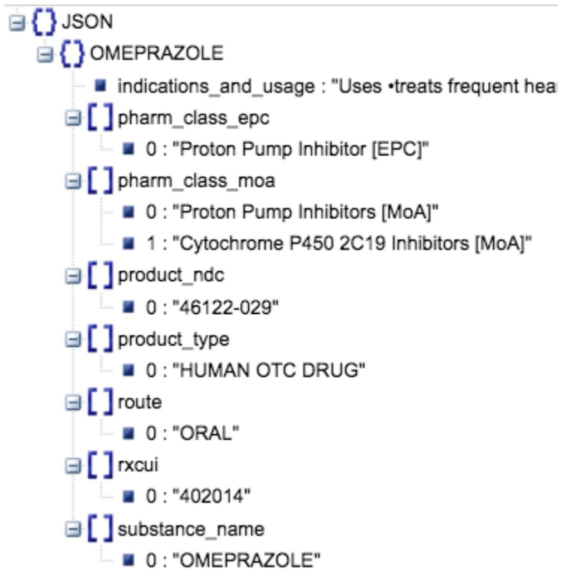
During the update() function, MedPack downloads and extracts the following from the original Open FDA data:

- indications_and_usage
- pharm_class_epc
- pharm_class_moa
- product_ndc
- product_type
- route
- rxcui
- substance_name

For a comparison of the original FDA structure and the structure for FDAdrugs.json, the original and new structure for the drug Omeprazole is shown below:



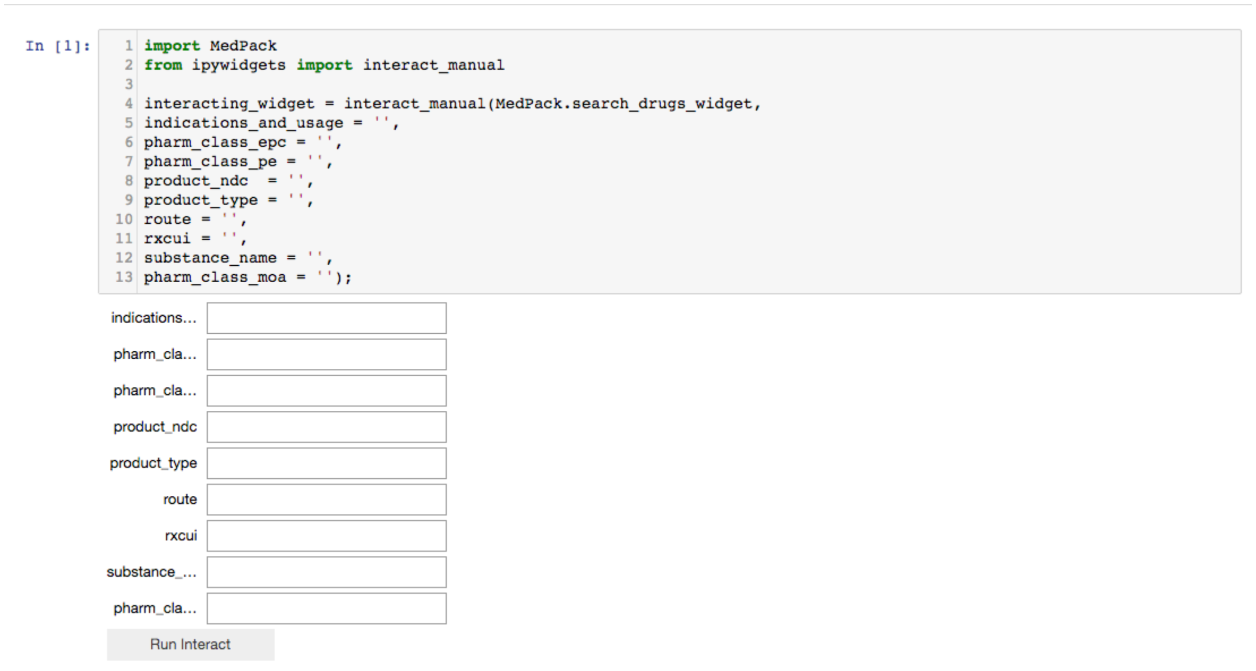
```
{ } JSON
  [ ] package_label_principal_display_panel
  { } openfda
  [ ] carcinogenesis_and_mutagenesis_and_impairment_of_fertility
  [ ] references
  [ ] pregnancy
  [ ] pharmacokinetics
  [ ] drug_interactions
      id : "64218455-e1e2-4e40-bd32-5002d2d1555d"
  [ ] indications_and_usage
  [ ] pharmacokinetics_table
  [ ] pharmacodynamics_table
  [ ] microbiology
  [ ] contraindications
      version : "1"
  [ ] spl_medguide
  [ ] pediatric_use
  [ ] information_for_patients
  [ ] dosage_forms_and_strengths
  [ ] microbiology_table
  [ ] recent_major_changes
  [ ] description
  [ ] dosage_and_administration
  [ ] clinical_pharmacology_table
      effective_time : "20130703"
  [ ] mechanism_of_action
  [ ] geriatric_use
  [ ] adverse_reactions
  [ ] clinical_pharmacology
  [ ] clinical_studies
  [ ] use_in_specific_populations
  [ ] nonclinical_toxicology
  [ ] spl_product_data_elements
  [ ] clinical_studies_table
  [ ] overdosage
  [ ] pharmacodynamics
  [ ] warnings_and_cautions
  [ ] dosage_and_administration_table
  [ ] animal_pharmacology_and_or_toxicology
  [ ] how_supplied
  [ ] nursing_mothers
      set_id : "8e3b29b0-d3a8-409a-b397-53dbbbcda094"
```



```
{ } JSON
  { } OMEPRAZOLE
      indications_and_usage : "Uses •treats frequent hea
    [ ] pharm_class_epc
        0 : "Proton Pump Inhibitor [EPC]"
    [ ] pharm_class_moa
        0 : "Proton Pump Inhibitors [MoA]"
        1 : "Cytochrome P450 2C19 Inhibitors [MoA]"
    [ ] product_ndc
        0 : "46122-029"
    [ ] product_type
        0 : "HUMAN OTC DRUG"
    [ ] route
        0 : "ORAL"
    [ ] rxcui
        0 : "402014"
    [ ] substance_name
        0 : "OMEPRAZOLE"
```

Jupyter Notebook Widgets

For simple code free browsing of the medications available, the Jupyter Notebook MedPackwidgets.ipynb was created. If you do not have this file, the code to produce the widgets is shown in the images. This must be run on a jupyter notebook in order to work.

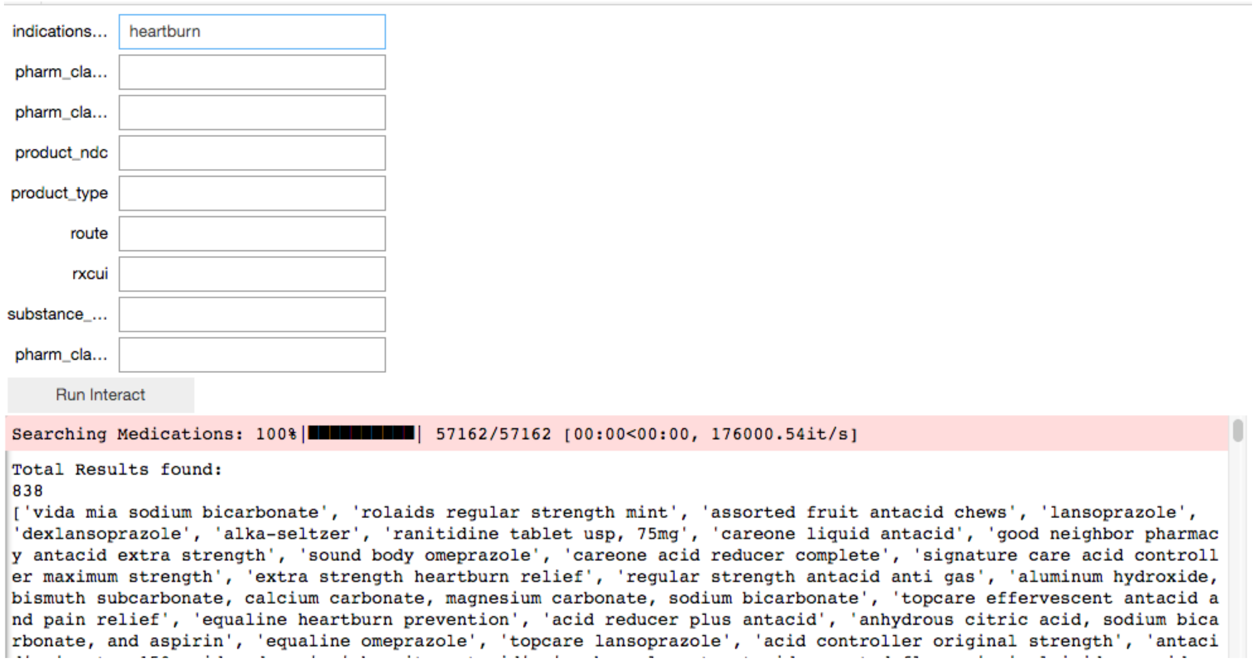


```
In [1]:
import MedPack
from ipywidgets import interact_manual

interacting_widget = interact_manual(MedPack.search_drugs_widget,
indications_and_usage = '',
pharm_class_epc = '',
pharm_class_pe = '',
product_ndc  = '',
product_type = '',
route = '',
rxcui = '',
substance_name = '',
pharm_class_moa = '');
```

Here is the basic search widget produced. The inputs for each are as follows:

- indications_and_usage
- pharm_class_epc
- pharm_class_pe
- product_ndc
- product_type
- route
- rxcui
- substance_name
- pharm_class_moa

Fill in text under any criteria and hit run interact. For this example, let's look for heartburn medications. This goes under the indications_and_usage section:

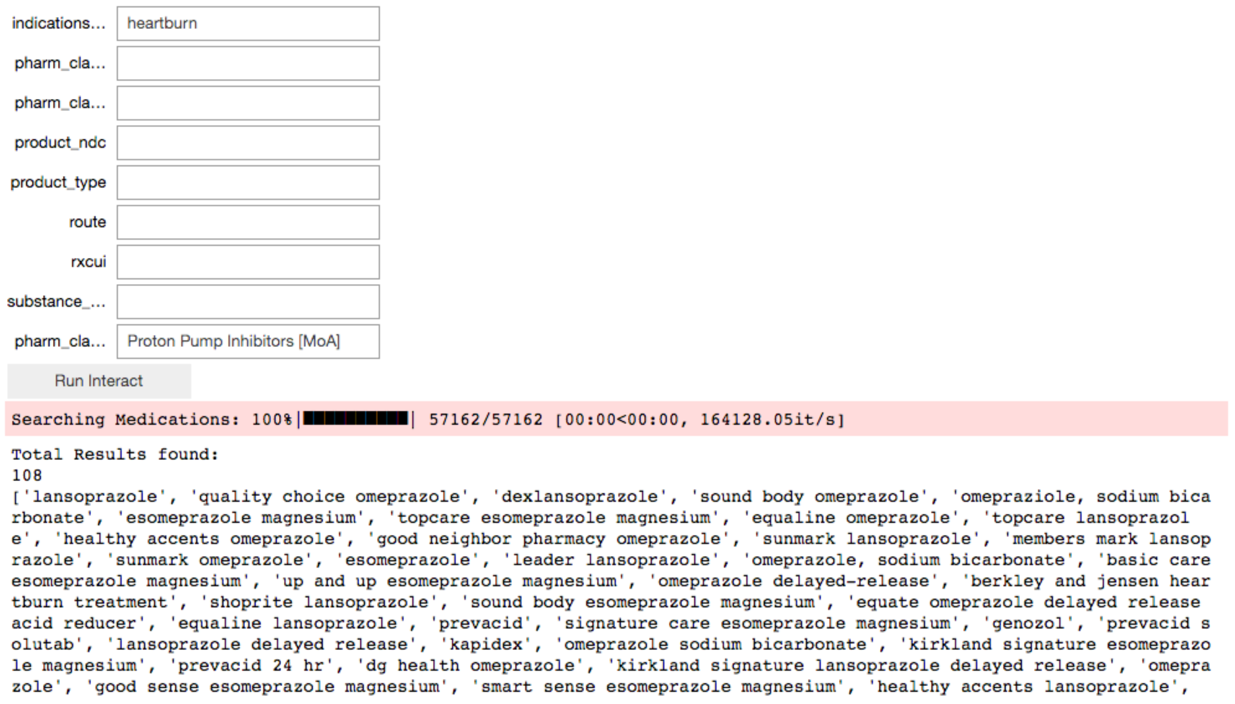


indications...	heartburn
pharm_cla...	
pharm_cla...	
product_ndc	
product_type	
route	
rxcui	
substance_...	
pharm_cla...	

```
Searching Medications: 100%|██████████| 57162/57162 [00:00<00:00, 176000.54it/s]
Total Results found:
838
['vida mia sodium bicarbonate', 'rolaids regular strength mint', 'assorted fruit antacid chews', 'lansoprazole',
'dexlansoprazole', 'alka-seltzer', 'ranitidine tablet usp, 75mg', 'careone liquid antacid', 'good neighbor pharmac
y antacid extra strength', 'sound body omeprazole', 'careone acid reducer complete', 'signature care acid controll
er maximum strength', 'extra strength heartburn relief', 'regular strength antacid anti gas', 'aluminum hydroxide,
bismuth subcarbonate, calcium carbonate, magnesium carbonate, sodium bicarbonate', 'topcare effervescent antacid a
nd pain relief', 'equaline heartburn prevention', 'acid reducer plus antacid', 'anhydrous citric acid, sodium bica
rbonate, and aspirin', 'equaline omeprazole', 'topcare lansoprazole', 'acid controller original strength', 'antaci
```

The output box displays the total results found. In this case it is 838, followed by a list of all the results.

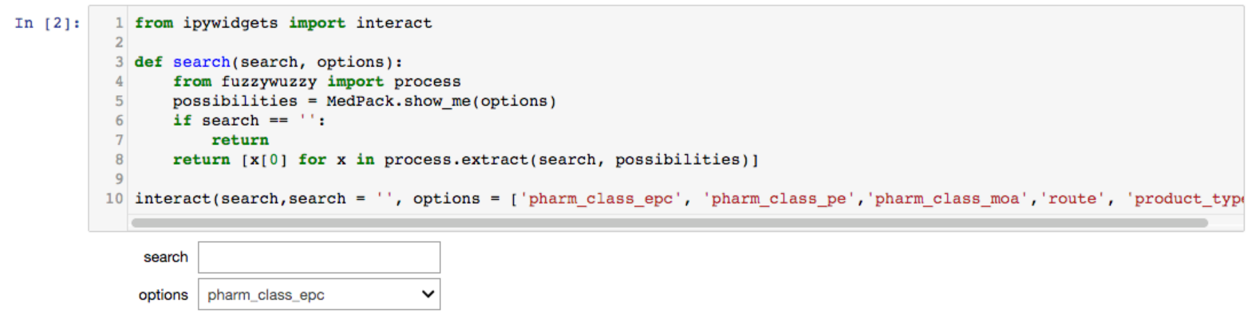
We can also be more precise in our search. Let's say we wish to find medications involving heartburn that are also proton pump inhibitors:



indications...	heartburn
pharm_cla...	
pharm_cla...	
product_ndc	
product_type	
route	
rxcui	
substance_...	
pharm_cla...	Proton Pump Inhibitors [MoA]

```
Searching Medications: 100%|██████████| 57162/57162 [00:00<00:00, 164128.05it/s]
Total Results found:
108
['lansoprazole', 'quality choice omeprazole', 'dexlansoprazole', 'sound body omeprazole', 'omepraziole, sodium bica
rbonate', 'esomeprazole magnesium', 'topcare esomeprazole magnesium', 'equaline omeprazole', 'topcare lansoprazol
e', 'healthy accents omeprazole', 'good neighbor pharmacy omeprazole', 'sunmark lansoprazole', 'members mark lansop
razole', 'sunmark omeprazole', 'esomeprazole', 'leader lansoprazole', 'omeprazole, sodium bicarbonate', 'basic care
esomeprazole magnesium', 'up and up esomeprazole magnesium', 'omeprazole delayed-release', 'berkley and jensen hear
tburn treatment', 'shoprite lansoprazole', 'sound body esomeprazole magnesium', 'equate omeprazole delayed release
acid reducer', 'equaline lansoprazole', 'prevacid', 'signature care esomeprazole magnesium', 'genozol', 'prevacid s
olutab', 'lansoprazole delayed release', 'kapidex', 'omeprazole sodium bicarbonate', 'kirkland signature esomeprazo
le magnesium', 'prevacid 24 hr', 'dg health omeprazole', 'kirkland signature lansoprazole delayed release', 'omepra
zole', 'good sense esomeprazole magnesium', 'smart sense esomeprazole magnesium', 'healthy accents lansoprazole',
```

If you notice the previous example, the input for a proton pump inhibitor was very specific. Each categorical variable varies widely and has very specific formatting related to FDA standards. In order to help navigate this, a second widget was created:

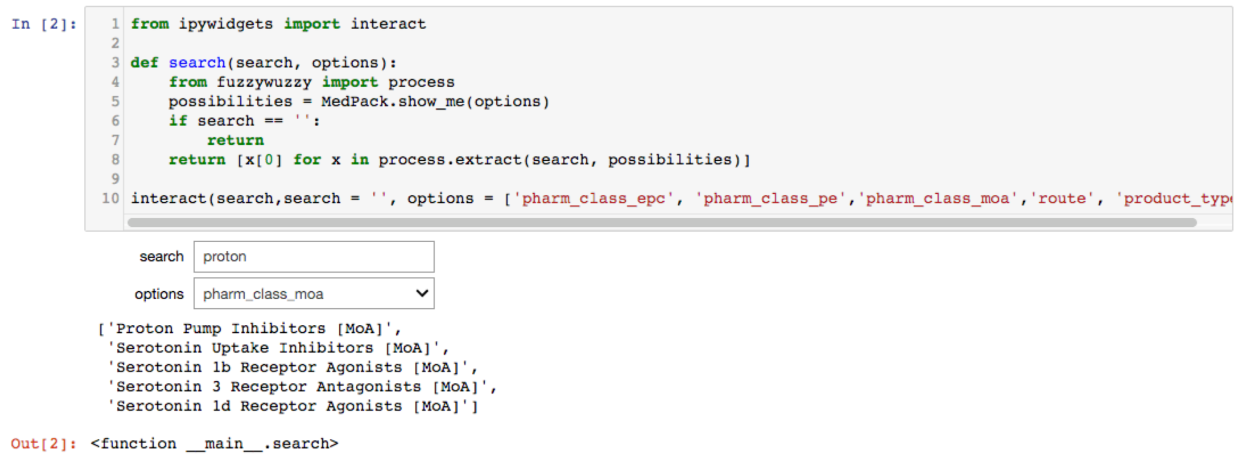


```
In [2]:
from ipywidgets import interact

def search(search, options):
    from fuzzywuzzy import process
    possibilities = MedPack.show_me(options)
    if search == '':
        return
    return [x[0] for x in process.extract(search, possibilities)]

interact(search,search = '', options = ['pharm_class_epc', 'pharm_class_pe','pharm_class_moa','route', 'product_typ
```

The options for this are:

- pharm_class_epc
- pharm_class_pe
- pharm_class_moa
- route
- product_type

From our previous example, we know we want to find proton pump inhibitors from our method of action but do not know what the exact name is under FDA standards. We select pharm_class_moa from our options and start typing into our search box. As soon as our search box has text, suggestions will begin to appear. From here we can just copy the exact name from this selection.



```
In [2]:
from ipywidgets import interact

def search(search, options):
    from fuzzywuzzy import process
    possibilities = MedPack.show_me(options)
    if search == '':
        return
    return [x[0] for x in process.extract(search, possibilities)]

interact(search,search = '', options = ['pharm_class_epc', 'pharm_class_pe','pharm_class_moa','route', 'product_typ
```

search: proton

options: pharm_class_moa

```
['Proton Pump Inhibitors [MoA]',
 'Serotonin Uptake Inhibitors [MoA]',
 'Serotonin 1b Receptor Agonists [MoA]',
 'Serotonin 3 Receptor Antagonists [MoA]',
 'Serotonin 1d Receptor Agonists [MoA]']

Out[2]: <function __main__.search>
```

Search Drugs Function

MedPack.py contains a search_drug function which is meant for integrating directly into a workflow.

The function for search drugs is structures as:
search_drugs(name *= 'no_name',* ***indications_and_usage*** *= None,* ***pharm_class_epc*** *= None,* ***pharm_class_pe*** *= None,* ***product_ndc*** *= None,* ***product_type*** *= None,* ***route*** *= None,* ***rxcui*** *= None,* ***substance_name*** *= None,* ***pharm_class_moa*** *= None, JSON = False, CSV = False,* ***pretty_list*** *= False):*

name : str, optional, default *'no_name'*
When outputting a file, the string entry becomes the file's name.

indications_and_usage : str, optional, default *None*
Entry searches under indications_and_usage parameter for items in the FDA drug database.

pharm_class_epc : str, optional, default *None*
Entry searches under pharm_class_epc parameter (Established Pharmacologic Class) for items in the FDA drug database.

pharm_class_pe : str, optional, default *None*
Entry searches under pharm_class_pe parameter (Physiologic Effect) for items in the FDA drug database.

product_ndc : str, optional, default *None*
Entry searches under product_ndc parameter (Product National Drug Code) for items in the FDA drug database.

product_type : str, optional, default *None*
Entry searches under product_type parameter (Human prescription drug or Human over the counter drug) for items in the FDA drug database.

route : str, optional, default *None*
Entry searches under route parameter (route of administration) for items in the FDA drug database.

rxcui : str, optional, default *None*

Entry searches under rxcui parameter (RxNorm concept unique identifer for the clinical drug or substance) for items in the FDA drug database.

substance_name : str, optional, default *None*

Entry searches under substance_name (Name of the main substance in the drug) parameter for items in the FDA drug database.

pharm_class_moa : str, optional, default *None*

Entry searches under pharm_class_moa (Drug's method of action) parameter for items in the FDA drug database.

JSON : str, optional, default *False*

When True, this tells the function to create a JSON file containing the results with the name under the *name* input.

CSV : bool, optional, default *False*

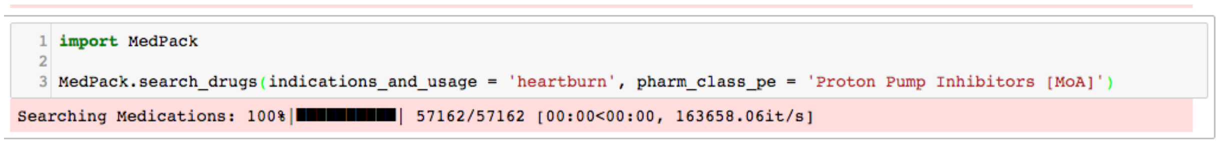
When True, this tells the function to create a CSV file containing the results with the name under the *name* input.

pretty_list : bool, optional, default *False*

When True, this takes the original return of the list function and returns only unique entries from the results.

Note that all of the entries must be string and in parenthesis ""

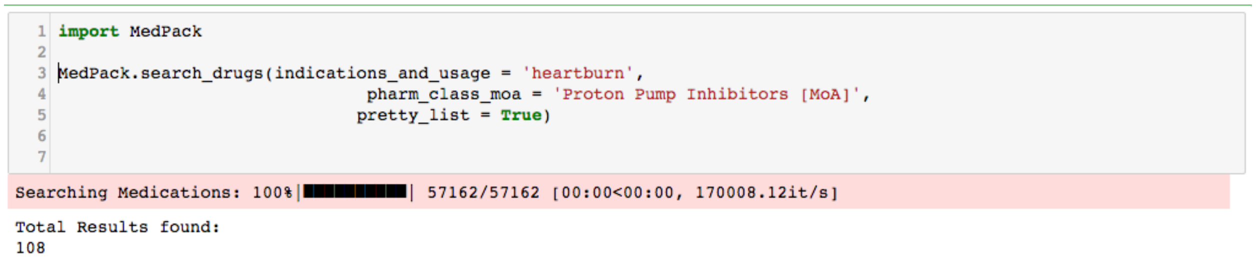
For an example of this, we will refer back to our previous search on the Jupyter Notebook widget. Our search consisted of *indications_and_usage = heartburn* and *pharm_class_moa = Proton Pump Inhibitors [MoA]*. In other words, we are looking for medications related to heartburn who act as proton pump inhibitors.



```
import MedPack

MedPack.search_drugs(indications_and_usage = 'heartburn', pharm_class_pe = 'Proton Pump Inhibitors [MoA]')
```

```
Searching Medications: 100%|██████████| 57162/57162 [00:00<00:00, 163658.06it/s]
```

Using our function, we have the above code. Notice the prompt showing that this is working. This particular code has no display. If we want to return a list of our results, we must enable pretty_list.

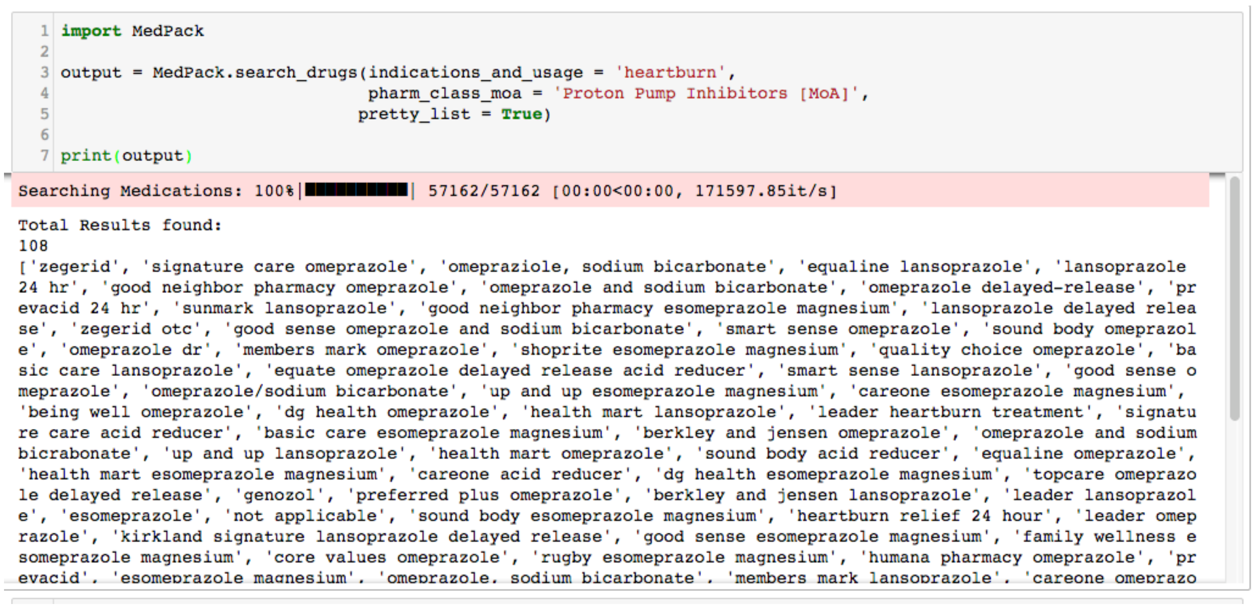


```
import MedPack

MedPack.search_drugs(indications_and_usage = 'heartburn',
                              pharm_class_moa = 'Proton Pump Inhibitors [MoA]',
                             pretty_list = True)


```

```
Searching Medications: 100%|██████████| 57162/57162 [00:00<00:00, 170008.12it/s]

Total Results found:
108
```

Now we can see that our search has resulted in 108 findings. Each of these findings are unique.

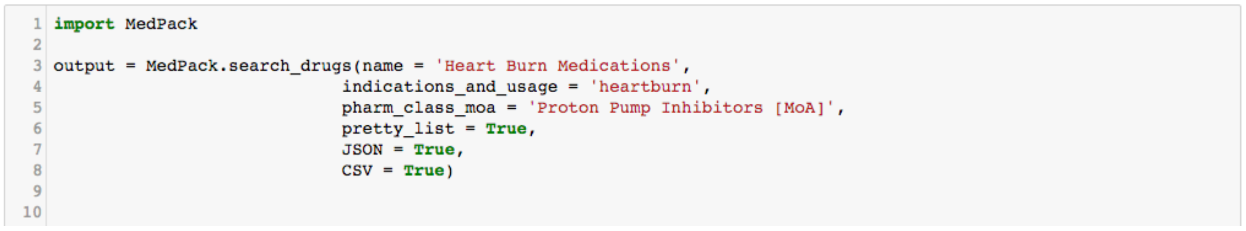
Since the return variable is a list, we can create a new object by giving the return list a name. In this case, we can give our results the name "output." To prove that output contains our results we print our output in the next line.



```
import MedPack

output = MedPack.search_drugs(indications_and_usage = 'heartburn',
                              pharm_class_moa = 'Proton Pump Inhibitors [MoA]',
                             pretty_list = True)

print(output)
```

```
Searching Medications: 100%|██████████| 57162/57162 [00:00<00:00, 171597.85it/s]

Total Results found:
108
['zegerid', 'signature care omeprazole', 'omepraziole, sodium bicarbonate', 'equaline lansoprazole', 'lansoprazole
24 hr', 'good neighbor pharmacy omeprazole', 'omeprazole and sodium bicarbonate', 'omeprazole delayed-release', 'pr
evacid 24 hr', 'sunmark lansoprazole', 'good neighbor pharmacy esomeprazole magnesium', 'lansoprazole delayed relea
se', 'zegerid otc', 'good sense omeprazole and sodium bicarbonate', 'smart sense omeprazole', 'sound body omeprazol
e', 'omeprazole dr', 'members mark omeprazole', 'shoprite esomeprazole magnesium', 'quality choice omeprazole', 'ba
sic care lansoprazole', 'equate omeprazole delayed release acid reducer', 'smart sense lansoprazole', 'good sense o
meprazole', 'omeprazole/sodium bicarbonate', 'up and up esomeprazole magnesium', 'careone esomeprazole magnesium',
'being well omeprazole', 'dg health omeprazole', 'health mart lansoprazole', 'leader heartburn treatment', 'signatu
re care acid reducer', 'basic care esomeprazole magnesium', 'berkley and jensen omeprazole', 'omeprazole and sodium
bicrabonate', 'up and up lansoprazole', 'health mart omeprazole', 'sound body acid reducer', 'equaline omeprazole',
'health mart esomeprazole magnesium', 'careone acid reducer', 'dg health esomeprazole magnesium', 'topcare omeprazo
le delayed release', 'genozol', 'preferred plus omeprazole', 'berkley and jensen lansoprazole', 'leader lansoprazol
e', 'esomeprazole', 'not applicable', 'sound body esomeprazole magnesium', 'heartburn relief 24 hour', 'leader omep
razole', 'kirkland signature lansoprazole delayed release', 'good sense esomeprazole magnesium', 'family wellness e
someprazole magnesium', 'core values omeprazole', 'rugby esomeprazole magnesium', 'humana pharmacy omeprazole', 'pr
evacid', 'esomeprazole magnesium', 'omeprazole, sodium bicarbonate', 'members mark lansoprazole', 'careone omeprazo
```

Here we can see that the output variable contains all our results for our search criteria. This can be extremely useful when data wrangling. For documentation of this list of results we can also create a JSON or CSV file containing all of these variables by typing them into our code with the value of True.



```python
import MedPack

output = MedPack.search_drugs(name = 'Heart Burn Medications',
                              indications_and_usage = 'heartburn',
                              pharm_class_moa = 'Proton Pump Inhibitors [MoA]',
                              pretty_list = True,
                              JSON = True,
                              CSV = True)
```

In this example, if we want to keep documentation on our Heartburn medications, we make a new JSON and CSV document both titled “Heart Burn Medications.” We can even add a date to the title in order to note when this query was set. It is also recommended to keep the search parameters and the FDAdrugs JSON file that was generated in order to keep complete documentation over what was used at the time.