**Lab 100 Knowledge Base**

[**Files and structure**](#_z8bnr5quo6so) **2**

[**Packages Used**](#_9oaki7360q4) **5**

[**Query format**](#_7m30m78ovr4d) **5**

[**Available API calls**](#_sgpmcpxwcaos) **6**

[**Bitbucket Link**](#_789vtrrlcojg) **8**

## Files and structure

* Patient\_normalization\_app.py
  + - Initializes flask app
* app/
  + Flask\_index.py
    - Contains methods for each api query to the kb, and supporting methods to parse data
* lib/
  + anthropometrics.py
    - Contains absi, sbsi, symmetry, waist to hip ratio, trunk to leg ratio calcs. Includes function to get absi data from NHANES
  + ACVSD.py
    - Calculates ACVSD
  + cognition.py
    - Calculates percentile, t\_score for cognition variables. Takes source from data/NIH/cognition\_standard\_score\_to\_tscore.csv by default
  + file.py
    - Helper file with IO functions
  + gen\_density\_plot.py
    - generate\_density\_plot(distribution = None, stats = None, min = 2.5, max = 97.5, number\_of\_points = 100, bandwidth = 'auto', method = 'sklearn')
    - Returns a set of XY coordinates with the distribution
    - If a distribution is provided, will smooth the distribution manually and make a graph of it. The smoothing I do works, but might not be as good as that done by other packages.
    - Else uses stats to generate a normal distribution from mean and standard deviation.
    - Only shows distributions between 2.275 and 97.725 percentile, but those numbers can be changed by altering up\_bound and lower\_bound
    - Number\_of\_points is the number of points in the output distribution. Default 100.
    - Bandwidth is automatically generated if bandwidth = “auto”, otherwise the user needs to provide it.
    - I tried two methods using both sklearn and scipy, but sklearn works significantly better.
  + helper.py
    - Utility functions - includes takeClosest which takes a list and a number, and finds the closest value in the list to the number in O(log n) time.
    - Also ordertest, which tests if a list is numeric and sorted
  + IPAQ.py
    - Returns IPAQ calcs from patient data
  + NHANES\_nutrition.pycreens
    - Reimplement nutrition scoring from the NHANES DSQ.
    - Includes function to calc NHANES scores from the NHANES dataset
  + NIH\_motor.py
    - Strength, Dexterity, and balance calcs.
    - Loads data from data/NIH/NIH\_Forumlas.csv
    - Loads tables from data/NIH/LookupTables/
  + normalizer.py
    - Takes filepath for NHANES, takes a list of target variables, and a list of variables to control for.
    - Iterates over NHANES csvs, and creates a dictionary with id numbers as keys, and target variables / normative variables as nested key value pairs. Pass over NHANES values that aren’t being used.
    - If there is NHANES data, it iterates over every demographic variable, starting with categorical variables, to create a subset of NHANES containing only subjects that are demographically identical/close close to the target.
      * Filters by matching categorical variable. (ex: Sex = female)
      * Filters by being close quantitatively (ex: 42 <= Age <= 47).
        + For more detail, see [Lab100kb Normalization Explanation](https://docs.google.com/document/d/1R-YbjQ9vpBTzKHhUzlkf-CWohusP8yCCurKS441mPug/edit?usp=sharing)
      * Returns patients that are demographically close/identical to patient.
    - If there are cutoffs associated with the target variable, list the cutoffs. Also, find which interpretation the target value fits into from the available cutoffs (which may differ based on demographic information). If there are no cutoffs, generic cutoffs are used based on standard deviations of the underlying distribution data if distribution data is available. Else, omit cutoffs/interpretation.
    - If there is NHANES distribution data, generate a density plot (see gen\_density\_plot.py), and generate summary statistics for the distribution including mean and standard deviation, and the percentile score of the target value.
    - If there is no NHANES distribution data, but there are static distribution variables (mean, sd) for the target variable (see static\_distributions.csv), then a normal distribution will be generated for the target value using said mean and standard deviation. Summary statistics and percentile scores will be generated using this distribution.
  + PHQ9.py
    - Patient health questionnaire calcs
  + PSQI.py
    - Pittsburgh Sleep Quality Index calcs
  + Sample\_post.py
    - Contains a sample post (as a dictionary) to send to the kb to test the calculated variables
  + Static\_vars.py
    - Initializes a number of static variables, including the list of variables (from variable\_reference.csv), the cutoffs (from cutpoints.csv), a list of the calculated variables, and a list of the static distributions (from static\_distributions.csv). The lists for static distributions and cutoffs include lists of functions to test which cutoff or distribution to use
  + Vitals.py
    - Calculates a couple lab stats (not used)
* data/
  + Static\_distributions.csv
    - A list of distribution stats for a select number of variables that don’t appear in NHANES. May vary by demogrpahics.
  + Variable\_reference.csv
    - A list of the variables used in lab100, and some detail. Includes both display and internal variable
  + Cutpoints.csv
    - A list of the cut offs for interpretation of data. Includes demographic conditions.
  + DSQ csvs - used for NHANES\_DSQ
    - Calib\_portion\_size.csv
    - Calib\_equation\_coeff.csv
    - calib\_DSQ\_cereal\_ntile.csv
    - Age\_groups.csv
    - DSQ\_calculated\_vars\_descriptions.csv
    - DSQ\_calculated\_vars.csv
    - DSQ\_coefficient\_to\_var.csv
    - DSQ\_NHANES\_var\_names.csv
    - max\_values\_DSQ.csv
  + NIH/
    - LookupTables/
      * Conversion tables for raw score to tscore by ethnicity and age.
    - Cognition\_standard\_score\_to\_tscore.csv
      * Interpretation for cognition scores
    - NIH\_Formulas.csv
      * Formulas extracted from NIH. Are executed programmatically in code.
* NHANES\_Downloader/
  + A program to scrape NHANES from the website.
* env/
  + Local python environment
* Patient\_normalization\_app.py
  + Run this to run flask app
* Scrape\_kb.py
  + Script to run through kb and check that things are working
* Upload\_nhanes\_to\_mysql.py
  + Script to upload nhanes data to a mysql database (la\_forge)

## Packages Used

Flask

ScikitLearn (version 0.19.1)

Numpy

Scipy

## Query format

Referred to as <query> in API calls

* 1. Targets, Variables, and Params (optional)
     1. Separated by “&”
  2. Each section has name of the target/param/variable and the value
     1. separated by a “+”
  3. Each section can have multiple targets, params, variables
     1. separated by a “,”
     2. Possible params:
        1. 'min\_subset\_size' : 50
           1. Min size of subsets to compare using KS test during normalization
        2. 'num\_subsets' : 50,
           1. Maximum number of subsets to use during normalization to find associated groups
        3. 'threshold\_to\_reject' : .10,
           1. Min pvalue to use subset in distribution data
        4. 'Size\_of\_distribution':200
           1. Number of points to return for the density plot output
        5. 'Bandwidth':’auto’,
           1. Used in creating density plot. Higher values create a smoother plot.
        6. 'Upper\_bound':97.725,
           1. Maximum percentile shown on the graph, to prevent too much unnecessary space being taken up by outliers.
        7. 'Lower\_bound':2.225
           1. Minimum percentile shown on graph, to prevent too much unnecessary space being taken up by outliers.
  4. Basic Format:
     1. targets=target\_name+target\_value&variables=var\_name+var\_val&params=param\_name
  5. Examples:
     1. http://kb.lab100.org/normative\_data/targets=triglycerides+48&variables=gender+1,age+40,ethnicity+3
     2. http://kb.lab100.org/normative\_data/targets=ldl+91&variables=gender+2,age+22,ethnicity+1
     3. http://kb.lab100.org/normative\_data/targets=weight+234.6,oxygen\_saturation+100 &variables=gender+1,height+190&params=threshold\_to\_reject+0.05

## Available API calls

Posts:

1. derived\_stats
   1. Gets calculated values for inputs. For input format, see /sample\_stats\_for\_derived . For output format, see /sample\_derived\_stats

Gets

1. normative\_data/<query>
   1. Gives normative data for your query string input
2. graph/<query>
   1. Displays sample graph of the distribution that you would get for a given query
3. papers/<topic>
   1. Returns a list of papers associated with topics
4. topics
   1. Returns a list of topics
5. sample\_derived\_stats
   1. Takes the sample data in sample\_post.py, runs the calculations on it, and outputs them.
6. Sample\_stats\_for\_derived
   1. Returns a json of the sample data in sample\_post.py for an example of the expected input to /derived\_stats
7. NHANES\_vars
   1. List of all nhanes variables
8. cutoffs
   1. List of all cutoffs
9. cutoffs/<variable>
   1. Shows the cutoffs for the specific <variable>
10. vars
    1. Prints all of the lab100 variables
11. cutoffs\_names
    1. List of vars that have specific cutoffs in cutpoints file.

## 

## [Bitbucket Link](https://bitbucket.org/MaxTomlinson/lab100kb/)