Search

Top of Form

Bottom of Form

Menu

1. [Data Analysis with Python](https://www.freecodecamp.org/learn/data-analysis-with-python)
2. [Data Analysis with Python Projects](https://www.freecodecamp.org/learn/data-analysis-with-python/#data-analysis-with-python-projects)

**Medical Data Visualizer**

You will be [working on this project with our Replit starter code](https://replit.com/github/freeCodeCamp/boilerplate-medical-data-visualizer).

We are still developing the interactive instructional part of the Python curriculum. For now, here are some videos on the freeCodeCamp.org YouTube channel that will teach you everything you need to know to complete this project:

* [Python for Everybody Video Course](https://www.freecodecamp.org/news/python-for-everybody/)(14 hours)
* [How to Analyze Data with Python Pandas](https://www.freecodecamp.org/news/how-to-analyze-data-with-python-pandas/) (10 hours)

In this project, you will visualize and make calculations from medical examination data using matplotlib, seaborn, and pandas. The dataset values were collected during medical examinations.

**Data description**

The rows in the dataset represent patients and the columns represent information like body measurements, results from various blood tests, and lifestyle choices. You will use the dataset to explore the relationship between cardiac disease, body measurements, blood markers, and lifestyle choices.

File name: medical\_examination.csv

| **Feature** | **Variable Type** | **Variable** | **Value Type** |
| --- | --- | --- | --- |
| Age | Objective Feature | age | int (days) |
| Height | Objective Feature | height | int (cm) |
| Weight | Objective Feature | Weight ok | float (kg) |
| Gender | Objective Feature | Gender ok | categorical code |
| Systolic blood pressure | Examination Feature | ap\_hi | int |
| Diastolic blood pressure | Examination Feature | ap\_lo | int |
| Cholesterol | Examination Feature | cholesterol | 1: normal, 2: above normal, 3: well above normal |
| Glucose | Examination Feature | gluc | 1: normal, 2: above normal, 3: well above normal |
| Smoking | Subjective Feature | smoke | binary |
| Alcohol intake | Subjective Feature | alco | binary |
| Physical activity | Subjective Feature | active | binary |
| Presence or absence of cardiovascular disease | Target Variable | cardio | binary |

**Tasks**

Create a chart similar to examples/Figure\_1.png, where we show the counts of good and bad outcomes for the cholesterol, gluc, alco, active, and smoke variables for patients with cardio=1 and cardio=0 in different panels.

Use the data to complete the following tasks in medical\_data\_visualizer.py:

* Add an overweight column to the data. To determine if a person is overweight, first calculate their BMI by dividing their weight in kilograms by the square of their height in meters. If that value is > 25 then the person is overweight. Use the value 0 for NOT overweight and the value 1 for overweight.
* Normalize the data by making 0 always good and 1 always bad. If the value of cholesterol or gluc is 1, make the value 0. If the value is more than 1, make the value 1.
* Convert the data into long format and create a chart that shows the value counts of the categorical features using seaborn's catplot(). The dataset should be split by 'Cardio' so there is one chart for each cardio value. The chart should look like examples/Figure\_1.png.
* Clean the data. Filter out the following patient segments that represent incorrect data:
  + diastolic pressure is higher than systolic (Keep the correct data with (df['ap\_lo'] <= df['ap\_hi']))
  + height is less than the 2.5th percentile (Keep the correct data with (df['height'] >= df['height'].quantile(0.025)))
  + height is more than the 97.5th percentile
  + weight is less than the 2.5th percentile
  + weight is more than the 97.5th percentile
* Create a correlation matrix using the dataset. Plot the correlation matrix using seaborn's heatmap(). Mask the upper triangle. The chart should look like examples/Figure\_2.png.

Any time a variable is set to None, make sure to set it to the correct code.

Unit tests are written for you under test\_module.py.

**Development**

For development, you can use main.py to test your functions. Click the "run" button and main.py will run.

**Testing**

We imported the tests from test\_module.py to main.py for your convenience. The tests will run automatically whenever you hit the "run" button.

**Submitting**

Copy your project's URL and submit it to freeCodeCamp.