SIT220/731 2024.T3: Task 3P

Working with **numpy** Matrices (Multidimensional Data)

Last updated: 15th November 2024

Contents

1 Introduction

This task is related to Module 1 and 2; see the *Learning Resources* on the unit site.

This task is due on **Week 11 (Sunday)**. Start tackling it as early as possible. If we find your first solution incomplete or otherwise incorrect, you will still be able to amend it based on the generous feedback we will give you. In case of any problems/questions, do hot hesitate to attend our on-campus/online classes or use the Discussion Board on the unit site.

Submitting after the aforementioned due date will incur a late penalty. This task is part of the **hurdle requirements** in this unit. Not submitting the correct version on time results in failing the unit.

All submissions will be checked for plagiarism. You are expected to work independently on your task solutions. Never share/show parts of solutions with/to anyone.

2 Question 1 - 10

Create a single Jupyter/IPython notebook (see the *Artefacts* section below for all the requirements), where you perform what follows.

Q1. From https://github.com/gagolews/teaching-data/tree/master/marek, download the two following excerpts from the National Health and Nutrition Examination Survey (NHANES dataset) that give body measurements of adult males and females.

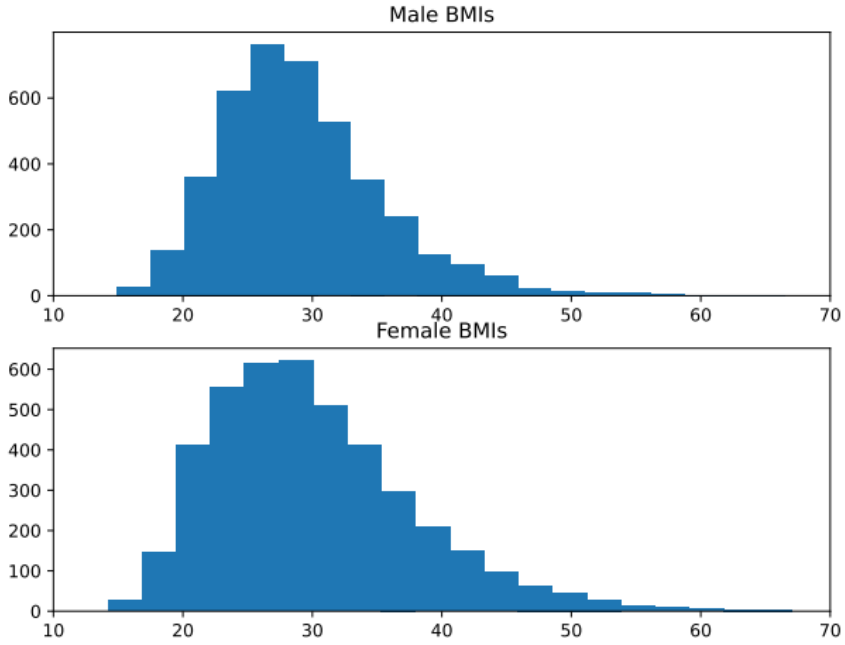
- `nhanes_adult_male_bmx_2020.csv`,
- `nhanes_adult_female_bmx_2020.csv`.

Q2. Read them as **`numpy`** matrices named `male` and `female` using `numpy.genfromtxt`. Each matrix consists of seven columns:

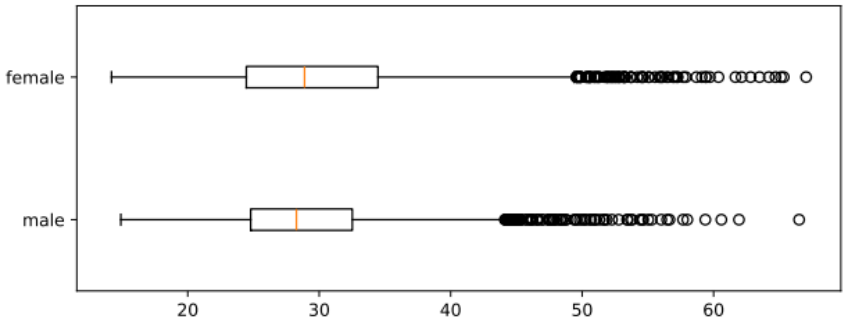
1. weight (kg),
2. standing height (cm),
3. upper arm length (cm),
4. upper leg length (cm),
5. arm circumference (cm),
6. hip circumference (cm),
7. waist circumference (cm).

Q3. In both cases, add the eight column which stores the body mass indices of the participants.

Q4. On a **single** plot, draw two histograms: for male BMIs (top subfigure) and for female BMIs (bottom subfigure) **one below another**. Set the number of histogram bins to 20. Use `matplotlib.pyplot.subplot` to create two subplots in one figure. Call `matplotlib.pyplot.xlim` to make the **x-axis limits identical for both subfigures** (work out the appropriate limits yourself). For example:



Q5. Using a *single* call to `matplotlib.pyplot.boxplot`, draw a box-and-whisker plot giving the male and female BMIs, with **two boxes one below another** (on one plot) so that they can be compared to each other. Note that the `boxplot` function can be fed with a list of two vectors like `[male_BMIs, female_BMIs]`. For example:

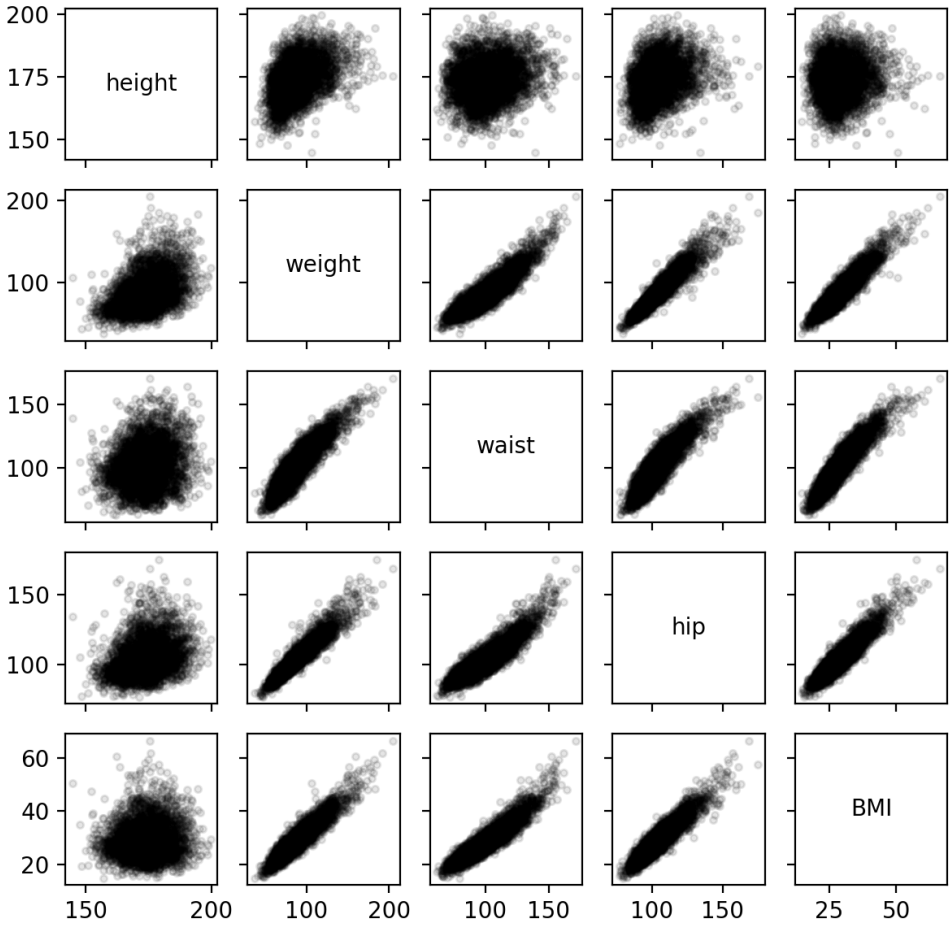


Q6. Compute the basic numerical aggregates of the male and female BMIs (measures of location, dispersion, and shape). Report them in a readable format. Example formatting of the aggregates:

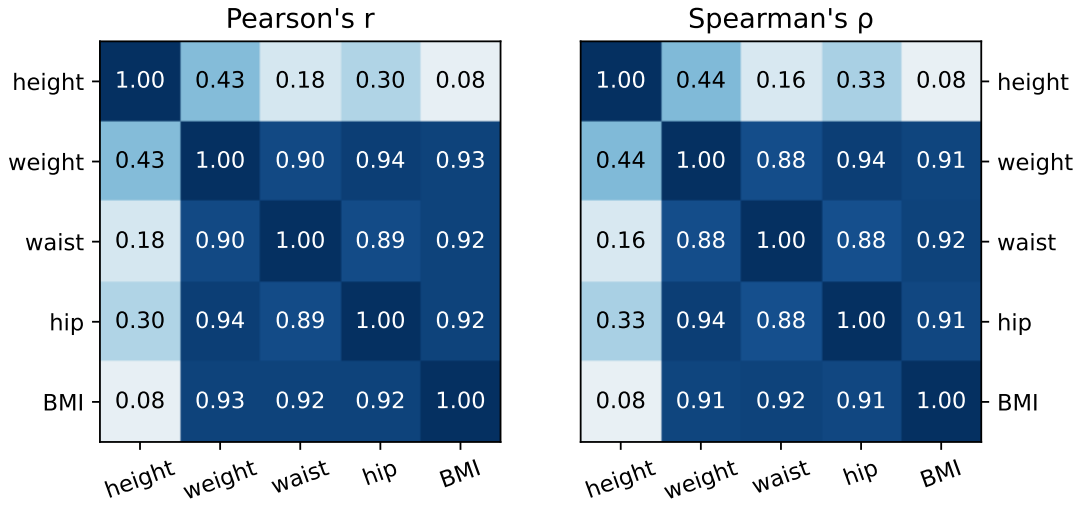
```
##             female   male
## BMI mean     30.10  29.14
##     median   28.89  28.27
##     min      14.20  14.91
##     max      67.04  66.50
##     std       7.76   6.31
##     IQR      10.01   7.73
##     skew      0.92   0.97
```

Q7. In your own words, describe the two distributions based on the results obtained in subtasks Q4, Q5, and Q6 above (e.g., are they left-skewed, how they differ, which one has more dispersion, and so forth).

Q8. Draw a scatterplot matrix (pairplot) for the male heights, weights, waist circumferences, hip circumferences, and BMIs (these five columns only); see the `pairplot` function in section 7.4.3 of our book. Example output (yours can be more aesthetic):



Q9. Compute Pearson's *and* Spearman's correlation coefficients for all pairs of variables mentioned in subtask Q8. Present/visualise these coefficients on two correlation heatmaps (with correlation coefficients printed inside the coloured cells); see the `corrheatmap` function in Section 9.1.2 of our book. Example outputs:



Q10. Discuss the findings from subtasks Q8 and Q9.

Important. Remember that this is an exercise where you demonstrate the mastery of **`numpy`** matrices, and not **`pandas`** data frames. The use of **`pandas`** is forbidden. You can use **`scipy`**, though.

All packages must be imported and data must be loaded at the beginning of the file (only once!).

3 Question 11 - 12 for Postgraduate (SIT731)

Postgraduate students are **additionally required** to solve/address/discuss what follows. Integrate these new requirements into the above subtasks (do not create a separate section of the report).

Q11. Create a new matrix `zmale` being a version of the `male` dataset with each of its eight columns standardised (by computing the z-scores of each column).

Q12. Perform the aforementioned subtasks Q8–Q10 on `zmale` *instead* of on the original `male` dataset (do not include two pairplots nor four heatmaps).

Note that the pairplot will look exactly like above, but with different ranges on the x- and y-axes (ca. from -3 to 3, but maybe a bit less/more sometimes). The correlation coefficients' values will be exactly the same as in the undergraduate version, because standardisation only involves the shifting and scaling of data.

4 Artefacts

The solution to the task must be included in a single Jupyter/IPython notebook (an .ipynb file) running against a Python 3 kernel. The use of G**gle Colab is discouraged. Nothing beats a locally-installed version where you have full control over the environment. Do not become dependent on third-party middlemen/distributors. Choose freedom instead.

Make sure that your notebook has a **readable structure**; in particular, that it is divided into sections. Use rich Markdown formatting (text in dedicated Markdown chunks – not just Python comments).

Do not include the questions/tasks from the task specification. Your notebook should read nicely and smoothly – like a report from data analysis that you designed yourself. Make the flow read natural (e.g., *First, let us load the data on… Then, let us determine… etc.*). Imagine it is a piece of work that you would like to show to your manager or clients — you certainly want to make a good impression. Check your spelling and grammar. Also, use formal language.

At the start of the notebook, you need to provide: the **title** of the report (e.g., *Task 42: How Much I Love This Unit*), your **name**, **student number**, **email address**, and whether you are an **undergraduate (SIT220) or postgraduate (SIT731)** student.

Then, add 1–2 introductory paragraphs (an introduction/abstract – what the task is about).

Before each nontrivial code chunk, briefly **explain** what its purpose is. After each code chunk, **summarise and discuss the obtained results** (in a few sentences).

Conclude the report with 1–2 paragraphs (summary/discussion/possible extensions of the analysis etc.).

Limitations of the OnTrack ipynb-to-pdf renderer:

Ensure that your report as seen in OnTrack is aesthetic (see *Download submision PDF* after uploading the .ipynb file). The OnTrack ipynb-to-pdf renderer is imperfect. We work with what we have. Here are the most common Markdown-related errors.

- Do not include any externally loaded images (via the `![label](href)` Markdown command), for they lead to upload errors.
- Do not input HTML code in Markdown.
- Make sure you leave one blank line before and after each paragraph and bullet list. Do not use backslashes at the end of the line.
- Currently, also *LaTeX formulae* and Markdown tables are not recognised. However, they do not lead to any errors.

Checklist:

1. Header, introduction, conclusion (Markdown chunks).
2. Text divided into sections, all major code chunks commented and discussed in your own words (Markdown chunks).
3. Every subtask addressed/solved. In particular, all reference results that are part of the task specification have been reproduced (plots, computed aggregates, etc.).
4. The report is readable and neat. In particular:
    - all code lines are visible in their entirety (they are not too long),
    - code chunks use consecutive numbering (select *Kernel - Restart and Run All* from the Jupyter menu),
    - rich Markdown formatting is used (`# Section Title`, `* bullet list`, `1. enumerated list`, `| table |`, `*italic*`, etc.),
    - the printing of unnecessary/intermediate objects is minimised (focus on reporting the results specifically requested in the task specification).

Submissions which do not *fully* (100%) conform to the task specification *on* the cut-off date will be marked as FAIL.

Good luck!

5 Intended Learning Outcomes

ULO	Is Related?
ULO1 (Data Processing/Wrangling)	YES
ULO2 (Data Discovery/Extraction)	YES
ULO3 (Requirement Analysis/Data Sources)	YES
ULO4 (Exploratory Data Analysis)	YES
ULO5 (Data Privacy and Ethics)	NO