67978: A Needle in a Data Haystack
Introduction to Data Science
Exercise #1: Clustering, RecSys, Frequent Itemsets
Due: May 18th, 11:59pm, on Moodle

Exercise 1 - Practical Part Instructions

Read the instructions carefully (that's a good idea in general).

- Each person submits their own theoretical part. The theoretical part should be a single file in pdf format only (no docx or jpg) named ex1t_ID.pdf (ID is your ID).
- If you are submitting handwritten answers, make sure they are crystal clear.
- Only one person from each group should submit the practical part code and answers. Both people (or all three if one is in milium) should write the name and id of their partners in the theoretical part pdf.
- For the practical part, the person who submits should submit a single ZIP file named ex1p_ID.zip (where ID is the ID of the person submitting). The zip should contain a folder named code and a folder called output.
- All outputs, printouts, and analyses must be included in a single consolidated report for submission.
- Points may be reduced for submissions that fail to comply.
- Make sure you follow the News forum and HW forum for any updates.

1 Overview

You will acquire, clean and integrate three country-level datasets, preparing them for future analysis.

1) Data acquisition: web-crawling + CSV ingestion with `pandas`.
2) Rigorous cleaning: types, missing values, standardised names.
3) Feature engineering and scaling.

2 Datasets

1) Dataset A – Demographics Data.
Crawl starting from `https://www.worldometers.info/demographics/`.
Required columns:

- `Country`
- `LifeExpectancy_Both` (Both Sexes, in years)
- `LifeExpectancy_Female` (Females, in years)
- `LifeExpectancy_Male` (Males, in years)
- `UrbanPopulation_Percentage` (percentage without commas)
- `UrbanPopulation_Absolute` (if available)
- `PopulationDensity` (per square kilometer)

2) Dataset B – GDP per Capita 2021.
File `gdp_per_capita_2021.csv` with `Country`, `GDP_per_capita_PPP`.

3) Dataset C – Population 2021.
File `population_2021.csv` with `Country`, `Population`.
Population distinguishes mega-states (e.g. India) from small rich states (e.g. Luxembourg) when GDP per capita is similar.

Typical ranges: Life expectancy ≈ 53–88 yr, GDP pc ≈ \$200–105,000, Population ≈ 10^5–1.4×10^9.

3 Part 1 – Data Acquisition

3.1 Web-crawling Demographics Data

Write a Python script to crawl and collect detailed demographics data:

a) Start your crawl at `https://www.worldometers.info/demographics/`

b) Programmatically extract all links to individual country pages from the "Demographics of Countries" section (that is, do not manually copy all links; if the webpage changes, we want your code to still work).

c) For each country page, your code should:

- Visit the page using `requests`
- Parse the HTML (e.g., with `BeautifulSoup`)
- Extract the following data:
  - Life Expectancy (Both Sexes, in years)
  - Life Expectancy (Females) in years
  - Life Expectancy (Males) in years
  - Urban Population percentage (without commas)
  - Urban Population absolute numbers (if available)
  - Population Density per square kilometer
- If any data point is missing, record "None" for that field

d) Load all extracted data into a **Pandas DataFrame** named `df_demographics`. Cast all numeric fields to appropriate numeric types, and save the resulting DataFrame to `output/demographics_data.csv`.

e) Print the first 10 rows of `df_demographics` before sorting by the `Country` column (using `df_demographics.head(10)`
then sort the DataFrame by `Country` in ascending order and print the first 10 rows again.

f) Save both printed tables to `output/demographics_before_sort.csv` and `output/demographics_after_sort.csv`,
and include them in the consolidated report.

3.2 Load the 2021 CSVs

Load and verify the provided CSV files:

a) Read the files into DataFrames `df_gdp` and `df_pop` using `na_values=["None"]` so textual "None" is treated as missing.

b) Confirm that `df_gdp` contains columns `Country` and `GDP_per_capita_PPP`, and that `df_pop` contains columns `Country` and `Population`.

c) Ensure numeric types with `pd.to_numeric` for both the GDP and Population columns.

d) Print the first 5 rows of each DataFrame before sorting by `Country`, then sort in ascending order and print the first 5 rows again. Include both outputs in the final report and save them as CSV files in `output/` (e.g., `gdp_before_sort.csv`, `gdp_after_sort.csv`, `pop_before_sort.csv`, `pop_after_sort.csv`).

e) Run `describe()` on both DataFrames. Save the resulting tables to `output/gdp_describe.csv` and `output/pop_describe.csv`, and include them in the consolidated report under the section titled "Part 1 preview".

3.3 Deliverables

For Part 1, submit the following:

- A well-documented Python script named `code/demographics_crawler.py` for crawling and extracting demographics data.
- The full crawled demographics table saved as `output/demographics_data.csv`.
- A section titled **"Part 1 preview"** in your consolidated report, including:

- The first 10 rows of the demographics dataset before and after sorting by `Country`, referencing `output/demographics_before_sort.csv` and `output/demographics_after_sort.csv`.
- The first 5 rows of the GDP and Population datasets before and after sorting by `Country`, referencing `output/gdp_before_sort.csv`, `output/gdp_after_sort.csv`, `output/pop_before_sort.csv`, and `output/pop_after_sort.csv`.
- Summary statistics tables for GDP and Population from `output/gdp_describe.csv` and `output/pop_describe.csv`.
- A printed confirmation of DataFrame shapes and column names for all three datasets.

• A section titled **"Demographics Data Analysis"** in your consolidated report, including:
  - For each numeric field crawled: mean, standard deviation, minimum, maximum, median, and count of missing values.
  - The Pearson correlation coefficient between `LifeExpectancy_Both` and `PopulationDensity`.

4 Part 2 – Data Cleaning

Definitions. Invalid = non-numeric, negative, or life expectancy outside 40–100 yr.
Missing = `None`.
Outlier (Tukey) = value outside $[Q_1 - 1.5\,\text{IQR}, Q_3 + 1.5\,\text{IQR}]$, where Q_1 is the first quartile (25th percentile), Q_3 is the third quartile (75th percentile), and IQR is the interquartile range ($Q_3 - Q_1$). This method identifies data points that fall far from the central tendency of the dataset. For more information, see: `https://en.wikipedia.org/wiki/Outlier#Tukey's_fences`.

4.1 Clean `df_demographics`

Clean the web-crawled demographics dataset:

a) Ensure all fields are properly converted to numeric format (float) where appropriate.
b) Remove rows with missing or invalid values in the Life Expectancy (Both Sexes) field. Invalid values are defined as:
  - Non-numeric values
  - Negative values
  - Life expectancy values outside the range of 40–100 years

c) For Urban Population and Population Density fields, convert to numeric ensuring any "None" values are properly handled.
d) Normalize the `Country` column consistently: remove the prefix `"the "` if present, convert to Title Case, and strip leading/trailing whitespace. (No other changes to country names are allowed.)
e) Document original and corrected country names in `output/name_mismatches.csv`.
f) Apply the name standardization and set `Country` as the DataFrame index to prepare for later merging operations.

4.2 Clean `df_gdp`

Process the GDP per capita dataset:

a) Verify `GDP_per_capita_PPP` is numeric, converting if necessary by removing commas or special characters.
b) Remove rows where GDP data is missing (None) and document these removals in `output/dropped_gdp.csv`.
c) Identify outliers in the GDP data using the Tukey method (values outside $[Q_1 - 1.5\,\text{IQR}, Q_3 + 1.5\,\text{IQR}]$). Count these outliers, print the count to console, and include this count in your report. Do not remove outliers at this stage.
d) Check for duplicate country entries. If duplicates are found, keep only one entry per country and document your decision process.
e) Apply the same country name mapping used for the demographics dataset.
f) Set `Country` as the DataFrame index for consistency.

4.3 Clean `df_pop`

Apply similar cleaning procedures to the population dataset:

a) Ensure `Population` is numeric, removing any non-numeric characters if necessary.

b) Remove rows where population data is missing (None) and document the number of rows removed.

c) For outlier detection, first apply log10 transformation to the population data (due to its extreme right-skew), then identify outliers using the Tukey method on the transformed data. Count and report these outliers but do not remove them.

d) Check for duplicates and apply the same country name harmonization approach used in previous datasets.

e) Set `Country` as the DataFrame index for consistent merging.

4.4 Normalisation Plan

Normalization will be performed on the final features in Section 5.3. No normalization is required in this section;

4.5 Deliverable

Create a comprehensive cleaning summary:

- Prepare a document named `output/cleaning_summary.pdf` that outlines all data issues found and cleaning methods applied.
- Format this as a bullet list with clear issue $\rightarrow$ action mapping for each dataset.
- For each dataset include:
  - Issues encountered (missing values, outliers, type conversions needed)
  - Actions taken to address each issue
  - Number of rows before and after cleaning
- Also include this same cleaning summary in your consolidated report.

5 Part 3 – Feature Engineering

5.1 New Feature

Create a Total GDP feature by multiplying GDP_per_capita_PPP and Population:

- Calculate: Total GDP = GDP_per_capita_PPP $\times$ Population
- Ensure population is in absolute numbers (not millions) for correct calculations
- Store this as a new column called `TotalGDP` in your merged dataset
- This feature will help distinguish between small rich countries and large developing economies with similar GDP per capita values

5.2 Log Transforms

Apply log transformations to address data skewness:

- Apply log10 transformation to `GDP_per_capita_PPP` and create a new column called `LogGDPperCapita`
- Apply log10 transformation to `Population` and create a new column called `LogPopulation`
- Do not transform `LifeExpectancy_Both` as it is already approximately normally distributed
- Ensure all values are positive before applying log transformation

5.3 Scaling

Apply z-score normalization to the three selected features:

$$x' = \frac{x - \mu}{\sigma}$$

where μ and σ are computed *after* data cleaning and log transformation.

- Normalize the following columns: `LifeExpectancy_Both`, `LogGDPperCapita`, and `LogPopulation`.
- Perform this step immediately following the log transformations in Section 5.2.
- Do not use any alternative scaling methods (e.g., min-max scaling).
- Save the resulting normalized columns as the final feature matrix in `output/X.npy`.

5.4 Data Integration (*inner join*)

Combine the three datasets into a single analysis dataset:

a) Before joining, ensure that the `Country` column is set as the index in all three DataFrames.

b) Perform an *inner join* on `Country` across all three DataFrames using:
```
df_final = df_demographics.join(df_gdp, how="inner").join(df_pop, how="inner")
```
An *inner join* combines rows from multiple tables based on a common key, and retains only those rows where the key (`Country`) appears in *all* tables. Countries without complete data in any one table will be excluded.

c) Record how many countries remain after the merge.

d) Save the list of countries lost during the join to `output/lost_countries.csv`, **sorted by** `Country` **ascending**.

e) Check for any remaining missing values in the merged dataset:

- Document any missing values in the consolidated report.
- For numeric columns, replace missing entries with the column mean.
- For categorical columns, remove rows with missing values.

f) Build the final feature matrix:

- Create a NumPy array of the selected scaled features, **ordered alphabetically by** `Country`.
- Save this array to `output/X.npy`.

5.5 Deliverables

Submit the following for Part 3:

- `output/X.npy` – the scaled feature matrix ($N \times 3$)
- Documentation in your consolidated report including:
  - Description of transformations applied (log transformation and your chosen scaling method)
  - Updated statistics table showing descriptive statistics after scaling
  - Number of countries in the final merged dataset and a list of the first 10 countries (alphabetically)
  - Table showing the overall descriptive statistics (mean, median, std, min, max) for each collected field from your demographics crawling
  - Sample of your crawled data (first 5 rows) as evidence of correct web crawling implementation
  - A verification section showing that you have successfully collected the additional fields (life expectancy for males/females, urban population data, and population density) from the web crawling process

6 Submission Requirements (Summary)

This assignment requires multiple deliverables that must be organized according to the following structure:

6.1 Theoretical Part (Individual Submission)

- Submit a single PDF file named `ex1t_ID.pdf`, where `ID` is your own ID number.
- If handwritten, make sure the submission is clearly legible.

6.2 Practical Part (One Submission per Group)

Only one group member should submit the following:

- ZIP file named `ex1p_ID.zip` (where ID is the submitter's ID)
- The ZIP must contain:
  a) **code/** folder containing:
     - `demographics_crawler.py` - Web crawling script
     - All other Python files necessary for the assignment
     - README.txt (if needed) with execution instructions

  b) **output/** folder containing:
     - `demographics_data.csv`
     - `name_mismatches.csv`
     - `dropped_gdp.csv`
     - `lost_countries.csv`
     - `cleaning_summary.pdf`
     - `X.npy`
     - `demographics_before_sort.csv`, `demographics_after_sort.csv`
     - `gdp_before_sort.csv`, `gdp_after_sort.csv`
     - `pop_before_sort.csv`, `pop_after_sort.csv`
     - `gdp_describe.csv`, `pop_describe.csv`

  c) **consolidated_report.pdf** - A comprehensive single document containing:
     - "Part 1 preview" section with required table outputs (i.e., data tables shown in the PDF, not console logs)
     - First 10 rows of the crawled demographics data
     - First 5 rows of each CSV dataset
     - Summary statistics from `describe()` for all datasets
     - "Demographics Data Analysis" section with:

        * Table of descriptive statistics (mean, median, std, min, max) for each field
        * Correlation coefficient between `LifeExpectancy_Both` and `PopulationDensity`
    - Counts of outliers found in each dataset
    - Cleaning summary (duplicate of `cleaning_summary.pdf`)
    - Updated statistics table after feature engineering
    - Sample of crawled data to verify correct implementation
    - Any additional required outputs as specified in the assignment

6.3 Submission Checklist

Before submitting, verify that:

- The `code/` folder contains exactly the required Python scripts (no extra files).
- The `output/` folder contains all required files, each sorted by `Country` where applicable:
    - `demographics_data.csv`
    - `demographics_before_sort.csv`, `demographics_after_sort.csv`
    - `gdp_before_sort.csv`, `gdp_after_sort.csv`
    - `pop_before_sort.csv`, `pop_after_sort.csv`
    - `gdp_describe.csv`, `pop_describe.csv`
    - `name_mismatches.csv`, `dropped_gdp.csv`
    - `lost_countries.csv`
    - `cleaning_summary.pdf`
    - `X.npy`
- The `consolidated_report.pdf` includes:
    - All printed tables before and after sorting under the ”Part 1 preview” section
    - Summary statistics tables with references to `output/gdp_describe.csv` and `output/pop_describe.csv`
    - The cleaning summary (mirroring `output/cleaning_summary.pdf`)
    - Descriptive statistics after scaling and feature engineering
    - A sample of the crawled data and verification of all collected fields
    - The final number of countries after merging and the first 10 countries in alphabetical order

Note: All generated outputs (raw tables, sorted tables, summary files, arrays) must be saved in the `output/` folder and referenced clearly in the `consolidated_report.pdf`.