Exercise 1 - Practical Part - Consolidated Report

Eliav Dayanof, 208674556
Yair Ben Yakar, 319013090

May 21, 2025

1 Part 1 – Data Aquisition

1.1 Preview

- Preview of the datasets:
  - The first 10 rows of the demographics dataset:

```
from demographics_crawler import crawl_demographics, OUTPUT_DIR
[3] 814ms

df_demographics = crawl_demographics()
df_demographics
[4] 3s 631ms
```

200 rows × 6 cols — Edit in Data Wrangler

Country	LifeExpectancy_Bo…	LifeExpectancy_Fem…	LifeExpectancy_Ma…	UrbanPopulation_Pe…	UrbanPopulation_Ab…	PopulationDensity
	None Count 0	None Count 0	None Count 0	None Count 0	None Count 10	None Count 0
Afghanistan	66.5	68.1	64.9	26.7	11704638.0	67.0
Albania	80.0	81.7	78.1	70.3	1948831.0	101.0
Algeria	76.7	78.1	75.3	74.4	35291973.0	20.0
Angola	65.0	67.5	62.4	68.8	26847887.0	31.0
Antigua and Barbuda	77.9	80.6	74.9	28.5	26823.0	214.0
Argentina	77.7	80.2	75.1	96.0	44010474.0	17.0
Armenia	76.0	79.7	71.8	63.8	1882155.0	104.0
Aruba	76.6	79.1	74.0	44.7	48340.0	601.0
Australia	84.2	86.0	82.4	86.5	23335357.0	4.0
Austria	82.3	84.6	80.0	58.6	5337973.0	111.0

Figure 1: The first 10 rows of demographics data (before and after sorting, stays the same)

  - The first 5 rows of the GDP dataset:

```
df_gdp.head()
[13] 143ms
```

5 rows × 2 cols — Edit in Data Wrangler

	Country	GDP_per_capita_PPP
0	Afghanistan	2144.1665
1	Albania	16353.8090
2	Algeria	14496.8650
3	Andorra	59332.2030
4	Angola	7408.1265

Figure 2: First 5 rows of GDP data (before and after sorting, stays the same)

  - The first 5 rows of the Population dataset:

```
df_pop.head()
[14] 153ms
```

5 rows × 2 cols — Edit in Data Wrangler

	Country	Population
0	Afghanistan	40000360
1	Africa	1413750475
2	Africa (UN)	1413753005
3	Albania	2849591
4	Algeria	44761051

Figure 3: First 5 rows of Population data (before and after sorting, stays the same)

- Summary statistics:
  –

```
df_gdp.describe()
[96]
```

8 rows × 1 cols — Edit in Data Wrangler

	GDP_per_capita_PPP
count	213.000000
mean	25822.604541
std	25794.941595
min	836.665600
25%	6214.017000
50%	16353.809000
75%	38862.090000
max	137947.340000

Figure 4: Summary statistics table for GDP

  –

```
df_pop.describe()
[98]
```

8 rows × 1 cols — Edit in Data Wrangler

	Population
count	2.600000e+02
mean	1.687752e+08
std	7.254974e+08
min	5.150000e+02
25%	5.224038e+05
50%	6.827910e+06
75%	3.425334e+07
max	7.954448e+09

Figure 5: Summary statistics table for Population

- Printed confirmation of DataFrame shapes and column names:
  –

```
print(df_demographics.shape, list(df_demographics.columns), sep="\n")
[19] 125ms
(200, 6)
['LifeExpectancy_Both', 'LifeExpectancy_Female', 'LifeExpectancy_Male',
'UrbanPopulation_Percentage', 'UrbanPopulation_Absolute', 'PopulationDensity']
```

Figure 6: Demographics table shape and columns

  –

```
gdp_dataset_expected_cols = ["Country", "GDP_per_capita_PPP"]
confirm_cols(df_gdp, gdp_dataset_expected_cols)
print(df_gdp.shape, df_gdp.columns)
✓ [16] 130ms

['Country', 'GDP_per_capita_PPP'] cols exist in dataframe
(213, 2) Index(['Country', 'GDP_per_capita_PPP'], dtype='object')
```

Figure 7: GDP table shape and columns

–

```
pop_dataset_expected_cols = ["Country", "Population"]
confirm_cols(df_pop, pop_dataset_expected_cols)
print(df_pop.shape, df_pop.columns)
✓ [15] 144ms

['Country', 'Population'] cols exist in dataframe
(260, 2) Index(['Country', 'Population'], dtype='object')
```

Figure 8: Population table shape and columns

1.2 Demographics Data Analysis

- Neumeric-fields Summary:

```
num_cols = df_demographics.select_dtypes("number").columns
summary = (
    df_demographics[num_cols]
    .agg(["mean", "std", "min", "max", "median"])
    .T
    .assign(missing=df_demographics[num_cols].isna().sum())
)
display(summary)  # nice HTML in notebook
[100]
```

6 rows · 6 rows × 6 cols — Static Output

	mean	std	min	max	median	missing
LifeExpectancy_Both	7.397750e+01	7.004589e+00	54.8	85.8	74.85	0
LifeExpectancy_Female	7.659650e+01	7.126054e+00	55.1	88.4	78.05	0
LifeExpectancy_Male	7.137950e+01	6.979282e+00	53.5	83.1	71.20	0
UrbanPopulation_Percentage	5.727600e+01	2.514603e+01	0.0	98.8	60.55	0
UrbanPopulation_Absolute	2.483416e+07	8.517914e+07	26661.0	956553854.0	5011189.50	10
PopulationDensity	3.768200e+02	1.866303e+03	2.0	24067.0	93.00	0

Figure 9: Demographics Data Neumeric-fields Summary

- Pearson correlation:

```
corr = df_demographics["LifeExpectancy_Both"].corr(
    df_demographics["PopulationDensity"]
)
print(f"Pearson correlation (LifeExpectancy Both vs PopulationDensity): {corr:.4f}")
[102]

Pearson correlation (LifeExpectancy Both vs PopulationDensity): 0.1796
```

Figure 10: Pearson correlation of LifeExpectancy Both vs PopulationDensity

2 Part 2 – Cleaning Summary

2.1 Demographics Dataset

Issues encountered → Actions taken

- **Non-numeric values in columns** → All columns except "Country" were explicitly converted to numeric using pd.to_numeric(errors='coerce').

- **Invalid Life Expectancy values (< 40 or > 100)** → Rows outside the valid range [40, 100] for LifeExpectancy_Both were removed.
- **Inconsistent country names** (e.g., "the Gambia") → Standardized with: strip(), title(), and removed "the " prefix.
- **Potential name mismatches after standardization** → Documented in output/name_mismatches.csv.

Row counts

- Rows before cleaning: 200
- Rows after cleaning: 200

2.1.1 GDP per Capita Dataset

Issues encountered → Actions taken

- **Missing or malformed GDP values** → GDP_per_capita_PPP was coerced to numeric, invalid/missing rows were dropped and logged to output/dropped_gdp.csv (in reality there were none).
- **Outliers in GDP values** → Identified (but not removed) using Tukey's method, outlier count of 6 printed to console:

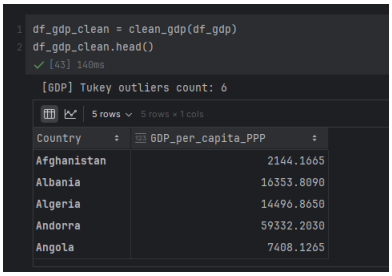

Figure 11: GDP After Cleaning – Outlier Count

- **Duplicate country entries** → Duplicates dropped, keeping the first occurrence per "Country" (in reality there were no duplicates, if there were, perhaps we would've chosen a smarter strategy).
- **Country name mismatches** → Mapped manually to match the Demographics dataset (e.g., "Cape Verde" → "Cabo Verde").
  - Note: We expirmented with trying to map mismatches using fuzzy search techniques to find words with similar spellings and match the Demographics dataset (assuming it as ground truth) spelling (post standardization), but it didn't work out well so we resorted to doing it manually (with sorting and some help of code of course).
    Also, another appraoch we considered was using edit distance.

Row counts

- Rows before cleaning: 213
- Rows after cleaning: 213

Population Dataset **Issues encountered** → Actions taken

- **Non-numeric or missing values in Population** → Coerced to numeric, missing values dropped and recorded in output/dropped_population.csv.
- **Extreme outliers due to scale differences** → Detected using log10 + Tukey's method, outlier count of 1 printed to console, but values retained:

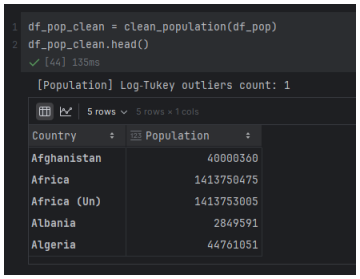

Figure 12: Population After Cleaning – Outlier Count

- **Duplicate entries by country** → Duplicates dropped (first occurrence kept).
- **Country name mismatches** → Mapped manually to match ground truth Demographics dataset (e.g., "Saint Vincent And The Grenadines" → "St. Vincent & Grenadines").

Row counts

- Rows before cleaning: 260
- Rows after cleaning: 260

3 Part 3 – Feature Engineering

Description of Transformations

- Log transformations were applied to GDP_per_capita_PPP (LogGDPperCapita) and Population (LogPopulation) to compress the range and reduce skew.
- We did not log-transform LifeExpectancy_Both, as it is already approximately normally distributed.
- Afterward, we applied z-score normalization to the following columns:
  - LifeExpectancy_Both → LifeExpectancy_z
  - LogGDPperCapita → LogGDPpc_z
  - LogPopulation → LogPop_z This produced three normalized features with zero mean and unit variance, suitable for further analysis.

```python
num_cols = df_final_engineered.select_dtypes("number").columns
summary = (
    df[num_cols]
    .agg(["mean", "std", "min", "max", "median"])
    .T
    .assign(missing=df[num_cols].isna().sum())
)
display(summary)  # nice HTML in notebook
```

✓ [125] 177ms

14 rows · 14 rows × 6 cols — Static Output

	mean	std	min	max	median	missing
LifeExpectancy_Both	7.378859e+01	7.002188e+00	5.480000e+01	8.580000e+01	7.475000e+01	0
LifeExpectancy_Female	7.639620e+01	7.121332e+00	5.510000e+01	8.840000e+01	7.780000e+01	0
LifeExpectancy_Male	7.120109e+01	6.986951e+00	5.350000e+01	8.310000e+01	7.105000e+01	0
UrbanPopulation_Percentage	5.770217e+01	2.386151e+01	0.000000e+00	9.880000e+01	6.025000e+01	0
UrbanPopulation_Absolute	2.528164e+07	8.652082e+07	0.000000e+00	9.565539e+08	4.956721e+06	0
PopulationDensity	3.905109e+02	1.944122e+03	2.000000e+00	2.406700e+04	9.300000e+01	0
GDP_per_capita_PPP	2.480939e+04	2.565802e+04	8.366656e+02	1.379473e+05	1.568298e+04	0
Population	4.243166e+07	1.530701e+08	8.718400e+04	1.426437e+09	8.954937e+06	0
TotalGDP	8.410245e+11	3.007097e+12	3.913878e+08	2.910892e+13	1.022966e+11	0
LogGDPperCapita	4.147027e+00	5.059535e-01	2.922552e+00	5.139713e+00	4.195429e+00	0
LogPopulation	6.874248e+00	8.599970e-01	4.940437e+00	9.154253e+00	6.952062e+00	0
LifeExpectancy_z	-7.723291e-17	1.002729e+00	-2.719207e+00	1.720060e+00	1.376764e-01	0
LogGDPpc_z	-6.999232e-16	1.002729e+00	-2.426737e+00	1.967364e+00	9.592474e-02	0
LogPop_z	3.547887e-16	1.002729e+00	-2.254761e+00	2.658412e+00	9.072911e-02	0

Figure 13: Updated Descriptive Statistics Table After Scaling

Updated Descriptive Statistics Table After Scaling

Evidence of Successful Crawling and Merging

- Final row count: 183 (as seen in the top left corner of the table below.

```python
manual_country_map = {
    "Cape Verde": "Cabo Verde", "Czechia": "Czech Republic (Czechia)", "Cote D'Ivoire": "Côte D'Ivoire", "Democratic Republic Of Congo": "Dr Congo", "Reunion": "Réunion", "Curacao": "Curaçao",
    "Micronesia (Country)": "Micronesia", "Palestine": "State Of Palestine", "Saint Vincent And The Grenadines": "St. Vincent & Grenadines", "Sao Tome And Principe": "Sao Tome & Principe",
    "United States Virgin Islands": "U.S. Virgin Islands", "East Timor": "Timor-Leste",
}
df_gdp_normalized = df_gdp_clean.rename(index=manual_country_map)
df_pop_normalized = df_pop_clean.rename(index=manual_country_map)
df_final = merge_dfs(
    df_demographics_clean, df_gdp_normalized, df_pop_normalized
)
df_final
```

✓ [130] 133ms

1-10 · 184 rows × 8 cols — Edit in Data Wrangler

Country	LifeExpectancy_Bo…	LifeExpectancy_Female	LifeExpectancy_Ma…	UrbanPopul…	UrbanPop…	PopulationDensity	GDP_per_capita_PPP	Population
	None Count 0	None Count 0	None Count 0	None Count 0	None Count 0	None Count 0	None Count 0	None Count 0
Afghanistan	66.5	68.1	64.9	26.7	11704638	67.0	2144.1665	40000360
Albania	80.0	81.7	78.1	70.3	1948831	101.0	16353.8090	2849591
Algeria	76.7	78.1	75.3	74.4	35291973	20.0	14496.8650	44761051
Angola	65.0	67.5	62.4	68.8	26847887	31.0	7408.1265	34532382
Antigua And Barbuda	77.9	80.6	74.9	28.5	26823	214.0	25733.2950	92316
Argentina	77.7	80.2	75.1	96.0	44010474	17.0	26300.2730	45312226
Armenia	76.0	79.7	71.8	63.8	1882155	104.0	15921.7050	2870293
Aruba	76.6	79.1	74.0	44.7	48340	601.0	35696.3100	107524
Australia	84.2	86.0	82.4	86.5	23335357	4.0	57321.5620	25956358
Austria	82.3	84.6	80.0	58.6	5337973	111.0	63576.2420	8967006

Figure 14: Final Merged Table with All Crawling Data