Analyzing Long-Term Demographic Shifts and Dependency Trends in Sweden (1860–2022)

2024-11-07

Project 1: Sweden's Shifting Demographics — A 160-Year Perspective on Dependency

(i) This project dives into over 160 years of demographic transformation in Sweden by analyzing the country's dependency ratio from 1860 to 2022. Using detailed population statistics sourced from SCB [1], including age and gender breakdowns, the data was processed and structured to calculate yearly dependency ratios and uncover long-term trends.

The dependency ratio, as defined by the World Health Organization [2], measures the pressure placed on the working-age population (15–64 years) by dependents—specifically, children (0–14) and the elderly (65+). Expressed as the number of dependents per 100 working-age individuals, this metric offers a powerful lens into societal and economic challenges related to aging populations and changing birth rates. Figure 1 illustrates how this ratio has evolved over time, highlighting key demographic shifts across centuries.

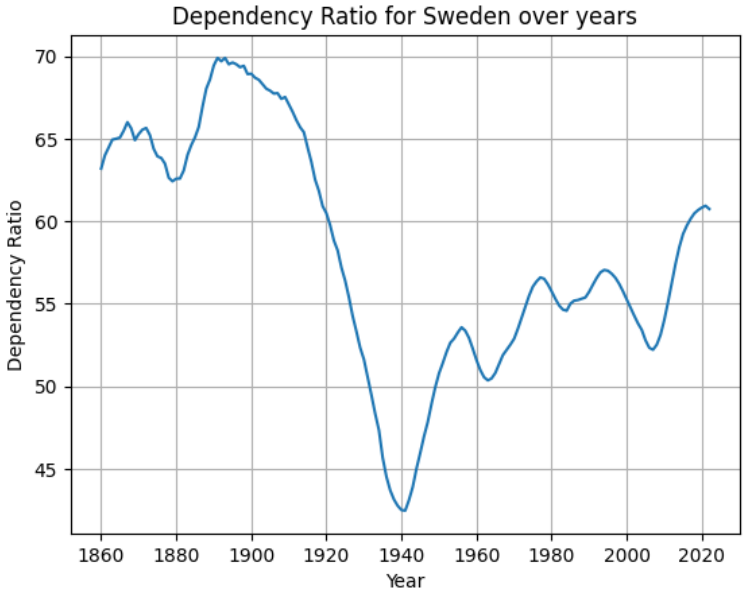

Figure 1: Dependency ratio in Sweden (1860–2022): Number of dependents per 100 working-age individuals

(ii) To paint a more detailed picture, the second part of the project examines how different dependent age groups—children and elderly—have changed as a fraction of the total population. This visualization reveals the distinct contributions of each subgroup to the overall dependency burden. Figure 2 presents the relative proportions of children, elderly, and total dependents from 1860 to 2022, providing insight into how Sweden's age structure has gradually shifted.

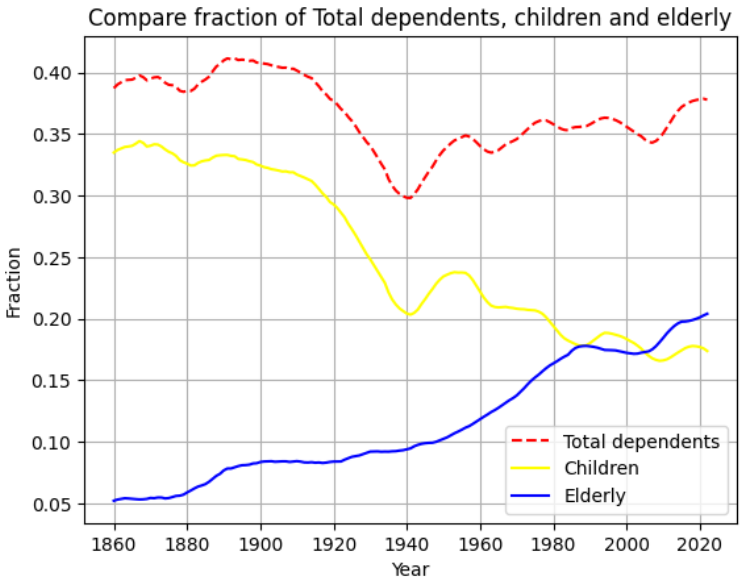

Figure 2: Proportion of children, elderly, and total dependents relative to Sweden's population (1860–2022)

(iii) These figures tell a compelling story of demographic change. One of the most noticeable patterns is the sharp dip in the dependency ratio between the 1910s and 1940s—a period marked by global conflict and economic uncertainty. During these decades, birth rates declined significantly, leading to a temporary drop in the proportion of children and a reduced overall dependency burden.

In contrast, the post-war era saw the beginning of a new trend: a steady rise in the elderly population. This shift reflects increased life expectancy and improvements in healthcare and living standards, hallmarks of an aging, industrialized society. Meanwhile, the proportion of children has declined steadily, driven by changing cultural norms, family planning choices, and dual-income households with less time and resources for larger families.

The demographic consequences of these trends are profound. A society with fewer children and more elderly individuals must contend with a growing dependency burden on a shrinking working-age population. This dynamic is already sparking policy debates in countries like Japan and South Korea—and Sweden is not exempt. Understanding these long-term changes is vital for anticipating future economic and social challenges, from pension systems to workforce sustainability.

References

[1] Statistiska centralbyrån. *Folkmängden efter ålder och kön. År 1860 - 2022.* Retrieved 2023-10-20. 2023. URL: `https://www.statistikdatabasen.scb.se/pxweb/sv/ssd/START__BE__BE0101__BE0101A/BefolkningR1860N/`.

[2] World Health Organization. *The Global Health Observatory Indicator Metadata Registry List: Dependency Ratio*. Retrieved 2023-10-25. 2023. URL: https://www.who.int/data/gho/indicator-metadata-registry/imr-details/1119.

Appendix

1. Importing the necessary libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

2. Connect to the Google Drive to input the raw file

```
from google.colab import drive
drive.mount('/content/drive')
```

```
input_data = pd.read_csv('/content/drive/MyDrive/
    swedish_population_by_year_and_sex_1860-2022.csv')
input_data
```

	age	sex	1860	1861	1862	1863	1864	1865	1866	1867	...	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022
0	0	men	60589	59797	62371	61515	61931	60998	63036	58645	...	58649	59584	59994	61005	59899	60032	59476	58485	58692	54095
1	0	women	58837	58136	60041	59384	60100	59622	60874	56586	...	55359	56296	55884	58018	56715	56807	55907	55104	55971	51091
2	1	men	56001	54544	52933	55776	57346	57776	57017	59071	...	59039	59489	60640	61352	62531	60973	60993	60058	59195	59411
3	1	women	54833	53762	52282	54500	55823	56641	56263	57539	...	55884	56083	57292	57216	59444	57789	57783	56533	55855	56712
4	2	men	52502	54062	51613	50710	53743	55227	55696	55254	...	58721	59807	60292	61817	62258	63361	61598	61348	60596	59723
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
217	108	women	0	0	0	0	0	0	0	0	...	6	8	5	3	3	10	7	9	4	8
218	109	men	0	0	0	0	0	0	0	0	...	0	0	0	0	3	0	0	0	0	0
219	109	women	0	0	0	0	0	0	0	0	...	4	2	6	3	3	1	6	2	6	1
220	110+	men	0	0	0	0	0	0	0	0	...	1	1	1	1	0	3	0	0	0	0
221	110+	women	0	0	0	0	0	0	0	0	...	2	4	2	5	2	2	1	2	1	3

222 rows × 165 columns

Figure 3: Image of raw input data

3. Transforming the dataset from wide format to long format by unpivoting the data

```
melted_data = pd.melt(input_data, id_vars=['age','sex'
    ], value_vars=None, var_name='year', value_name='
    population', col_level=None, ignore_index=True)
print(melted_data.dtypes)
melted_data
```

```
age           object
sex           object
year          object
population     int64
dtype: object
```

	age	sex	year	population
0	0	men	1860	60589
1	0	women	1860	58837
2	1	men	1860	56001
3	1	women	1860	54833
4	2	men	1860	52502
...	...	...	...	...
36181	108	women	2022	8
36182	109	men	2022	0
36183	109	women	2022	1
36184	110+	men	2022	0
36185	110+	women	2022	3

36186 rows × 4 columns

Figure 4: Image of wide format to long format data

4. **Data cleaning: Removing the '+' sign from age values such as '110+' to facilitate conversion to integer format.**

```
melted_data['age'] = melted_data['age'].str.replace('+', '', regex=False)
```

Converting the age, year and population columns from strings to integers.

```
melted_data[['age', 'year', 'population']] = melted_data[['age', 'year', 'population']].astype(int)
print(melted_data.dtypes)
```

5. **We are using `melted_data` dataframe to create a new dataframe called children. It selects only those rows where the age column is less than or equal to 14. This effectively isolates the population of children within the specified age range.**

```
children = melted_data[melted_data['age'] <= 14]
```

The total child population for each year by summing the values in the population column of the grouped data. The result is a series named `children_population_per_year`, where the index represents years and the values represent the total number of children in that age group for each corresponding year.

```
children_per_year = children.groupby('year')
children_population_per_year = children_per_year['population'].sum()
```

Outputs the `children_population_per_year` Series, which contains the summed population of children aged 0 to 14 for each year in the dataset.

```
children_population_per_year
```

year	population
1860	1292962
1861	1318714
1862	1338720
1863	1362024
1864	1380235
...	...
2018	1819729
2019	1834821
2020	1837798
2021	1839103
2022	1829093

163 rows × 1 columns

dtype: int64

Figure 5: Overview of melted data for the children

6. **In the same way the elderly population above or equal to 65 is created.**

```
elderly = melted_data[melted_data['age'] >= 65]
elderly_per_year = elderly.groupby('year')
elderly_population_per_year = elderly_per_year['
    population'].sum()
elderly_population_per_year
```

7. **In the same way the labor force between the age of 15-64 is created.**

```
labor_force = melted_data[(melted_data['age'] < 65) &
    (melted_data['age'] > 14)]
labor_force_per_year = labor_force.groupby('year')
labor_force_population_per_year = labor_force_per_year
    ['population'].sum()
labor_force_population_per_year
```

8. **The dependency ratio is calculated using the population per year for the three different demographics.**

```
dependency_ratio = 100*((children_population_per_year
    + elderly_population_per_year)/
    labor_force_population_per_year)
print(dependency_ratio)
```

9. **Plotting the dependency ratio.**

```
plt.plot(dependency_ratio)
plt.xlabel('Year')
plt.ylabel('Dependency Ratio')
plt.title('Dependency Ratio for Sweden over years')
plt.grid(True)
plt.show()
```

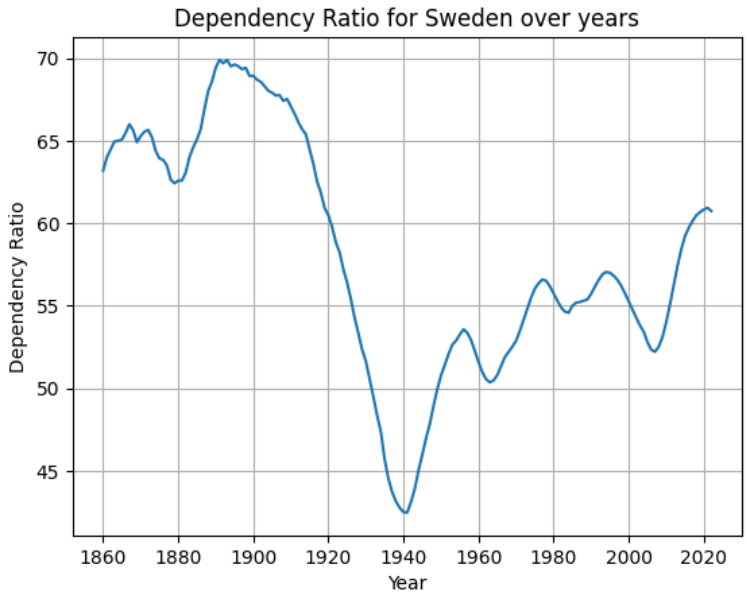

Figure 6: Dependency ratio for Sweden 1860 - 2022

10. **Calculating Population Fractions for Children, Elderly, and Dependents**

```
total_population = melted_data.groupby('year')['
    population'].sum()

fraction_children = children_population_per_year /
    total_population

fraction_elderly = elderly_population_per_year /
    total_population

fraction_dependents = (children_population_per_year +
    elderly_population_per_year) / total_population

fraction_dependents
fraction_children
fraction_elderly
```

11. **Plotting the fraction of children, elderly and total dependents in relation to total population.**

```
plt.plot(fraction_dependents, color = 'red', linestyle ='dashed', label='Total dependents')
plt.plot(fraction_children, color = 'yellow', label='Children')
plt.plot(fraction_elderly, color = 'blue', label='Elderly')
plt.xlabel('Year')
plt.ylabel('Fraction')
plt.title('Compare fraction of Total dependents, children and elderly')
plt.grid(True)
plt.legend()
plt.show()
```

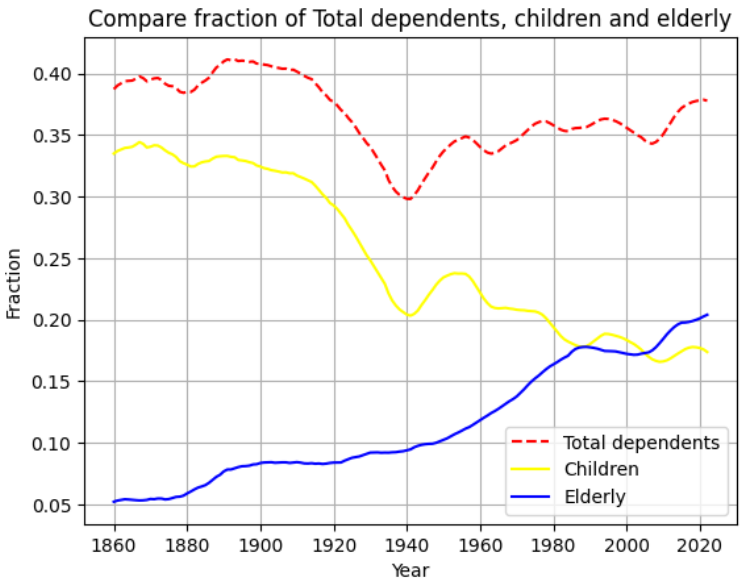

Figure 7: Comparing the fraction of total dependents, children and elderly to the total population