DigHum100Project

June 14, 2024

[1]:
```
pip install --upgrade seaborn
```

```
Requirement already satisfied: seaborn in /srv/conda/lib/python3.11/site-
packages (0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in
/srv/conda/lib/python3.11/site-packages (from seaborn) (1.26.4)
Requirement already satisfied: pandas>=1.2 in /srv/conda/lib/python3.11/site-
packages (from seaborn) (2.2.2)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in
/srv/conda/lib/python3.11/site-packages (from seaborn) (3.7.3)
Requirement already satisfied: contourpy>=1.0.1 in
/srv/conda/lib/python3.11/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(1.2.1)
Requirement already satisfied: cycler>=0.10 in /srv/conda/lib/python3.11/site-
packages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/srv/conda/lib/python3.11/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(4.53.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
/srv/conda/lib/python3.11/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(1.4.5)
Requirement already satisfied: packaging>=20.0 in
/srv/conda/lib/python3.11/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(24.0)
Requirement already satisfied: pillow>=6.2.0 in /srv/conda/lib/python3.11/site-
packages (from matplotlib!=3.6.1,>=3.4->seaborn) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/srv/conda/lib/python3.11/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in
/srv/conda/lib/python3.11/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(2.9.0)
Requirement already satisfied: pytz>=2020.1 in /srv/conda/lib/python3.11/site-
packages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in /srv/conda/lib/python3.11/site-
packages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: six>=1.5 in /srv/conda/lib/python3.11/site-
packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.16.0)
```

Note: you may need to restart the kernel to use updated packages.

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


def isolate_vals(dataset):
    new =  dataset[['Location', 'Value', 'Period']]
    value_str = dataset['Indicator'].values[0]
    new = new.rename(columns = {'Location' : 'income_group', 'Value' :␣
 ↪value_str, 'Period' : 'year'})
    return new
def extract_percentage(dataset, column_name):
    percentage_str = dataset[column_name].str[:4]
    dataset[column_name] = percentage_str.str.strip('[')
    return dataset
```

0.1 Staff availability

[3]:
```
doctors = pd.read_csv('dighum doctors dataset.csv')
doctors
cleaned_doc = isolate_vals(doctors)
cleaned_doc = cleaned_doc.rename(columns = {'Medical doctors (per 10,000)' :␣
 ↪'count'})
cleaned_doc
```

[3]:

	income_group	count	year
0	Upper-middle-income	23.2	2022
1	Low-income	3.1	2022
2	High-income	36.0	2022
3	Lower-middle-income	7.7	2022

[4]:
```
nurses = pd.read_csv('dighum nurse dataset.csv')
nurses_cleaned = isolate_vals(nurses)
nurses_cleaned = nurses_cleaned.rename(columns = {'Nursing and midwifery␣
 ↪personnel (per 10,000)' : 'count'})

nurses_cleaned
```

[4]:

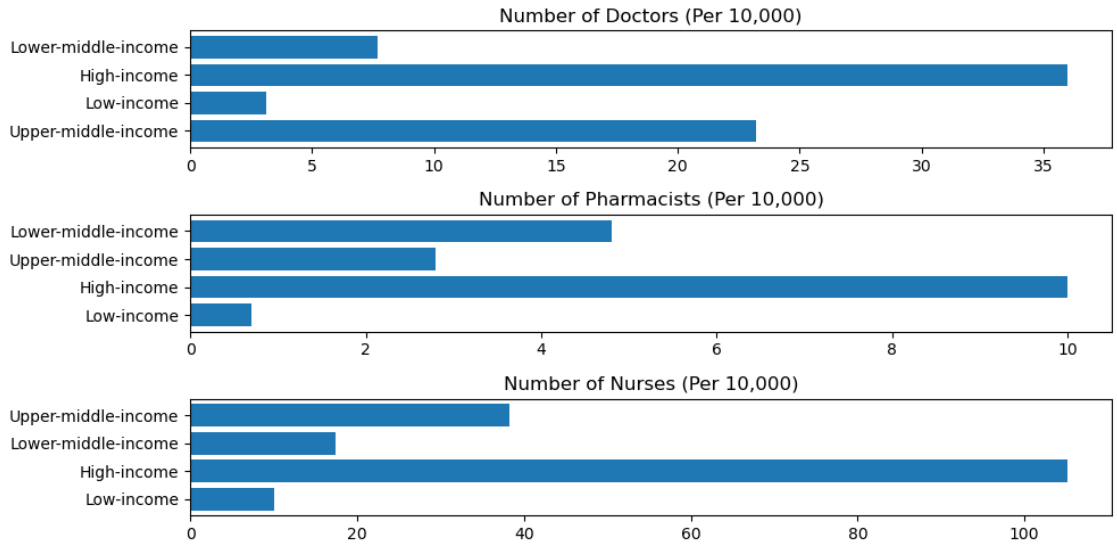
	income_group	count	year
0	Low-income	10.0	2022
1	High-income	105.2	2022
2	Lower-middle-income	17.4	2022
3	Upper-middle-income	38.3	2022

[5]:
```
pharma = pd.read_csv('dighum pharma dataset.csv')
pharma_cleaned = isolate_vals(pharma)
pharma_cleaned = pharma_cleaned.rename(columns = {pharma['Indicator'].values[0]␣
 ↪: 'count'})
pharma_cleaned
```

[5]:

	income_group	count	year
0	Low-income	0.7	2022
1	High-income	10.0	2022
2	Upper-middle-income	2.8	2022
3	Lower-middle-income	4.8	2022

[6]:
```
df_list = [cleaned_doc, pharma_cleaned, nurses_cleaned]
fig, ax = plt.subplots(3, 1, figsize = (10, 5))
for i in range(len(df_list)):
    ax[i].barh(y = 'income_group', width = 'count', data = df_list[i])
ax[0].set_title('Number of Doctors (Per 10,000)')
ax[1].set_title('Number of Pharmacists (Per 10,000)')
ax[2].set_title('Number of Nurses (Per 10,000)')
plt.tight_layout();
```



0.2 Nutrition related conditions

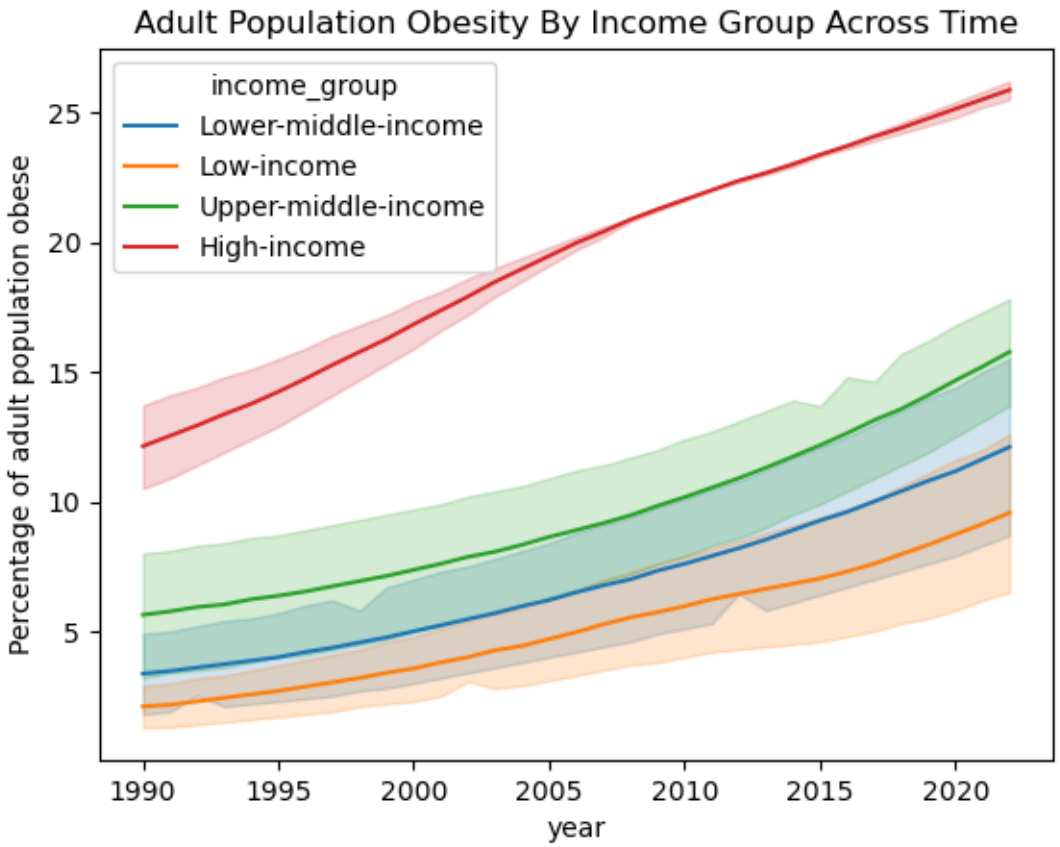
[7]:
```
obesity = pd.read_csv('dighum obesity dataset.csv')
obesity
cleaned_obesity = isolate_vals(obesity)
cleaned_obesity = extract_percentage(cleaned_obesity, 'Prevalence of obesity␣
 ↪among adults, BMI &GreaterEqual; 30 (age-standardized estimate) (%)')
```

```
cleaned_obesity['Prevalence of obesity among adults, BMI &GreaterEqual; 30␣
 ↪(age-standardized estimate) (%)'] = cleaned_obesity['Prevalence of obesity␣
 ↪among adults, BMI &GreaterEqual; 30 (age-standardized estimate) (%)'].
 ↪astype(float)
grouped = cleaned_obesity.drop(columns = ['year']).groupby(['income_group']).
 ↪mean()
grouped
```

[7]:

income_group	Prevalence of obesity among adults, BMI ≥ 30 (age-standardized estimate) (%)
High-income	19.423232
Low-income	5.175758
Lower-middle-income	6.884848
Upper-middle-income	9.527273

[8]:
```
lp = sns.lineplot(data = cleaned_obesity, x = 'year', y = 'Prevalence of␣
 ↪obesity among adults, BMI &GreaterEqual; 30 (age-standardized estimate) (%)',
            hue = 'income_group')
lp.set_ylabel('Percentage of adult population obese')
lp.set_title('Adult Population Obesity By Income Group Across Time');
```

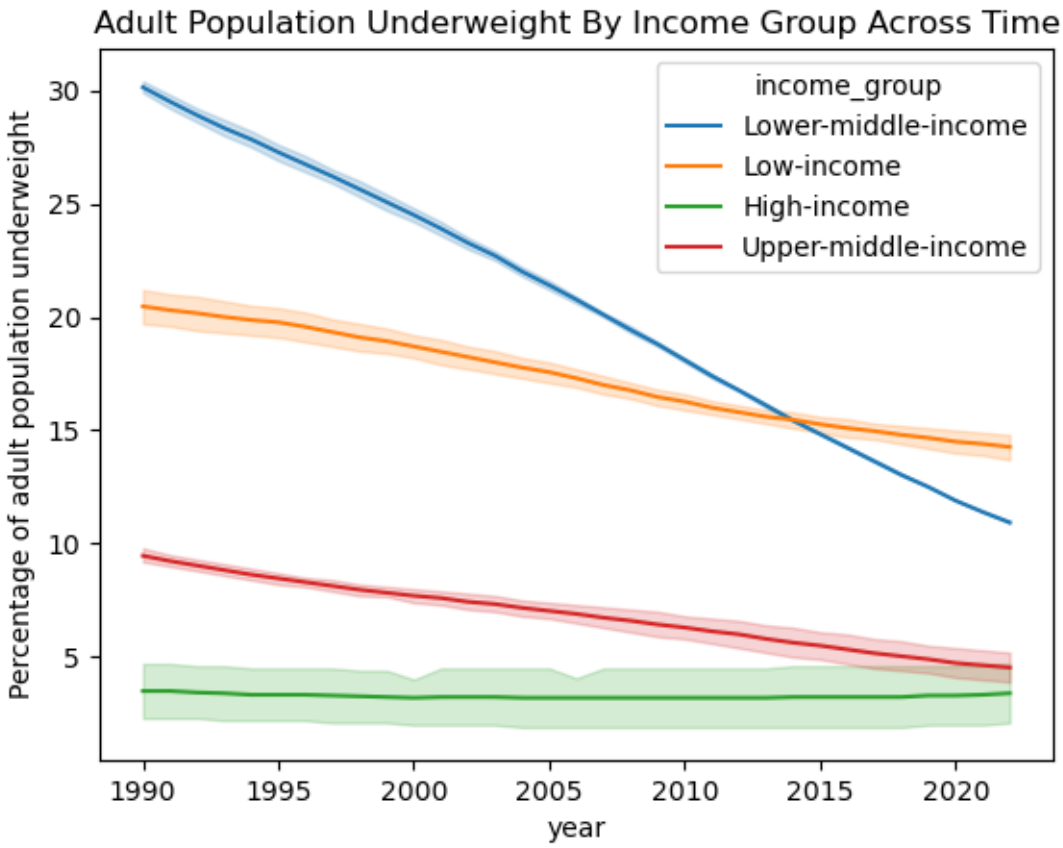


[9]:
```
under = pd.read_csv('dighum underweight dataset.csv')
under_cleaned = isolate_vals(under)
under_cleaned = extract_percentage(under_cleaned, 'Prevalence of underweight␣
 ↪among adults, BMI < 18 (age-standardized estimate) (%)')
under_cleaned['Prevalence of underweight among adults, BMI < 18␣
 ↪(age-standardized estimate) (%)'] = under_cleaned['Prevalence of underweight␣
 ↪among adults, BMI < 18 (age-standardized estimate) (%)'].astype(float)
grouped = under_cleaned.drop(columns = ['year']).groupby(['income_group']).
 ↪mean()
grouped
```

[9]:

income_group	Prevalence of underweight among adults, BMI < 18 (age-standardized estimate) (%)
High-income	3.273737
Low-income	17.298990
Lower-middle-income	20.567677
Upper-middle-income	6.864646

[11]:
```
lp = sns.lineplot(data = under_cleaned, x = 'year', y = 'Prevalence of␣
 ↪underweight among adults, BMI < 18 (age-standardized estimate) (%)',
             hue = 'income_group')
lp.set_ylabel('Percentage of adult population underweight')
lp.set_title('Adult Population Underweight By Income Group Across Time');
```



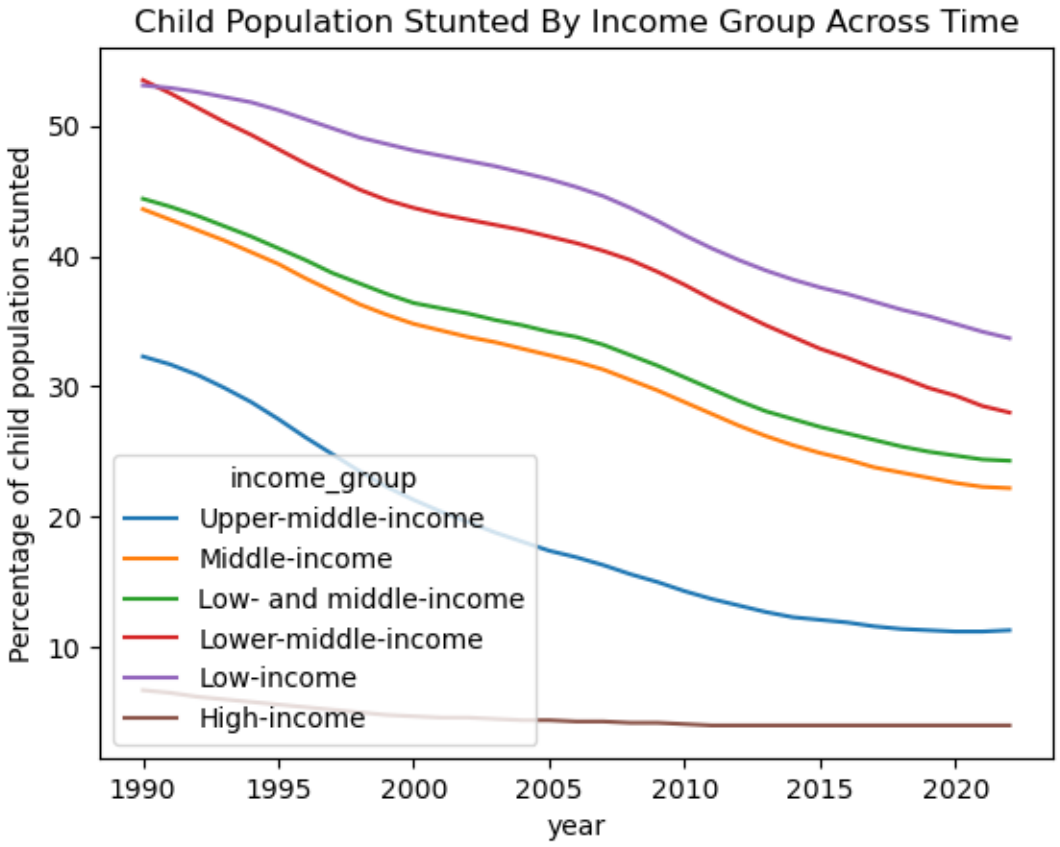
[17]:
```
stunted = pd.read_csv('dighum stunted dataset.csv')
stunted_cleaned = isolate_vals(stunted)
stunted_cleaned = extract_percentage(stunted_cleaned, 'Stunting prevalence␣
 ↪among children under 5 years of age (% height-for-age <-2 SD), model-based␣
 ↪estimates')
stunted_cleaned['Stunting prevalence among children under 5 years of age (%␣
 ↪height-for-age <-2 SD), model-based estimates'] = stunted_cleaned['Stunting␣
 ↪prevalence among children under 5 years of age (% height-for-age <-2 SD),␣
 ↪model-based estimates'].astype(float)
grouped = stunted_cleaned.drop(columns = ['year']).groupby(['income_group']).
 ↪mean()
grouped
```

[17]:

income_group	Stunting prevalence among children under 5 years of age (% height-for-age <-2 SD), model-based estimates
High-income	4.651515
Low- and middle-income	33.336364
Low-income	44.078788
Lower-middle-income	40.148485

```
Middle-income                                                    31.627273
Upper-middle-income                                              18.648485
```

[18]:
```
lp = sns.lineplot(data = stunted_cleaned, x = 'year', y = 'Stunting prevalence␣
 ↪among children under 5 years of age (% height-for-age <-2 SD), model-based␣
 ↪estimates',
             hue = 'income_group')
lp.set_ylabel('Percentage of child population stunted')
lp.set_title('Child Population Stunted By Income Group Across Time');
```



[21]:
```
wasting =  pd.read_csv('dighum wasting dataset.csv')
wasting_cleaned = isolate_vals(wasting)
wasting_cleaned = extract_percentage(wasting_cleaned, 'Wasted prevalence among␣
 ↪children under 5 years of age (%), model-based estimates')
wasting_cleaned['Wasted prevalence among children under 5 years of age (%),␣
 ↪model-based estimates'] = wasting_cleaned['Wasted prevalence among children␣
 ↪under 5 years of age (%), model-based estimates'].astype(float)
wasting_cleaned
```

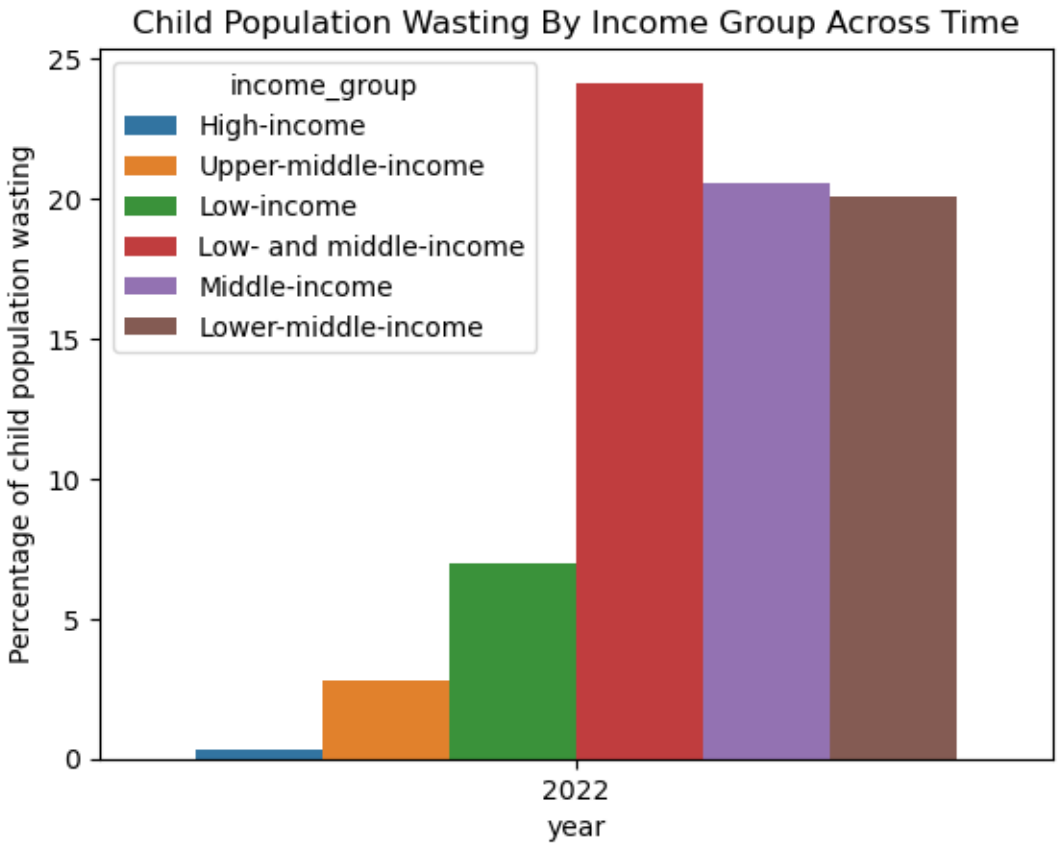
[21]:
```
               income_group  \
0               High-income
1      Upper-middle-income
2                Low-income
3   Low- and middle-income
4             Middle-income
5      Lower-middle-income
6               High-income
7      Upper-middle-income
8      Lower-middle-income
9             Middle-income
10  Low- and middle-income
11               Low-income

    Wasted prevalence among children under 5 years of age (%), model-based
estimates  \
0                                                   0.4
1                                                   2.1
2                                                   6.7
3                                                   6.9
4                                                   7.0
5                                                   9.6
6                                                   0.2
7                                                   3.5
8                                                  30.5
9                                                  34.1
10                                                 41.4
11                                                  7.3

    year
0   2022
1   2022
2   2022
3   2022
4   2022
5   2022
6   2022
7   2022
8   2022
9   2022
10  2022
11  2022
```

[32]:
```
lp = sns.barplot(data = wasting_cleaned, x = 'year', y = 'Wasted prevalence␣
 ↪among children under 5 years of age (%), model-based estimates',
                 errorbar = None, hue = 'income_group')
lp.set_ylabel('Percentage of child population wasting')
```

```
lp.set_title('Child Population Wasting By Income Group Across Time');
```



0.3 Vaccine Available Diseases Coverage

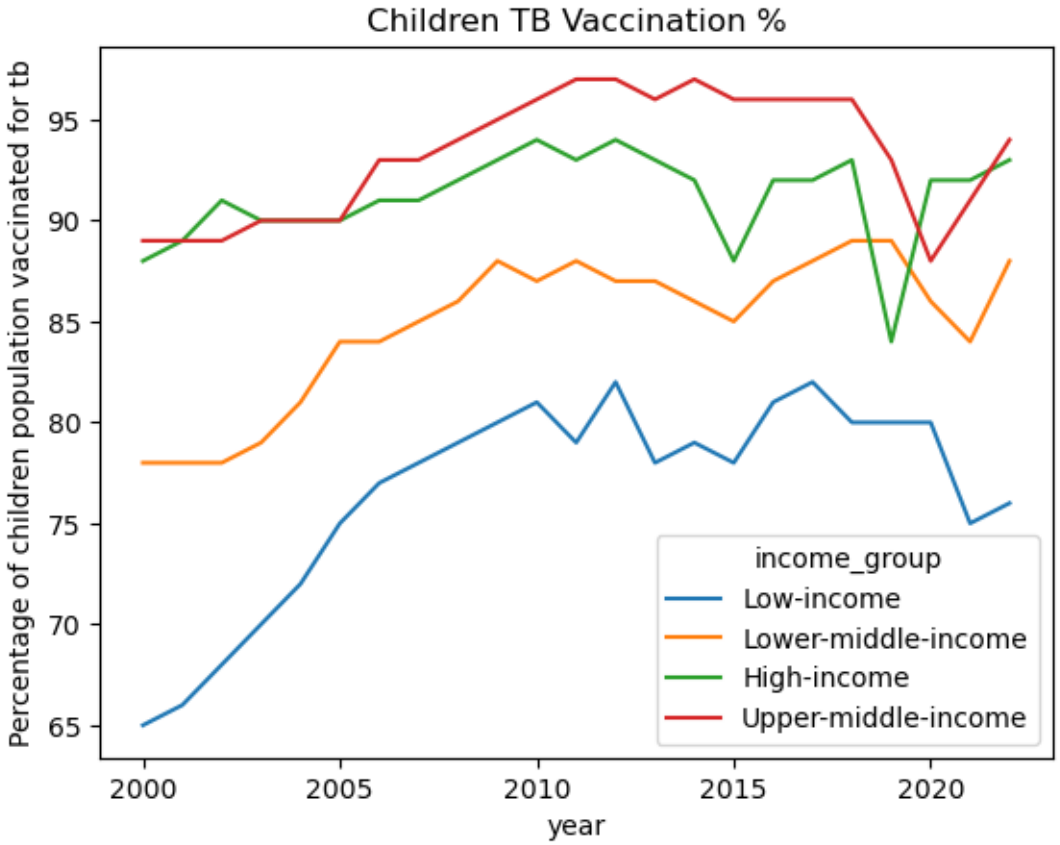
[33]:
```
tb = pd.read_csv('dighum tb dataset.csv')
cleaned_tb = isolate_vals(tb)
cleaned_tb
grouped = cleaned_tb.drop(columns = ['year']).groupby(['income_group']).mean()
grouped
```

[33]:

income_group	BCG immunization coverage among 1-year-olds (%)
High-income	91.173913
Low-income	76.565217
Lower-middle-income	84.869565
Upper-middle-income	93.260870

[34]:
```
lp = sns.lineplot(data = cleaned_tb, x = 'year', y = 'BCG immunization coverage␣
↪among 1-year-olds (%)',
```

```
            hue = 'income_group')
lp.set_ylabel('Percentage of children population vaccinated for tb')
lp.set_title('Children TB Vaccination %');
```

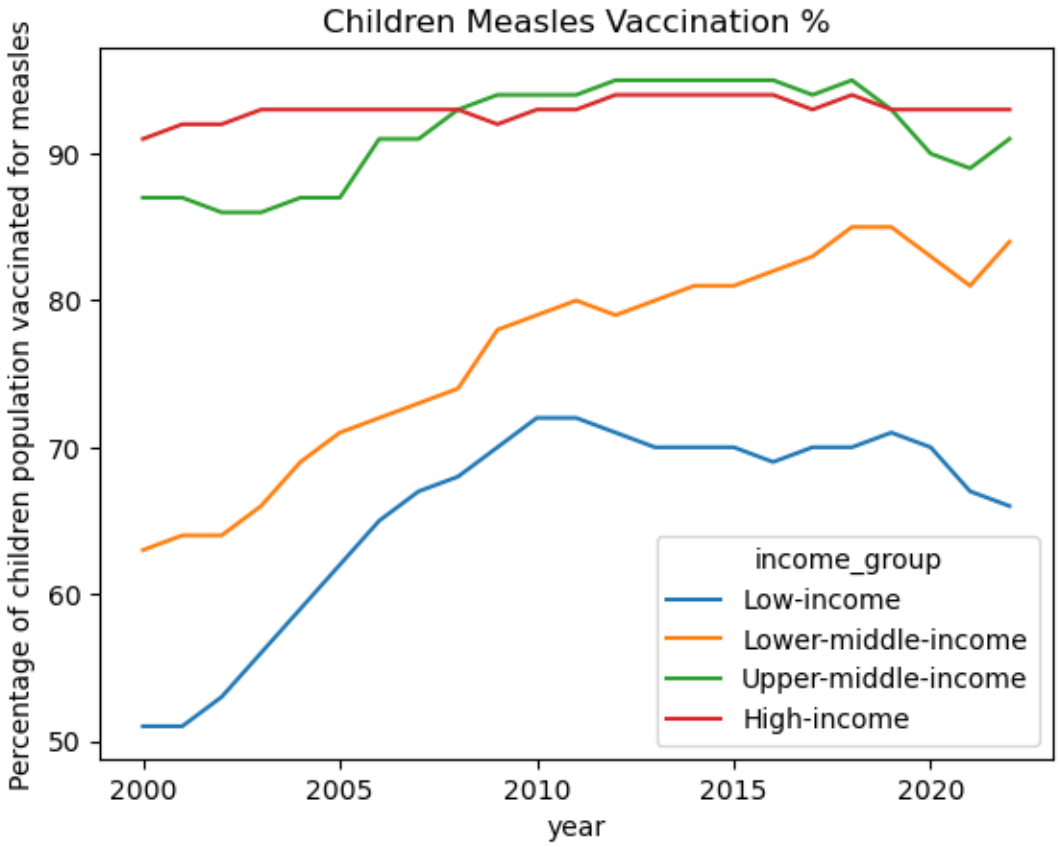


[35]:
```
measles = pd.read_csv('dighum mealesvacc dataset.csv')
measles_cleaned = isolate_vals(measles)
measles_cleaned
grouped = measles_cleaned.drop(columns = ['year']).groupby(['income_group']).
 ↪mean()
grouped
```

[35]:

income_group	Measles-containing-vaccine first-dose (MCV1) immunization coverage among 1-year-olds (%)
High-income	93.043478
Low-income	65.652174
Lower-middle-income	76.391304
Upper-middle-income	91.478261

[37]:
```
lp = sns.lineplot(data = measles_cleaned, x = 'year', y =␣
 ↪'Measles-containing-vaccine first-dose (MCV1) immunization coverage among␣
 ↪1-year-olds (%)',
            hue = 'income_group')
lp.set_ylabel('Percentage of children population vaccinated for measles')
lp.set_title('Children Measles Vaccination %');
```



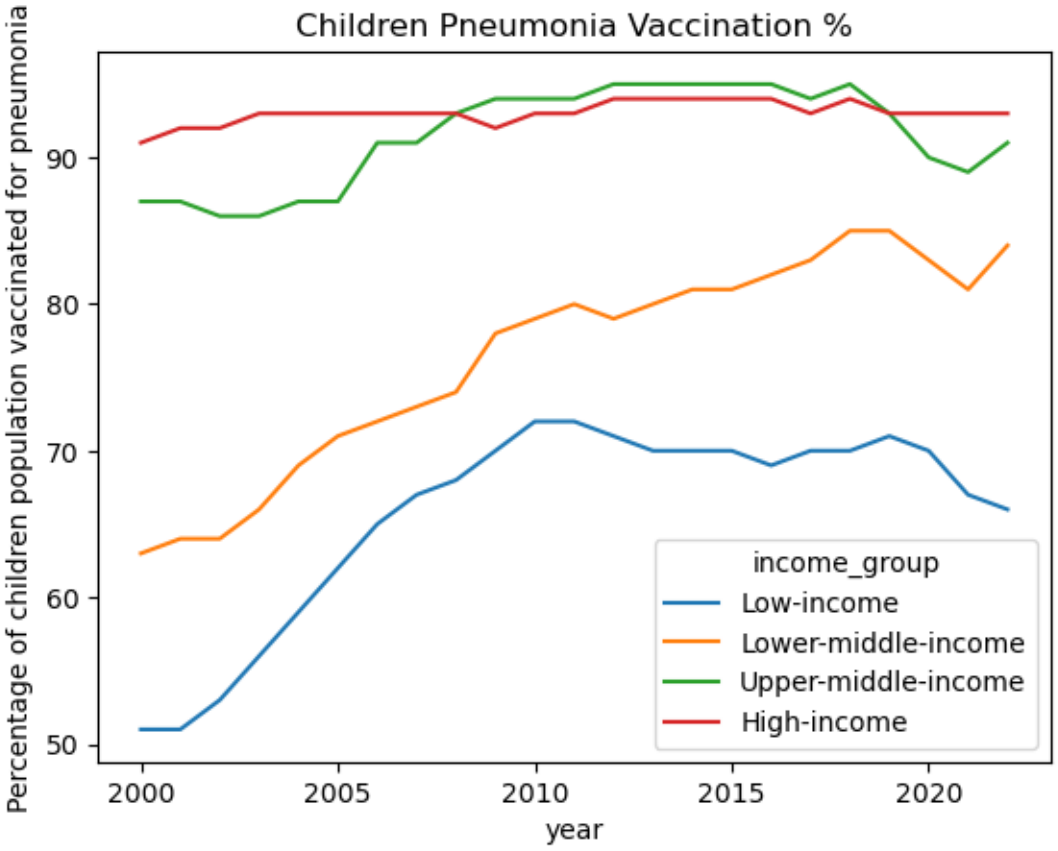
[38]:
```
pneum = pd.read_csv('dighum pneum dataset.csv')
pneum_cleaned = isolate_vals(measles)
pneum_cleaned
grouped = pneum_cleaned.drop(columns = ['year']).groupby(['income_group']).
 ↪mean()
grouped
```

[38]:

income_group	Measles-containing-vaccine first-dose (MCV1) immunization coverage among 1-year-olds (%)
High-income	93.043478
Low-income	65.652174
Lower-middle-income	76.391304

```
Upper-middle-income                                    91.478261
```

[39]:
```
lp = sns.lineplot(data = pneum_cleaned, x = 'year', y =␣
↪'Measles-containing-vaccine first-dose (MCV1) immunization coverage among␣
↪1-year-olds (%)',
             hue = 'income_group')
lp.set_ylabel('Percentage of children population vaccinated for pneumonia')
lp.set_title('Children Pneumonia Vaccination %');
```



[ ]: