THE UNIVERSITY OF CHICAGO
HARRIS SCHOOL OF PUBLIC POLICY

Final Project for Python Programming

Population Health and Economic Growth

Jeremy Xu, Kelsey Xing, Alex Zhao
University of Chicago Harris School of Public Policy

Research Question

Our research focuses on how the improvement in population health impacts economic growth. We hypothesized that as the mortality rate decreases and/or life expectancy increases, people should in turn have a higher flexibility to perform business and economic trade, thus resulting in economic expansion on a global basis.

Data Wrangling

For the data wrangling part, we retrieved data from the World Bank and FRED via data API. We included GDP per capita as the economic indicator, and male and female mortality rates, life expectancy as healthcare indicators. Since social determinants of health are critical factors that influence health outcomes, we also involved total population, urban population, population aged 65 and higher, health expenditure, internet access, and trade openness for each country in our research. We selected four representative countries - Brazil, India, Uganda, and the United States, based on their geographic regions as well as their income levels (World Bank country classifications). Two final data frames were produced, in which the time series data frame contains all the variables from 2000 to 2010 across these four countries, and the geographic data frame contains life expectancy data for all countries in 2010.

Plotting

To get a general overview on the spatial difference in population health, we first plotted a choropleth world map that indicates life expectancy for different countries in 2010. We found a salient disparity in life expectancy across the different continents. To further understand the dynamic pattern of population health, we used Seaborn to plot a bubble chart that demonstrates the relationship between life expectancy and GDP per capita. This chart provides some interesting insights on the health status that varies by countries and reveals the underlying association between health and economy.

We adapted our interactive charts in the Shiny. The panel named "Social Determinants of Health" displays heatmaps for the dynamic change of health and social science factors across four countries from 2000 to 2010. This allows us to have a rough idea on how these variables differ in each country and year before we conduct regression analysis. The panel named "Historical Data of GDP & Mortality" displays line plots for the change of mortality rate and GDP with different countries chosen. We saw similar patterns across countries, where GDP increases over the years, mortality rates decrease for both genders, and male mortality rate is consistently higher than that in females.

Regression Analysis

We conducted three Ordinary Least Squares(OLS) models on each country (Brazil, India, Uganda, and the United States) to estimate the impact of changes in male mortality rate, female mortality rate and life expectancy on GDP.

Our general model is

$$Y_t = \alpha + \beta X_t + \sum \lambda_{nt} \cdot X_{nt} + \varepsilon$$

$Y_t = (gdp_t - gdp_{t-1})/gdp_{t-1}$ is the growth rate of GDP per capita in the year t.
X_t is the male mortality rate, female mortality rate or life expectancy in year t.
X_{nt} is a vector of time-varying controls, including total population, proportion of urban residents, proportion of residents with internet access, proportion of the people aged 65 and more, government health expenditure, and openness to the global economy.

First, we tested two key assumptions of OLS regression and displayed the results on the panel named “Regression Analysis: Conditions Check” in shiny:

1. To examine the linearity and additivity of the relationship between dependent and independent variables, we plotted the predictors(male mortality rate, female mortality rate and life expectancy) against GDP. The scatter plots demonstrate that this assumption is satisfied by each country and each predictor.
2. To test the multi-collinearity between independent variables, we used chi-square tests of the predictors and the control variables. Except for the models of Brazil-male mortality, Brazil-female mortality, Uganda-male mortality and Uganda-female mortality, there is no significant relationship between the independent variables of every model.

The regression part contains the results of the regression analysis. From the results, we can conclude:

1. An increase in life expectancy in Brazil is estimated to increase GDP by $2,097.
2. An additional male mortality rate (per 1000) in India is estimated to reduce GDP by 126.

Other results are not significant, instead of recklessly concluding that there is no significant relationship between the rest of the predictor and GDPs, we believe that our regression model requires further refinement. A major challenge we faced in this part is to find a good instrument variable. We were aware that our model can probably induce endogeneity problems/reverse causality problems, but finally we failed to find a suitable IV to solve this problem.

Text Processing

For Text Processing, we used a webpage from the IMF, to conduct text sentiment analysis on its description for countries' backgrounds during the pandemic and policy responses. The point of this component in our project is to discover how countries in general are economically responding to the pandemic, which we believe is having a huge impact on population health in all countries. We decided to use the IMF's description for polarity value analysis as a benchmark, to somewhat evaluate different countries' performance while satisfying the requirement for this project.

The text processing part was conducted as follows: First, after finding permission for web scraping in IMF's terms of service, we scraped the text from its webpage using BeautifulSoup. Upon examination on its html codes, we managed to sort out texts for each country, and separate them into background description and policy responses. For the sake of clarity, we limited the target for our analysis to the G20 countries. Using NLP, we then produced 2 scatter plots for the polarity of IMF's description on each country and plotted them along their respective GDP per capita (data retrieved using API from the World Bank), in hopes to find a trend whether countries' performance in pandemic and their economic responses were correlated to their size of economy. One of the major challenges that we encountered in this part was the errors of the original webpage (in text and in codes). It took us a long time to finally come up with a sorting logic that works for the page. In future projects, we believe that leaving out more time in advance for web scraping tasks is a good solution to the problem.

The scatter plots did not show a convincing correlation. We conclude from the results that there is no correlation between the aforementioned factors. The plots as well as the original data frame used for plotting is shown on our Shiny webpage.

Conclusion

In conclusion, we find that countries with higher income levels and a larger GDP are likely to have a higher life expectancy. And the static plots show a negative relationship between male/female mortality rates and the GDP of countries at all income levels. Our regression model, however, only proves a small part of our hypothesis, with the majority of the results being insignificant. Finally, we find no evidence that countries' performance in pandemics and economic responses are correlated with their economic sizes. In future research, a more fitted regression model is expected, and a good instrument variable is needed to solve the current endogeneity problems and reverse causality problems.

```python
import os
import re
from pandas_datareader import wb
import pandas_datareader.data as web
import datetime
import numpy as np
import geopandas
import pycountry
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from mpl_toolkits.axes_grid1 import make_axes_locatable
from shiny import App, render, ui, reactive
from scipy.stats import chi2_contingency
import statsmodels.api as sm
from statsmodels.iolib.summary2 import summary_col
import requests
from bs4 import BeautifulSoup
import spacy
from spacytextblob.spacytextblob import SpacyTextBlob


### manually change path
path = r'/Users/jeremyxu/Desktop/Sample Project'
os.chdir(path)


### data wrangling - time series data
## retrieve world bank data api
wb_indicator = ['NY.GDP.PCAP.CD', 'SP.DYN.AMRT.MA', 'SP.DYN.AMRT.FE', 'SP.DYN.LE00.IN',
                'SP.POP.TOTL', 'SP.URB.TOTL.IN.ZS', 'SH.XPD.CHEX.GD.ZS', 'IT.NET.USER.ZS',
                'SP.POP.65UP.TO.ZS']
wb_columns = ['gdp per capita', 'male mortality rate (per 1000)', 'female mortality rate (per 1000)',
              'life expectancy', 'total population', 'urban population (% of population)',
              'health expenditure (% of GDP)', 'internet access (% of population)',
              'age 65+ (% of population)']
wb_data = wb.download(indicator=wb_indicator, country=['US', 'IN', 'BR','UG'], start=2000, end=2010)
# change column names
wb_data.columns = wb_columns
wb_data = wb_data.reset_index()
wb_data['year'] = wb_data['year'].astype(int)

## retrieve fred data api
start = datetime.datetime(2000, 1, 1)
end = datetime.datetime(2010, 12, 31)
fred_indicator = ['OPENRPUSA156NUPN','OPENRPBRA156NUPN','OPENRPINA156NUPN', 'OPENRPUGA156NUPN']
countries = ['United States','Brazil','India', 'Uganda']
fred_data = web.DataReader(fred_indicator, 'fred', start, end)
# change column names
fred_data.columns = countries
fred_data.index.names = ['year']
fred_data = fred_data.reset_index()
# reshape dataframe
fred_data['year'] = fred_data['year'].dt.year
fred_data = fred_data.melt(id_vars='year', value_vars=None,
                           var_name='country', value_name='trade openness (%)')

## merge and prepare dataframe
data = wb_data.merge(fred_data, on=['country', 'year'], how='outer')
# unify indicator units
data['male mortality rate (%)'] = (data['male mortality rate (per 1000)'] / 10)
data['female mortality rate (%)'] = (data['female mortality rate (per 1000)'] / 10)

## export final timeseries dataframe
data.to_csv(os.path.join(path, 'data', 'final_timeseries_dataframe.csv'), index=False)
```

```python
### data wrangling - geographic data
## retrieve world bank age 65 data api
wb_data_age = wb.download(indicator='SP.DYN.LE00.IN', country='all', start=2010, end=2010)
wb_data_age.columns = ['life expectancy']
wb_data_age = wb_data_age.reset_index()
wb_data_age['year'] = wb_data_age['year'].astype(int)
# create and apply a function to match country name with country iso code
def do_fuzzy_search(country):
    try:
        result = pycountry.countries.search_fuzzy(country)
        return result[0].alpha_3
    except:
        return np.nan
iso_map = {country: do_fuzzy_search(country) for country in wb_data_age["country"]} # create dictionary
wb_data_age["iso_a3"] = wb_data_age["country"].map(iso_map) # mapping country iso code

## retrieve world geometry data
world = geopandas.read_file(geopandas.datasets.get_path('naturalearth_lowres'))
world = world[['iso_a3', 'geometry']]

## merge and prepare dataframe
geo_data = wb_data_age.merge(world, on=['iso_a3'], how='inner')
geo_data = geo_data.dropna() # drop missing data

## export final geographic dataframe
geo_data.to_csv(os.path.join(path, 'data', 'final_geographic_dataframe.csv'), index=False)


### plotting the map - life expectancy in different countries
# import data
world = geopandas.read_file(geopandas.datasets.get_path('naturalearth_lowres'))
geo_data = pd.read_csv(os.path.join(path, 'data', 'final_geographic_dataframe.csv'))
```

```
geo_population = geopandas.GeoDataFrame(geo_data,
                                        geometry=geopandas.GeoSeries.from_wkt(geo_data['geometry']),
                                        crs="EPSG:4326")
# plot data
fig, ax = plt.subplots(figsize=(8, 5), dpi=300)
world[world.continent != 'Antarctica'].plot(ax=ax, color='whitesmoke', edgecolor='black')
divider = make_axes_locatable(ax)
cax = divider.append_axes('right', size='5%', pad=0.1)
geo_population.plot(ax=ax, column='life expectancy',
                    legend=True, cmap='Blues', cax=cax)
ax.set_title('Life Expectancy for Different Countries in 2010 Map')
ax.axis('off')
# export figure
fig.savefig(os.path.join(path, 'images', 'static plot 1.png'))


### plotting the bubble plot - life expectancy
# import data
data = pd.read_csv(os.path.join(path, 'data', 'final_timeseries_dataframe.csv'))
data_health = data[['country', 'year', 'life expectancy', 'health expenditure (% of GDP)']]
# plot data
fig, ax = plt.subplots(figsize=(8, 5), dpi=300)
sns.scatterplot(data=data, x="gdp per capita", y="life expectancy",
                size="health expenditure (% of GDP)",
                hue="country", palette="viridis", alpha=0.8, sizes=(20, 400))
ax.set_xlabel('GDP per Capita')
ax.set_ylabel('Life Expectancy')
ax.set_title('Plot of Life Expectancy and GDP per Capita for Different Countries')
# export figure
fig.savefig(os.path.join(path, 'images', 'static plot 2.png'))


### data analysis - regression
```

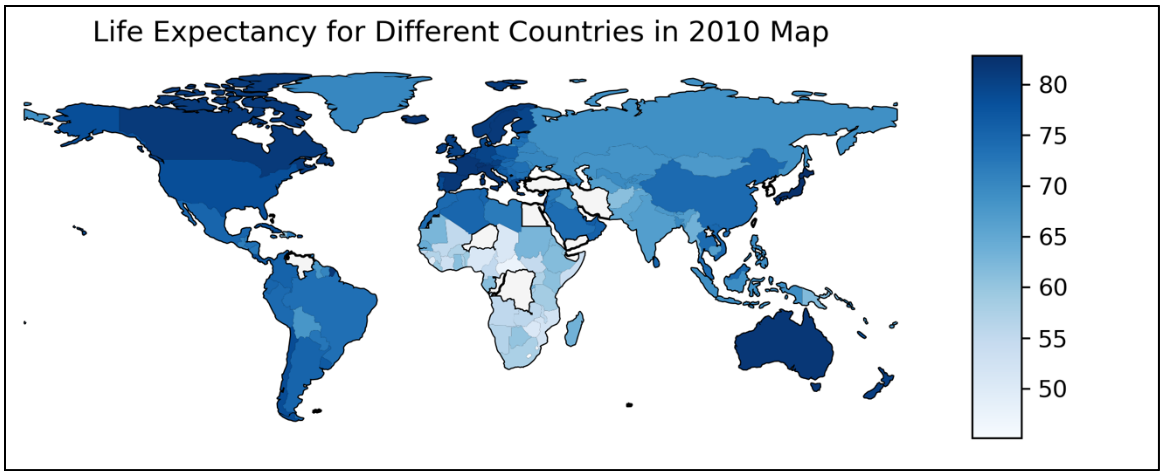
```python
data = pd.read_csv(os.path.join(path, 'data', 'final_timeseries_dataframe.csv'))
## plot to check for linearity
predictor = ['male mortality rate (per 1000)',
             'female mortality rate (per 1000)',
             'life expectancy']
controls = ['total population', 'urban population (% of population)',
            'health expenditure (% of GDP)', 'internet access (% of population)',
            'age 65+ (% of population)', 'trade openness (%)']

## OLS regression
def regression(country):
    y = data[data.country == country]['gdp per capita']
    x1 = data[data.country == country][['male mortality rate (per 1000)'] + controls]
    x2 = data[data.country == country][['female mortality rate (per 1000)'] + controls]
    x3 = data[data.country == country][['life expectancy'] + controls]

    res_ols1 = sm.OLS(y, sm.add_constant(x1)).fit()
    res_ols2 = sm.OLS(y, sm.add_constant(x2)).fit()
    res_ols3 = sm.OLS(y, sm.add_constant(x3)).fit()

    result = summary_col(
        [res_ols1, res_ols2, res_ols3],
        model_names=['x1', 'x2', 'x3'],
        stars=True, regressor_order=['const', 'male mortality rate (per 1000)',
                                     'female mortality rate (per 1000)', 'life expectancy'],
        info_dict={'': lambda x: '',
                   'Observation': lambda x: str(int(x.nobs))}
    )
    return result

regression('United States')
regression('Brazil')
regression('India')
regression('Uganda')


### text processing
## function for cleaning sentence strings in soup text
def clean_str(s):
    s = s.replace('\n    ', ' ').replace('\n', ' ').replace('\xa0', ' ')
    s = s.replace('   ', ' ').replace('  ', ' ').replace('. .', '.').strip()
    return s

def get_soup(link):
    response = requests.get(link)
    soup = BeautifulSoup(response.text, 'lxml')
    return soup

## function for identifying the part of the imf policy tracker
def sort_description(l):
    bg = []
    policy = []
    separator_index = None
    for t in l:
        # find 'key policy responses' string to use as separator
        if re.search(r'K.. P..... R........ .+', t):
            separator_index = l.index(t)

    for t in l:
        # those above 'key policy responses' header are bg info on webpage
        if 0 <= l.index(t) < separator_index:
            bg.append(t)
        # those below are policy responses
        elif separator_index < l.index(t) < len(l):
            policy.append(t)
    description = [bg, policy]
    return description

## function for extracting target texts from soup
def get_text(headers):
    txt_by_country = []
    # use country name headers as key to sort text in soup
    for h in headers:
        if headers.index(h) != len(headers) - 1:
            txt = []
            for sib in h.next_siblings:
                if sib in headers[headers.index(h) + 1].previous_siblings:
                    # leave out redundant lines
                    if sib.name not in ['br', 'h2'] and sib.text != '\n':
                        txt.append(clean_str(sib.text))
                else:
                    break
            txt_by_country.append(txt)
        else:  # for the last header
            txt = []
            for sib in h.next_siblings:
                if sib.name not in ['br', 'h2'] and sib.text != '\n':
                    txt.append(clean_str(sib.text))
            txt_by_country.append(txt)
    # put repeated text for Vanuatu to correct place
    txt_by_country[185] = txt_by_country[177]
    # combine 2 elements due to html coding error from the original webpage
    txt_by_country[130] += txt_by_country[131]
    txt_by_country.pop(177)
    txt_by_country.pop(131)
    # divide each country's text into bg and policy
    clean_text = [sort_description(l) for l in txt_by_country]
    return clean_text
```

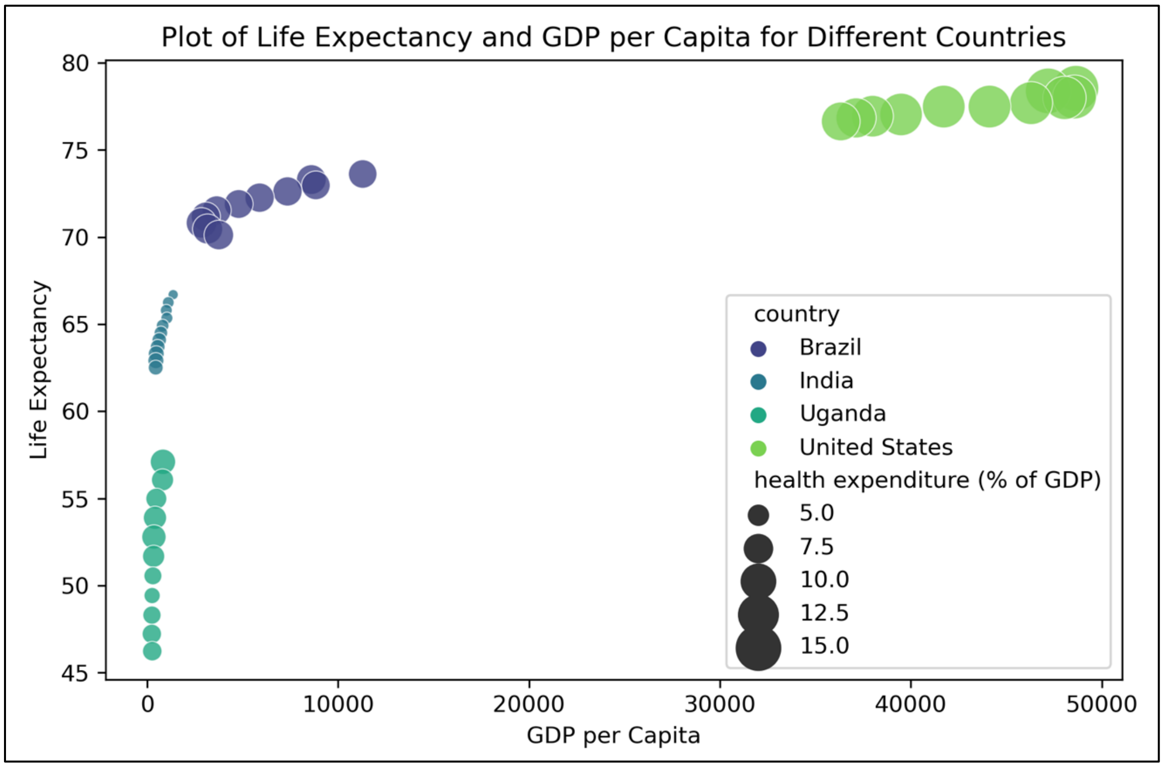
```
## function for getting average polarity value of a chosen list of text
def get_sent(l):
    headers = ['Fiscal', 'Monetary and macro-financial',
                'Exchange rate and balance of payments', 'Back to Top']
    values = [nlp(t)._.blob.polarity for t in l if t not in headers]
    pol = np.mean(values)
    return pol

## function for indicator column in dataframe
def get_reopen_indicator(l):
    indicator = 'Economy not reopened'
    for t in l[0]:
        # if mentioned reopening, then the country is reopened
        if re.search(r'.eopening of the .conomy', t):
            indicator = 'Economy reopened'
    return indicator

## create dataframe of polarity values by web-scraping
def get_df(headers):
    # for clarity of dataframe and plot, limited to g20 countries
    g20_countries = ['AR', 'AU', 'BR', 'CA', 'CN', 'DE', 'FR', 'GB', 'ID', 'IN', 'IT',
                     'JP', 'KR', 'MX', 'RU', 'SA', 'TR', 'US', 'ZA']

    g20_names = [pycountry.countries.get(alpha_2=c).name for c in g20_countries]

    df = wb.download(indicator='NY.GDP.PCAP.CD', country=g20_countries, start=2020, end=2020)
    df.columns = ['gdp per capita']
    df = df.reset_index()
    df['year'] = df['year'].astype(int)

    g20_index = []
    for name in g20_names:
        for h in headers:
            if h.text == name or h.text.startswith(name) or name.startswith(h.text):
                g20_index.append(headers.index(h))

    sentiment_val = []
    reopening_indicator = []
    for i in g20_index:
        for l in sorted_text:
            if sorted_text.index(l) == i:
                sentiment_val.append([get_sent(txt_list) for txt_list in l])
                reopening_indicator.append(get_reopen_indicator(l))

    df = df.join(pd.DataFrame(sentiment_val, columns=['bg_polarity', 'policy_polarity'],
                              dtype=float))
    df = df.join(pd.DataFrame(reopening_indicator, columns=['reopen_ind']))
    df['country'] = df['country'].str.replace('Turkiye', 'Turkey')

    return df

## function for sentiment value plotting
def plot_sent(df, y):
    y_dict = {'bg_polarity': "Polarity values of G20 countries' background description",
              'policy_polarity': "Polarity values of G20 countries' policy description"}
    inserted_text = y_dict[y]

    fig, ax = plt.subplots(dpi=300)
    p1 = sns.scatterplot(data=df, x=df['gdp per capita'], y=df[y], hue=df['reopen_ind'])

    plt.legend(loc='best')
    # add country name label to markers
    for line in range(0, df.shape[0]):
        p1.text(df['gdp per capita'][line] + 0.01, df[y][line],
                df['country'][line], horizontalalignment='left',
                size='xx-small', color='black', weight='normal')
    ax.set_xlabel('GDP per capita', fontsize=7)
    ax.set_ylabel('Sentiment values on polarity', fontsize=7)
    ax.set_title(inserted_text + " by IMF's COVID Policy Response database, 2021", fontsize=8)
    fig.savefig(os.path.join(path, 'images', 'txt processing_' + inserted_text + ' plot.png'))


nlp = spacy.load('en_core_web_sm')
nlp.add_pipe('spacytextblob')

url = r'https://www.imf.org/en/Topics/imf-and-covid19/Policy-Responses-to-COVID-19'
soup = get_soup(url)

# sort text in soup and clean the header list for easy access on country names
article = soup.find('article')
h3 = article.find_all('h3')
sorted_text = get_text(h3)
h3.pop(177)
h3.pop(131)

# get df for plotting and export
data = get_df(h3)
data.to_csv(os.path.join(path, 'data', 'final_text_processing_dataframe.csv'), index=False)

# plot and export
plot_sent(data, 'bg_polarity')
plot_sent(data, 'policy_polarity')


### Shiny
data = pd.read_csv(os.path.join(path, 'data', 'final_timeseries_dataframe.csv'))
text_df = pd.read_csv(os.path.join(path, 'data', 'final_text_processing_dataframe.csv'))

# plotting interactive plot with Shiny
```

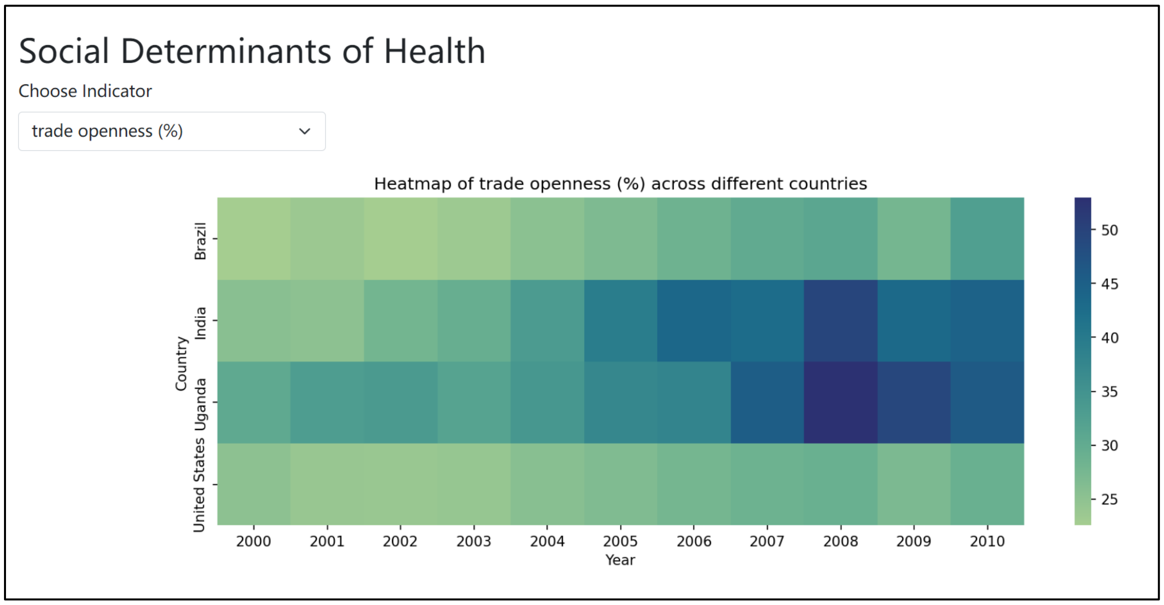
```
population_indicator = ['trade openness (%)', 'health expenditure (% of GDP)',
                        'internet access (% of population)',
                        'urban population (% of population)']
countries = ['United States', 'Brazil', 'India', 'Uganda']

predictor = ['male mortality rate (per 1000)',
             'female mortality rate (per 1000)',
             'life expectancy']
controls = ['total population', 'urban population (% of population)',
            'health expenditure (% of GDP)', 'internet access (% of population)',
            'age 65+ (% of population)', 'trade openness (%)']

description_category = ['Background Description', 'Policy Description']

app_ui = ui.page_fluid(
    ui.navset_pill_list(
        ui.nav('Social Determinants of Health',
               ui.h2("Social Determinants of Health"),
               ui.input_select(id="determinants", label="Choose Indicator",
                               choices=population_indicator),
               ui.output_plot('data_plot')
               ),
        ui.nav('Historical Data of GDP & Mortality',
               ui.h2("Historical Data of GDP & Mortality"),
               ui.input_select(id='gdp_mortality', label='Choose Country',
                               choices=countries),
               ui.output_plot('gdp_mortality_plot')
               ),
        ui.nav('Regression Analysis: Conditions Check',
               ui.h2('Regression Analysis: Conditions Check'),
               ui.input_select(id='country', label='Choose Country',
                               choices=countries),
               ui.input_radio_buttons(id="predictor", label='Choose Predictor',
                                        choices=predictor),
                    ui.h3('Linearity Check'),
                    ui.output_plot('linearity_plot'),
                    ui.h3('Chi-square Test'),
                    ui.output_text("chi_sq"),
                    ),
              ui.nav('Text Processing',
                    ui.br(),
                    ui.row(
                        ui.h2('Text Processing'),
                        ui.hr(),
                        ui.h3("Polarity Values of IMF's Description on \
                              G20 Countries' COVID Policy Response"),
                        ui.h5(
                            "GDP per capita low to high from left to right; \
                                Collected from IMF's webpage, last updated in 2021"
                        ),
                        ui.input_select(id='description', label='Choose Description Category',
                                        choices=description_category)
                    ),
                    ui.row(ui.output_image('polarity_plot')),
                    ui.row(
                        ui.br(),
                        ui.br(),
                        ui.br(),
                        ui.h5('Appendix: Original dataframe of the above plots'),
                        ui.output_table('g20_info')
                    )
                    )
          )
      )
```

```
def server(input, output, session):
    @reactive.Calc
    def get_population():
        population = pd.read_csv(os.path.join(path, 'data', 'final_timeseries_dataframe.csv'))
        return population

    @output
    @render.plot
    def data_plot():
        population = get_population()
        fig, ax = plt.subplots(dpi=300)
        population = population[['country', 'year', f'{input.determinants()}']]
        population = population.pivot('country', 'year', f'{input.determinants()}')
        sns.heatmap(population, cmap='crest')
        ax.set_xlabel('Year')
        ax.set_ylabel('Country')
        ax.set_title(f'Heatmap of {input.determinants()} across different countries')
        return ax

    @output
    @render.plot
    def gdp_mortality_plot():
        fig, ax1 = plt.subplots(dpi=300)
        ax1.grid(True)
        sns.lineplot(data=data[data.country == input.gdp_mortality()],
                     ax=ax1, x="year", y='male mortality rate (per 1000)',
                     markers=True,
                     label='male mortality rate', color='navy')
        sns.lineplot(data=data[data.country == input.gdp_mortality()],
                     ax=ax1, x="year", y='female mortality rate (per 1000)',
                     markers=True,
                     label='female mortality rate', color='orange')
        ax2 = ax1.twinx()
        sns.lineplot(data=data[data.country == input.gdp_mortality()],
                     ax=ax2,
                     x="year",
                     y='gdp per capita',
                     markers=True,
                     label='GDP', color='green', linestyle='--')
        ax1.legend(loc=6)
        ax2.legend(loc=7)
        ax1.set_xlabel('Year')
        ax1.set_ylabel('Mortality Rate(per 1000)')
        ax2.set_ylabel('GDP')
        ax1.set_title(f'GDP & Mortality Data of {input.gdp_mortality()} from 2000 to 2010')
        return ax1

    @output
    @render.plot
    def linearity_plot():
        fig, ax = plt.subplots(figsize=(8, 5))
        sns.scatterplot(data=data[data.country == input.country()],
                        x=input.predictor(), y='gdp per capita', ax=ax)
        ax.set_xlabel(input.predictor().capitalize())
        ax.set_ylabel('GDP Per Capita')
        ax.set_title('GDP Per Capita Vs ' + f'{input.predictor()}'.capitalize())
        return ax

    @output
    @render.text
    def chi_sq():
        variables = data[data.country == input.country()][
            [input.predictor(), 'total population', 'urban population (% of population)',
             'health expenditure (% of GDP)', 'internet access (% of population)',
             'age 65+ (% of population)', 'trade openness (%)']]
        # H0: No relationship between categorical variables
        stat, p, dof, expected = chi2_contingency(variables)

        alpha = 0.05
        if p <= alpha:
            result = f'Result: p value is {str(p)}. \
                The categorical variables are dependent (reject H0)'
        else:
            result = f'Result: p value is {str(p)}. \
                No significant relationship between categorical variables (H0 holds true)'
        return result

    @output
    @render.image
    def polarity_plot():
        if input.description() == 'Background Description':
            return {
                'src': os.path.join(path, 'images',
                                    "txt processing_Polarity values of \
                                        G20 countries' background description plot.png"),
                'contentType': 'image/png',
                'width': '50%'
            }
        else:
            return {
                'src': os.path.join(path, 'images',
                                    "txt processing_Polarity values of \
                                        G20 countries' policy description plot.png"),
                'contentType': 'image/png',
                'width': '50%'
            }

    @output
    @render.table
    def g20_info():
        return text_df

app = App(app_ui, server)
```

Appendix

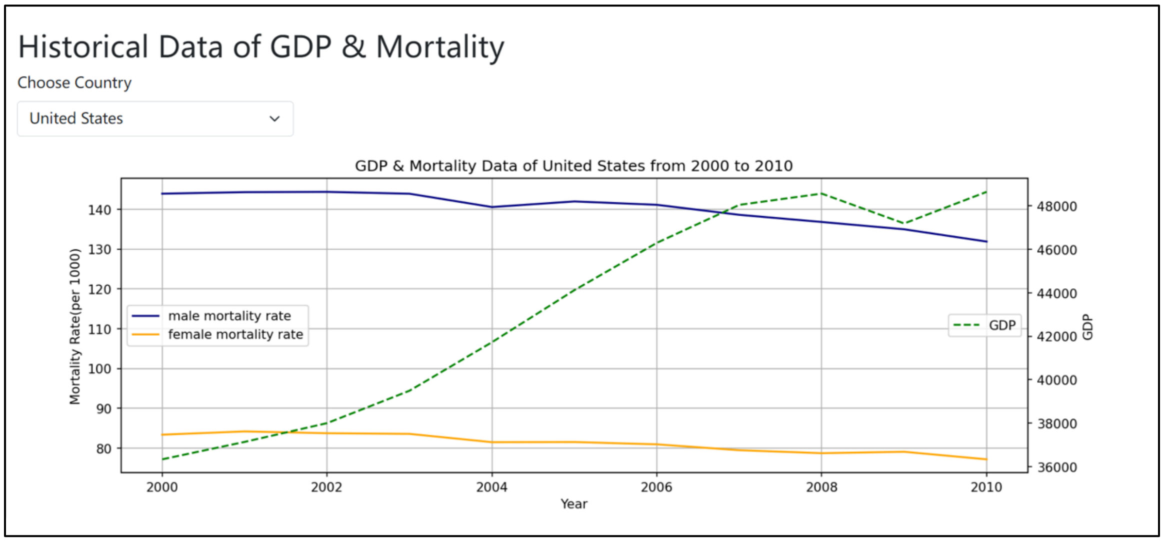

Life Expectancy for Different Countries in 2010 Map
80
75
70
65
60
55
50


Plot of Life Expectancy and GDP per Capita for Different Countries
Life Expectancy
GDP per Capita
80
75
70
65
60
55
50
45
0
10000
20000
30000
40000
50000
country
Brazil
India
Uganda
United States
health expenditure (% of GDP)
5.0
7.5
10.0
12.5
15.0


Social Determinants of Health

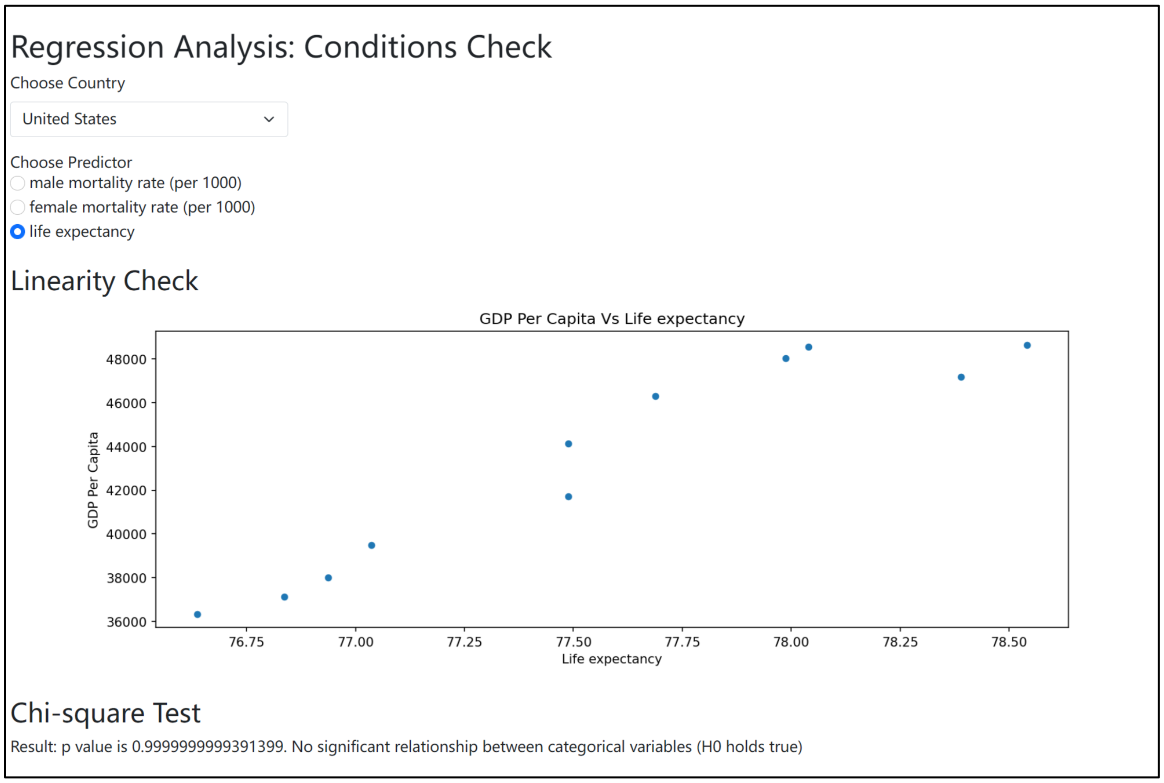
Choose Indicator

trade openness (%)


Historical Data of GDP & Mortality

Choose Country

United States



Regression Analysis: Conditions Check

Choose Country

United States

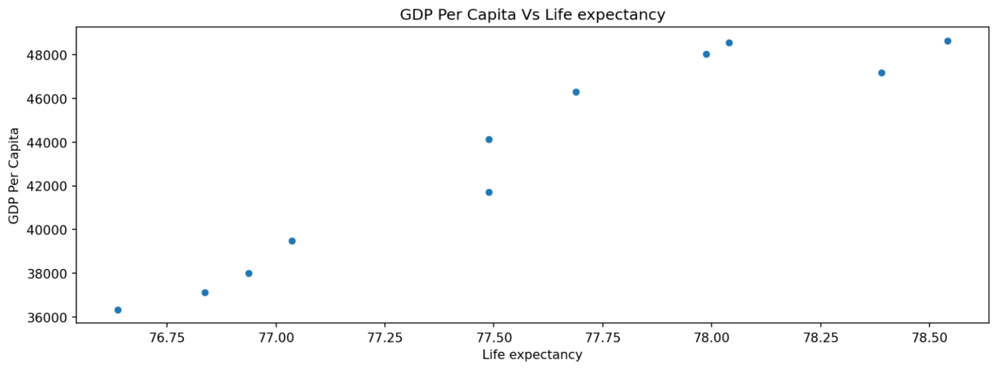
Choose Predictor

male mortality rate (per 1000)

female mortality rate (per 1000)

[✓ life expectancy]

Linearity Check



Chi-square Test

Result: p value is 0.9999999999391399. No significant relationship between categorical variables (H0 holds true)

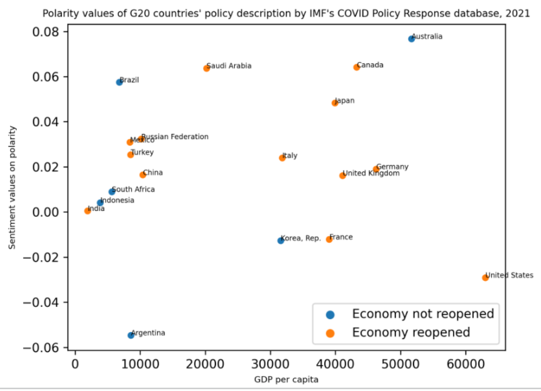
Text Processing

Polarity Values of IMF's Description on G20 Countries' COVID Policy Response

GDP per capita low to high from left to right; Collected from IMF's webpage, last updated in 2021

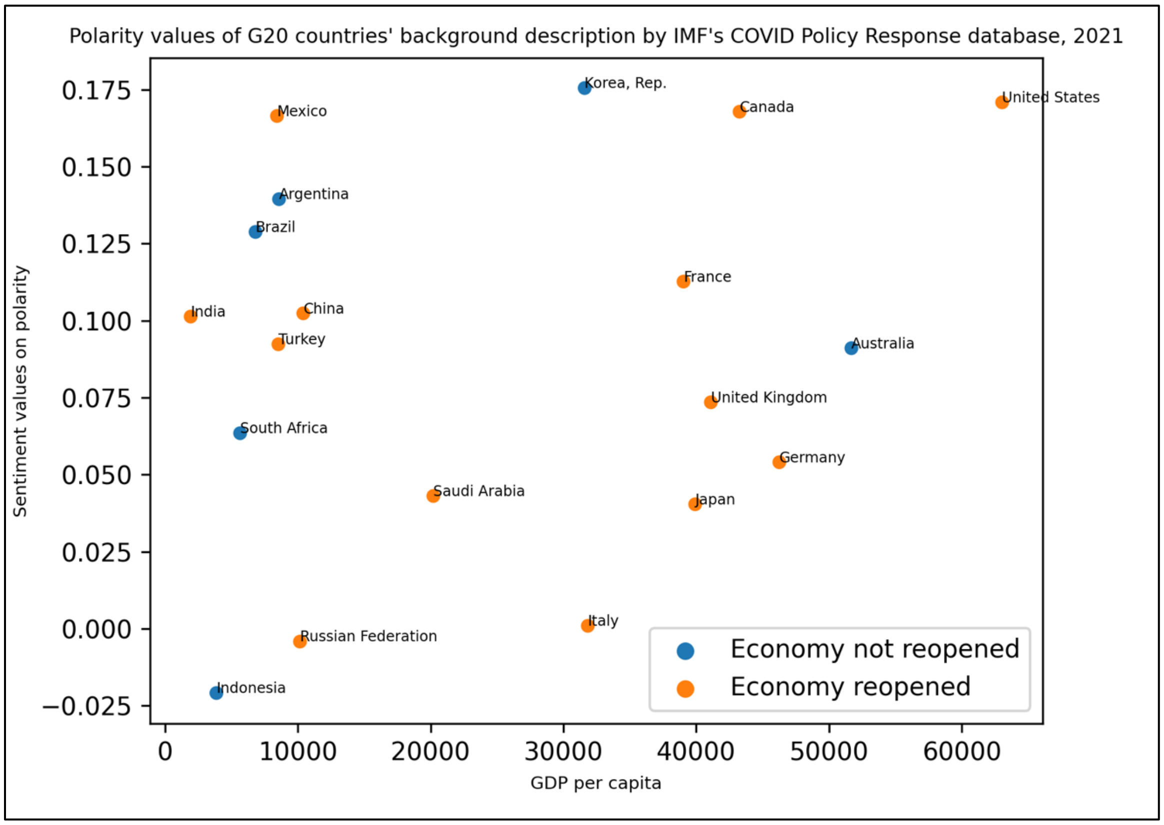
Choose Description Category

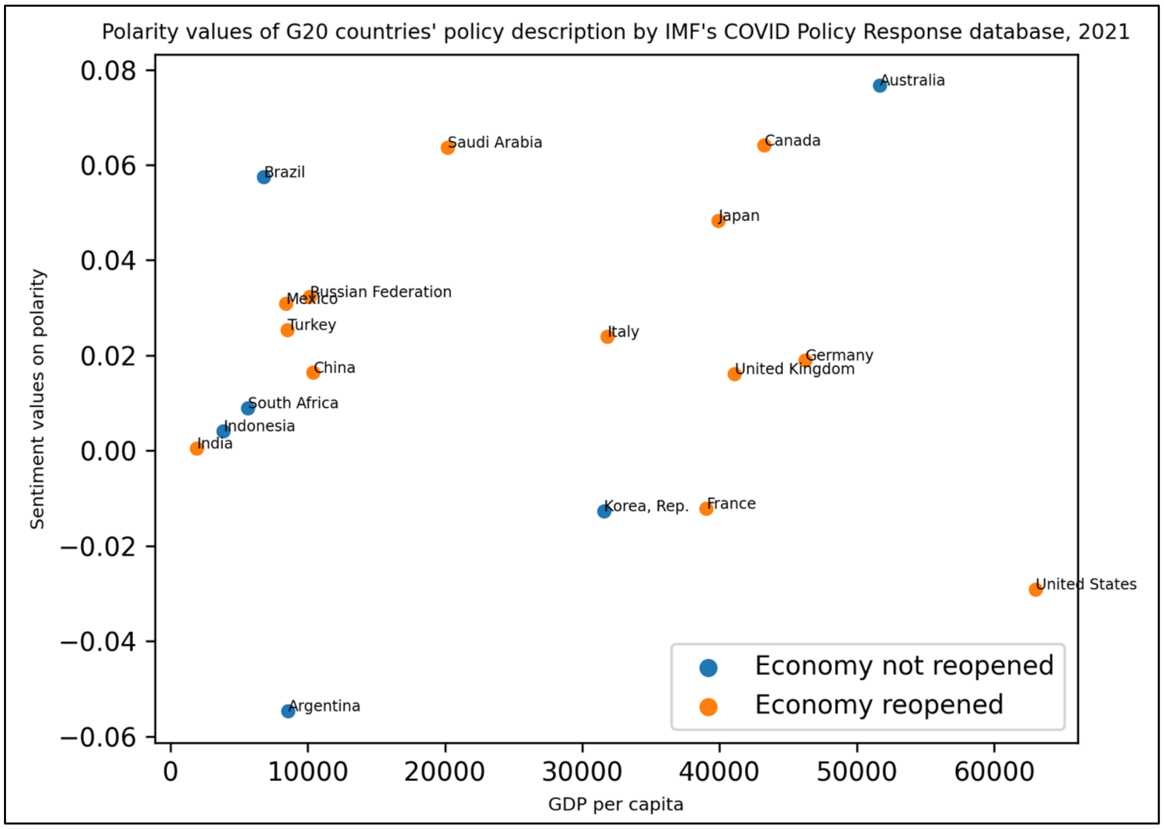
Policy Description



Appendix: Original dataframe of the above plots

country	year	gdp per capita	bg_polarity	policy_polarity	reopen_ind
Argentina	2020	8585.694742	0.139430	-0.054729	Economy not reopened
Australia	2020	51680.316523	0.091050	0.076653	Economy not reopened
Brazil	2020	6814.875632	0.128788	0.057431	Economy not reopened
Canada	2020	43258.263872	0.167824	0.064094	Economy reopened
China	2020	10408.669756	0.102369	0.016376	Economy reopened
Germany	2020	46252.689304	0.054015	0.018928	Economy reopened
France	2020	39037.122631	0.112670	-0.012202	Economy reopened
United Kingdom	2020	41098.078653	0.073486	0.016063	Economy reopened


Polarity values of G20 countries' background description by IMF's COVID Policy Response database, 2021
Sentiment values on polarity
GDP per capita
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
−0.025
0
10000
20000
30000
40000
50000
60000
Korea, Rep.
Mexico
Canada
United States
Argentina
Brazil
France
India
China
Turkey
Australia
United Kingdom
South Africa
Germany
Saudi Arabia
Japan
Italy
Russian Federation
Indonesia
● Economy not reopened
● Economy reopened


Polarity values of G20 countries' policy description by IMF's COVID Policy Response database, 2021
Sentiment values on polarity
GDP per capita
0.08
0.06
0.04
0.02
0.00
−0.02
−0.04
−0.06
0
10000
20000
30000
40000
50000
60000
Australia
Saudi Arabia
Canada
Brazil
Japan
Russian Federation
Mexico
Turkey
Italy
Germany
China
United Kingdom
South Africa
Indonesia
India
Korea, Rep.
France
United States
Argentina
● Economy not reopened
● Economy reopened