World Data Visualization

Geographical, Population, Economic Data

BigData Programming Final Project
Student Number: 20160796
BoSung Park

June 13, 2019

Project Purpose

Purpose and overall outline

TOPIC

Data Crawling and Visualization of World Data

각종 전 세계 데이터 크롤링 및 시각화 작업

Data: Demographic, Geographic, Economic

Detailed Description (6)

▶Demographic: Age structure, Median Age

▶Geographic: Land usage, Type of Land(water, land)

▶Economic: GDP Composition, Unemployment Rate

Crawling Source

index mundi

United States Government Profile 2018

Home > Factbook > Countries > United States

Country name	**conventional long form:** United States of America **conventional short form:** United States **abbreviation:** US or USA **etymology:** the name America is derived from that of Amerigo VESPUCCI (1454-1512) - Italian explorer, navigator, and cartographer - using the Latin form of his name, Americus, feminized to America
Government type	constitutional federal republic
Capital	**name:** Washington, DC **geographic coordinates:** 38 53 N, 77 02 W **time difference:** UTC-5 (during Standard Time) **daylight saving time:** +1hr, begins second Sunday in March; ends first Sunday in November **note:** the 50 United States cover six time zones
Administrative divisions	50 states and 1 district*; Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, District of Columbia*, Florida, Georgia, Hawaii, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming
Dependent areas	American Samoa, Baker Island, Guam, Howland Island, Jarvis Island, Johnston Atoll, Kingman Reef, Midway Islands, Navassa Island, Northern Mariana Islands, Palmyra Atoll, Puerto Rico, Virgin Islands, Wake Island **note:** from 18 July 1947 until 1 October 1994, the US administered the Trust Territory of the Pacific Islands; it entered into a political relationship with all four political entities: the Northern Mariana Islands is a commonwealth in political union with the US (effective 3 November 1986); the Republic of the Marshall Islands signed a Compact of Free Association with the US (effective 21 October 1986); the Federated States of Micronesia signed a Compact of Free Association with the US (effective 3 November 1986); Palau concluded a Compact of Free Association with the US (effective 1 October 1994)

Indexmundi.com

- Updated data from CIA World Factbook
- Not enough visualized graph
- Cannot compare multiple countries' data
  Only provide individual country's data in separate page

Project Progress Flow

Project flow

URL Collection

Input: CountryNameCode.csv

1.'https://www.indexmundi.com/'+str(mtitle)+'/demographics_profile.html'

2.'https://www.indexmundi.com/'+str(mtitle)+'/age_structure.html'

3.'https://www.indexmundi.com/'+str(mtitle)+'/unemployment_rate.html'

4.'https://www.indexmundi.com/'+str(mtitle)+'/economy_profile.html'

5.'https://www.indexmundi.com/'+str(mtitle)+'/geography_profile.html'

Read country names from input file and form 150 url

Data Web Crawling

Country Data Crawling(population, geography, economy).ipynb

class population crawling:
- def median_age()
- def age structure()

class economy_crawling:
- def Unemployment()
- def GDP_composition()

Class geography_crawling:
- def area_comp()
- def land_use()

Checktime(Decorator)
WrongDataRecord(Exception)

Data Visualization

Input File

1. median_age.csv
2. age_structure.csv
3. unemployment.csv
4. GDP_composition.csv
5. area_comp.csv
6. land_use.csv

Dataframe

1. Age_structure
2. Age_structure1
3. Land_use2
4. Area_comp2
5. GDP_comp2
6. unemployment2

7 charts

- seaborn
- plotly

Project Code Review

Project code and result

Data Crawling

CountryNameCode.csv

```
malaysia
japan
india
china
thailand
vietnam
cambodia
singapore
indonesia
canada
united_states
united_kingdom
germany
france
italy
spain
philippines
south_korea
mexico
austria
poland
portugal
egypt
denmark
australia
cambodia
belgium
sudan
russia
norway
```

```
class population_crawling:

    def age_structure():
        with open('age_structure.csv', 'w', encoding='UTF-8', newline='') as f:
            writer = csv.writer(f)
            writer.writerow( ['country name','0-14', '15-24', '25-54', '54-64', '65+'] )

            pf= pandas.read_csv('CountryNameCode.csv', engine='python')

            for i in range(30):

                mtitle=pf[pf.columns[0]].iloc[i]


                url='https://www.indexmundi.com/'+str(mtitle)+'/age_structure.html'

                print(url)

                Age0_14=re.compile(r'.0-14 years: </strong>(.*)<br/><strong>15.*')
                Age15_24=re.compile(r'.15-24 years: </strong>(.*)<br/><strong>25.*')
                Age25_54=re.compile(r'.25-54 years: </strong>(.*)<br/><strong>55.*')
                Age54_64=re.compile(r'.55-64 years: </strong>(.*)<br/><strong>65.*')
                Age65=re.compile(r'.65 years and over: </strong>(.*)<br/>.*')

                driver=webdriver.Chrome("C:/Users/ATIV/Downloads/chromedriver_win32/chromedriver")
                driver.get(url)
                soup = BeautifulSoup(driver.page_source, "html.parser")

                link = soup.find_all("div", {"class":"c"})
                print("------------------------------------")
```

```
from selenium import webdriver
from bs4 import BeautifulSoup
import urllib.request
import re
import time
import numpy as np
import pandas
import csv
```

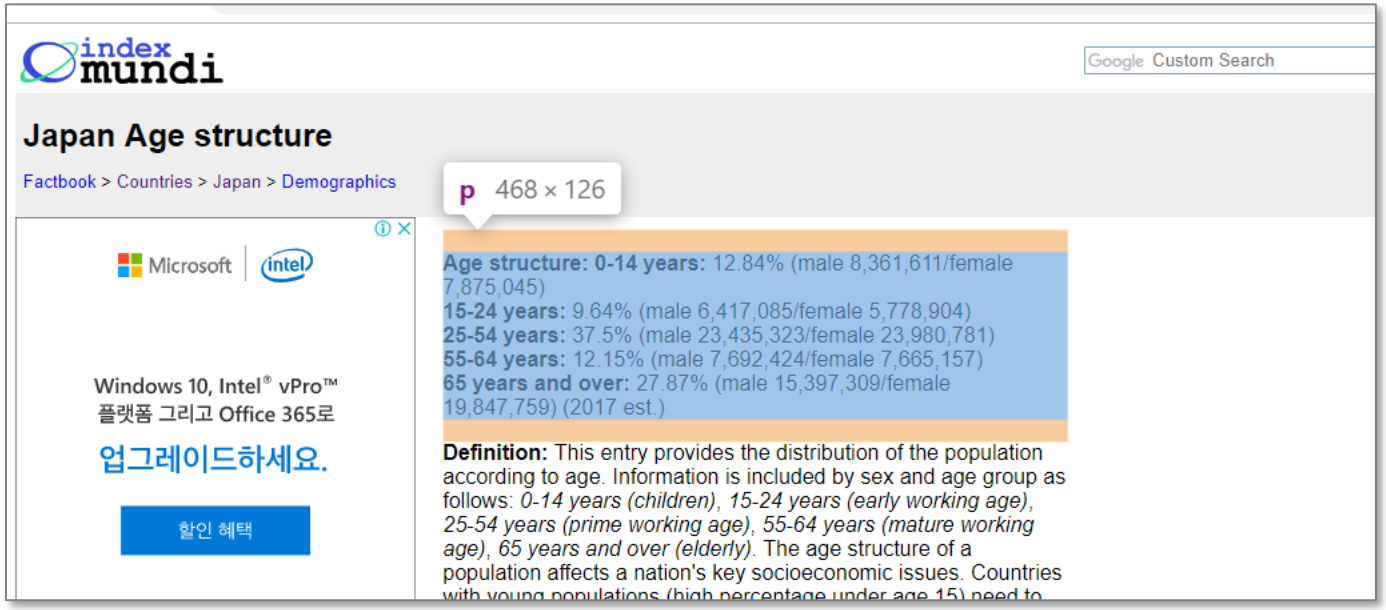
◀정규식으로 데이터 가져오기

◀find_all로 div class c 가져오기

Project Code Review

Project code and result

Data Crawling _ 정규식 표현 유의점




```
▼<div id="mainpanel" class="c">
  ▼<p> == $0
    <b>Age structure: </b>
    <strong>0-14 years: </strong>
    "12.84% (male 8,361,611/female 7,875,045)"
    <br>
    <strong>15-24 years: </strong>
    "9.64% (male 6,417,085/female 5,778,904)"
    <br>
    <strong>25-54 years: </strong>
    "37.5% (male 23,435,323/female 23,980,781)"
    <br>
    <strong>55-64 years: </strong>
    "12.15% (male 7,692,424/female 7,665,157)"
    <br>
    <strong>65 years and over: </strong>
    "27.87% (male 15,397,309/female 19,847,759) (2017 est.)"
```

```
https://www.indexmundi.com/japan/age_structure.html
[<div class="c" id="mainpanel">
<p><b>Age structure: </b><strong>0-14 years: </strong>12.84% (male 8,361,611/female 7,875,045)<br/><strong>15-24 years: </strong>9.64%
 (male 6,417,085/female 5,778,904)<br/><strong>25-54 years: </strong>37.5% (male 23,435,323/female 23,980,781)<br/><strong>55-64 years:
 </strong>12.15% (male 7,692,424/female 7,665,157)<br/><strong>65 years and over: </strong>27.87% (male 15,397,309/female 19,847,759)
 (2017 est.)<br/></p>
<p><b>Definition:</b> This entry provides the distribution of the population according to age. Information is included by sex and age g
roup as follows: <em>0-14 years (children)</em>, <em>15-24 years (early working age)</em>, <em>25-54 years (prime working age)</em>, <e
m>55-64 years (mature working age)</em>, <em>65 years and over (elderly)</em>. The age structure of a population affects a nation's key
 socioeconomic issues. Countries with young populations (high percentage under age 15) need to invest more in the health sector. Countries
 with older populations (high percentage ages 65 and over) need to invest more in the health sector. The age structure can also be used
```

◀개발자도구로 확인하는 표현과 실제 print된 값이 다른 경우가 존재
직접 확인해서 가져와야 함

Project Code Review

Project code and result

Data Crawling _ 정규식 표현

```
print(url)
total = re.compile(r'.<strong>total: </strong>(.*)<br/><strong>male.*')
male=re.compile(r'.<strong>male: </strong>(.*)<br/><strong>female.*')
female=re.compile(r'.<strong>female: </strong>(.*)<br/>.*')

driver=webdriver.Chrome("C:/Users/ATIV/Downloads/chromedriver_win32/chromedriver")
driver.get(url)
soup = BeautifulSoup(driver.page_source, "html.parser")


link = soup.find_all("div", {"class":"col-md-8 c"})
print("==================================")
```

```
print(url)
HS_consumption = re.compile(r'.household consumption: </strong>(.*)<br/><strong>government.*')
Gov_consumption=re.compile(r'.government consumption: </strong>(.*)<br/><strong>investment in fixed capital.*')
inv_fixedcapital=re.compile(r'.investment in fixed capital: </strong>(.*)<br/><strong>investment in inventories.*')
inv_inventories=re.compile(r'.inventories: </strong>(.*)<br/><strong>exports of goods and services.*')
exports = re.compile(r'.exports of goods and services: </strong>(.*)<br/><strong>imports.*')
imports = re.compile(r'.imports of goods and services: </strong>(.*)<br/>.*')

driver=webdriver.Chrome("C:/Users/ATIV/Downloads/chromedriver_win32/chromedriver")
driver.get(url)
soup = BeautifulSoup(driver.page_source, "html.parser")


link = soup.find_all("div", {"class":"col-md-8 c"})
```

```
print(url)
unemp_2006=re.compile(r'.2006</td><td>(.*)</td><td>.*')
unemp_2007=re.compile(r'.2007</td><td>(.*)</td><td>.*')
unemp_2008=re.compile(r'.2008</td><td>(.*)</td><td>.*')
unemp_2009=re.compile(r'.2009</td><td>(.*)</td><td>.*')
unemp_2010=re.compile(r'.2010</td><td>(.*)</td><td>.*')
unemp_2011=re.compile(r'.2011</td><td>(.*)</td><td>.*')
unemp_2012=re.compile(r'.2012</td><td>(.*)</td><td>.*')
unemp_2013=re.compile(r'.2013</td><td>(.*)</td><td>.*')
unemp_2014=re.compile(r'.2014</td><td>(.*)</td><td>.*')

driver=webdriver.Chrome("C:/Users/ATIV/Downloads/chromedriver_win32/chromedriver")
driver.get(url)
soup = BeautifulSoup(driver.page_source, "html.parser")


link = soup.find_all("table", {"class":"tblrank"})
```

```
print(url)
total = re.compile(r'.total: </strong>(.*)<br/><strong>land.*')
land=re.compile(r'.land: </strong>(.*)<br/><strong>water.*')
water=re.compile(r'.water: </strong>(.*)<br/>.*')

driver=webdriver.Chrome("C:/Users/ATIV/Downloads/chromedriver_win32/chromedriver")
driver.get(url)
soup = BeautifulSoup(driver.page_source, "html.parser")


link = soup.find_all("div", {"class":"col-md-8 c"})
```

```
agri=re.compile(r'.agricultural land: </strong>(.*)<br/>arable.*')
forest=re.compile(r'.forest: </strong>(.*)<br/><strong>other.*')
other=re.compile(r'.other: </strong>(.*)<br/></td></tr>.*')

driver=webdriver.Chrome("C:/Users/ATIV/Downloads/chromedriver_win32/chromedriver")
driver.get(url)
soup = BeautifulSoup(driver.page_source, "html.parser")

link = soup.find_all("div", {"class":"col-md-8 c"})
print("==================================")
```

Project Code Review

Project code and result

Data Crawling

```
for posts in link:
    time.sleep(0.33)
    z=str(posts)
    zz=str(posts).split(' ')
    str_index=0
    count = 0
    for aaa in zz:
        #index = 0
        str_index+=len(aaa)
        if word in aaa:
            count+=1
            if(count == num):

                index=aaa.find(word[len(word)-1])
                print(z[str_index-len(aaa)+len(word):])
                print(z[str_index:])
                break
        str_index+=1

    a=Age0_14.search(z).group(1).strip()
    b=Age15_24.search(z).group(1).strip()
    c=Age25_54.search(z).group(1).strip()
    d=Age54_64.search(z).group(1).strip()
    e=Age65.search(z).group(1).strip()
Field1=mtitle
Field3=a.split()[0][:len(a.split()[0])-1]
Field4=b.split()[0][:len(b.split()[0])-1]
Field5=c.split()[0][:len(c.split()[0])-1]
Field6=d.split()[0][:len(d.split()[0])-1]
Field7=e.split()[0][:len(e.split()[0])-1]

writer.writerow([Field1,Field3,Field4,Field5, Field6, Field7])
```

age_structure.csv

	A	B	C	D	E	F
1	country name	0-14	15-24	25-54	54-64	65+
2	japan	12.84	9.64	37.5	12.15	27.87
3	india	27.34	17.9	41.08	7.45	6.24
4	china	17.15	12.78	48.51	10.75	10.81
5	thailand	16.93	14.17	46.32	12	10.58
6	vietnam	23.55	16.23	45.56	8.55	6.12
7	cambodia	31.01	18.36	40.68	5.69	4.25
8	singapore	12.82	16.56	50.53	10.46	9.63
9	indonesia	25.02	16.99	42.4	8.58	7.01
10	canada	15.44	11.85	39.99	14.1	18.63
11	united_states	18.73	13.27	39.45	12.91	15.63
12	united_kingdom	17.53	11.9	40.55	11.98	18.04
13	germany	12.82	10.09	40.45	14.58	22.06
14	france	18.53	11.79	37.78	12.42	19.48
15	italy	13.65	9.66	42.16	12.99	21.53
16	spain	15.38	9.58	44.91	12.14	17.98
17	philippines	33.39	19.16	36.99	5.97	4.49
18	south_korea	13.21	12.66	45.52	14.49	14.12
19	mexico	26.93	17.54	40.81	7.64	7.09
20	austria	14.01	11.07	42.42	13.23	19.26
21	poland	14.76	10.7	43.48	14.21	16.86
22	portugal	15.34	11.36	41.72	12.18	19.4
23	egypt	33.29	18.94	37.6	5.95	4.22
24	denmark	16.41	13.08	38.76	12.52	19.23
25	australia	17.8	12.79	41.45	11.83	16.14
26	cambodia	31.01	18.36	40.68	5.69	4.25
27	belgium	17.16	11.34	40.05	12.86	18.58
28	sudan	38.68	21.04	32.77	4.24	3.27
29	russia	17.12	9.46	44.71	14.44	14.28
30	norway	18	12.58	41.01	11.71	16.71
31	saudi_arabia	26.1	18.57	46.86	5.03	3.44

Project Code Review

Project code and result

Data Crawling exception, checktime

```python
@CheckTime
def get_population():
    population_crawling.age_structure()
    population_crawling.median_age()

@CheckTime
def get_Unemp():
    try:
        economy_crawling.Unemployment()
    except AttributeError:
        pass

    except IndexError:
        raise WrongDataRecord("CountryNameCode.csv 파일의 나라 개수를 다시 확인해주세요")

    except:
        pass


@CheckTime
def get_GDP():
    try:
        economy_crawling.GDP_composition()
    except:
        raise WrongDataRecord("에러가 발생했습니다. 크롤링을 다시 시작해주세요")

@CheckTime
def get_GEO():
    try:
        geography_crawling.area_comp()
        geography_crawling.land_use()
    except:
        raise WrongDataRecord("에러가 발생했습니다. 크롤링을 다시 시작해주세요")
```

```
get_GEO()
https://www.indexmundi.com/egypt/geography_profile.html
====================================
https://www.indexmundi.com/denmark/geography_profile.html
====================================
https://www.indexmundi.com/australia/geography_profile.html
====================================
https://www.indexmundi.com/cambodia/geography_profile.html
====================================
https://www.indexmundi.com/belgium/geography_profile.html
====================================
https://www.indexmundi.com/sudan/geography_profile.html
====================================
https://www.indexmundi.com/russia/geography_profile.html
====================================
https://www.indexmundi.com/norway/geography_profile.html
====================================
https://www.indexmundi.com/saudi_arabia/geography_profile.html
====================================

실행시간 :  1101.619479894638
```

▲실행시간: 각 크롤링에 걸리는 총 시간

◀AttributeError, IndexError에 대한 처리 ErrorMsg

Project Code Review

Project code and result

Data Frame and Visualization

land_use.csv

	country name	agriculture_land	forest	other	total
0	japan	12.5	68.5	19	100
1	india	60.5	23.1	16.4	100
2	china	54.7	22.3	23	100
3	thailand	41.2	37.2	21.6	100
4	vietnam	34.8	45.0	20.2	100
5	cambodia	32.1	56.5	11.4	100
6	singapore	1.0	3.3	95.7	100
7	indonesia	31.2	51.7	17.1	100
8	canada	6.8	34.1	59.1	100
9	united_states	44.5	33.3	22.2	100
10	united_kingdom	71.0	11.9	17.1	100
11	germany	48.0	31.8	20.2	100
12	france	52.7	29.2	18.1	100
13	italy	47.1	31.4	21.5	100

Module	Plotly
Chart Type	100% horizontal bar chart

area_comp.csv

	country name	total	land	water
0	japan	377,915	364,485	13,430
1	india	3,287,263	2,973,193	314,070
2	china	9,596,960	9,326,410	270,550
3	thailand	513,120	510,890	2,230
4	vietnam	331,210	310,070	21,140
5	cambodia	181,035	176,515	4,520
6	singapore	719.2	709.2	10
7	indonesia	1,904,569	1,811,569	93,000
8	canada	9,984,670	9,093,507	891,163
9	united_states	9,833,517	9,147,593	685,924
10	united_kingdom	243,610	241,930	1,680
11	germany	357,022	348,672	8,350
12	france	643,801	640,427	3,374
13	italy	301,340	294,140	7,200

Module	Plotly
Chart Type	Combined line chart

age_structure.csv

	country name	0-14	15-24	25-54	54-64	65+
0	japan	12.84	9.64	37.50	12.15	27.87
1	india	27.34	17.90	41.08	7.45	6.24
2	china	17.15	12.78	48.51	10.75	10.81
3	thailand	16.93	14.17	46.32	12.00	10.58
4	vietnam	23.55	16.23	45.56	8.55	6.12
5	cambodia	31.01	18.36	40.68	5.69	4.25
6	singapore	12.82	16.56	50.53	10.46	9.63
7	indonesia	25.02	16.99	42.40	8.58	7.01
8	canada	15.44	11.85	39.99	14.10	18.63
9	united_states	18.73	13.27	39.45	12.91	15.63
10	united_kingdom	17.53	11.90	40.55	11.98	18.04
11	germany	12.82	10.09	40.45	14.58	22.06
12	france	18.53	11.79	37.78	12.42	19.48

Module	Plotly, seaborn
Chart Type	World map, bar chart

Project Code Review

Project code and result

median_age.csv

	country name	total	male	female
0	japan	47.3	46.0	48.7
1	india	27.9	27.2	28.6
2	china	37.4	36.5	38.4
3	thailand	37.7	36.6	38.7
4	vietnam	30.5	29.4	31.7
5	cambodia	25.3	24.6	26.0
6	singapore	34.6	34.5	34.7
7	[illegible]	[illegible]	[illegible]	30.8
				43.5

GDP_composition.csv

	country name	household consumption	government consumption	investment in fixed capital	investment in inventories	exports of S&G	imports of S&G
0	japan	55.9	19.5	23.5	0.2	17.8	-16.8
1	india	58.7	11.6	27.5	4.0	18.4	-20.2
2	china	39.1	14.6	43.3	1.1	19.6	-17.7
3	thailand	50.1	17.0	24.2	-7.0	70.4	-54.7
4	vietnam	68.5	6.6	24.8	2.9	98.6	-101.4
5	cambodia	76.4	5.4	22.0	0.9	62.8	-67.4
6	singapore	34.7	11.4	23.5	1.9	179.2	-150.6
7	indonesia	57.5	8.9	32.1	0.7	19.2	-18.4
8	canada	58.1	20.9	22.8	0.3	31.4	-33.6
9	united_states	69.1	17.2	16.3	0.3	12.2	-15.1
		65.3	19.0	16.6	0.7	30.1	-31.7
		53.7	19.9	20.1	-1.0	47.3	-40.0
		54.8	23.5	22.0	1.3	30.3	-32.0
		60.2	18.7	17.2	0.1	31.8	-28.0

Unemployment.csv

	country name	2006	2007	2008	2009	2010	2011	2012	2013	2014
0	spain	8.4	8.2	11.0	17.0	19.0	21.0	24.0	26.	24.0
1	philippines	7.9	7.3	7.4	7.4	7.3	7.0	6.9	7.0	6.8
2	south_korea	3.4	3.2	3.1	3.6	3.7	3.4	3.2	3.1	3.5
3	mexico	3.5	3.6	3.8	5.3	5.2	5.1	4.8	4.9	4.7
4	austria	4.7	4.4	3.8	4.7	4.4	4.1	4.3	4.9	5.0
5	poland	13.0	9.6	7.1	8.1	9.6	9.6	10.0	10.	8.9
6	portugal	7.6	7.9	7.5	9.4	10.0	12.0	15.0	16.	13.0
7	egypt	10.0	9.2	8.6	9.3	9.2	10.0			
8	denmark	3.9	3.7	3.4	5.9	7.4	7.5			
9	australia	4.7	4.3	4.2	5.5	5.2	5.0			

Module	Plotly
Chart Type	Line graph

Module	Plotly
Chart Type	Pie chart

Visualization

Dataframe visualization

Demographic Data seaborn

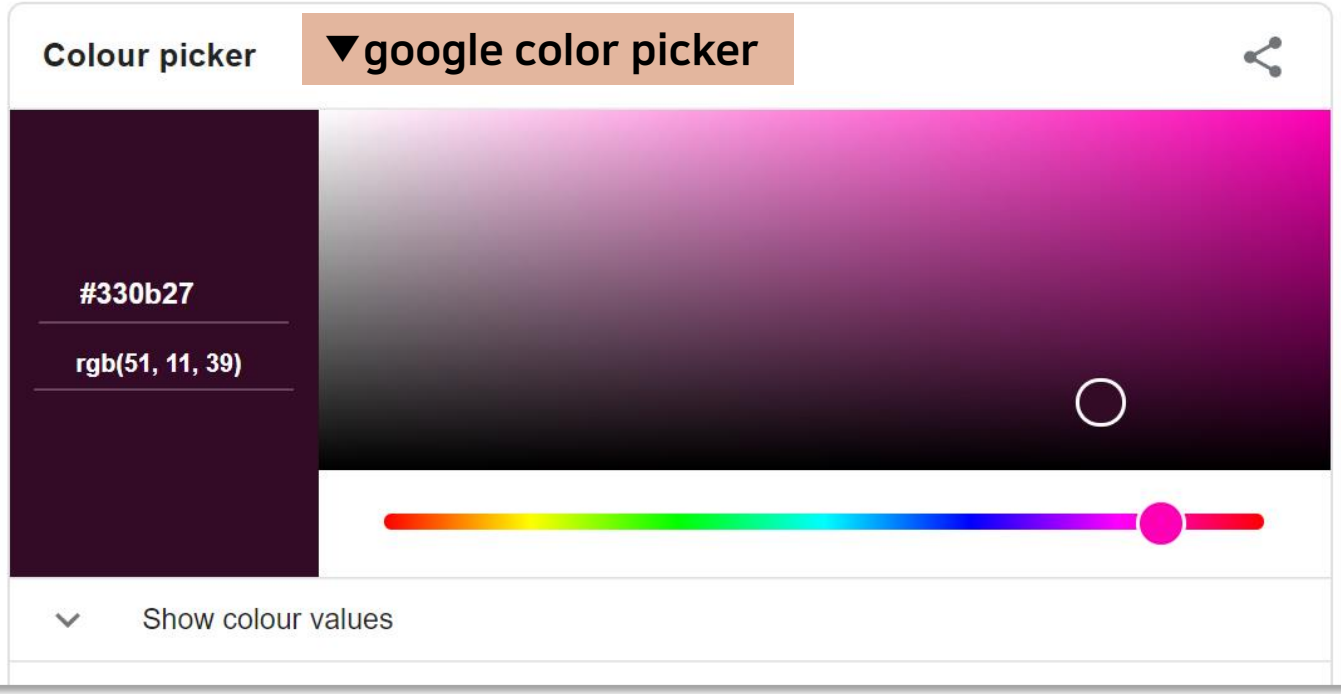
```
import matplotlib.pyplot as plt
# 주피터에 포함시키기
%matplotlib inline
import seaborn as
```

▼각 국가마다 팔레트 안에 색 지정

```
pkmn_type_colors = ['#78C850',  # japan
                    '#F08030',  # india
                    '#6890F0',  # china
                    '#A8B820',  # thailand
                    '#A8A878',  # vietnam
                    '#A040A0',  # cambodia
                    '#F8D030',  # singapore
                    '#E0C068',  # indonesia
                    '#EE99AC',  # canada
                    '#C03028',  # united_states
                    '#F85888',  # united_kingdom
                    '#B8A038',  # germany
                    '#705898',  # france
                    '#98D8D8',  # spain
                    '#7038F8',  # philippines
                    '#5BC7E5',  # south_korea
                    '#1c9ca5',  # mexico
                    '#6840A0',  # poland
                    '#52206a',  # portugal
                    '#6A1748',  # egypt
                    '#5BC7E5',  # denmark
                    '#09CAFF',  # australia
                    '#5BC7E5',  # cambodia
                    '#e2ceae',  # belgium
                    '#5BC7E5',  # sudan
                    '#E55BAD',  # russia
                    '#dc0e88',  # norway
                    '#E5D05B',  # saudi_arabia
```

```
sns.set(style="ticks")
plt.xticks(rotation=-40)
age_structure1= age_structure.sort_values(by='65+', ascending = False)
age_graph1= sns.barplot(x='country name',y='65+', data=age_structure1, palette=pkmn_type_colors)
age_graph1.figure.set_size_inches(20,9)
age_graph1.axes.set_title('Aged Population by Country', fontsize=34,color="b",alpha=0.5)
age_graph1.set_xlabel("Country Name",size = 20,color="r",alpha=0.5)
age_graph1.set_ylabel("Population % over 65",size = 20,color="g",alpha=0.5)
age_graph1.tick_params(labelsize=14,labelcolor="black")
```

▼google color picker



Visualization

Dataframe visualization

Demographic Data seaborn

▼subplot으로 각 age group 정보 보여주기

```
import seaborn as sns
from matplotlib.gridspec import GridSpec

sns.set(style="whitegrid")
#sns.set_color_codes("Spectral")

plt.figure(2, figsize=(33,24))
the_grid = GridSpec(2, 2)

plt.subplot(the_grid[0, 1],  title='Population % over 65+')
sns.barplot(x='country name',y='65+', data=age_structure1, palette=pkmn_type_colors)
plt.xticks(rotation=-50)
plt.subplot(the_grid[0, 0], title='%Age 25-54')
sns.barplot(x='country name',y='25-54', data=age_structure1, palette=pkmn_type_colors)
plt.xticks(rotation=-50)
plt.subplot(the_grid[1, 1],  title='%Age 15-24')
sns.barplot(x='country name',y='15-24', data=age_structure1, palette=pkmn_type_colors)
plt.xticks(rotation=-50)
plt.subplot(the_grid[1, 0], title='%Age 54-64')
sns.barplot(x='country name',y='54-64', data=age_structure1, palette=pkmn_type_colors)
plt.xticks(rotation=-50)

plt.suptitle('Age Structure', fontsize=30,color="b",alpha=0.5)
age_graph1.tick_params(labelsize=9,labelcolor="blue")
```

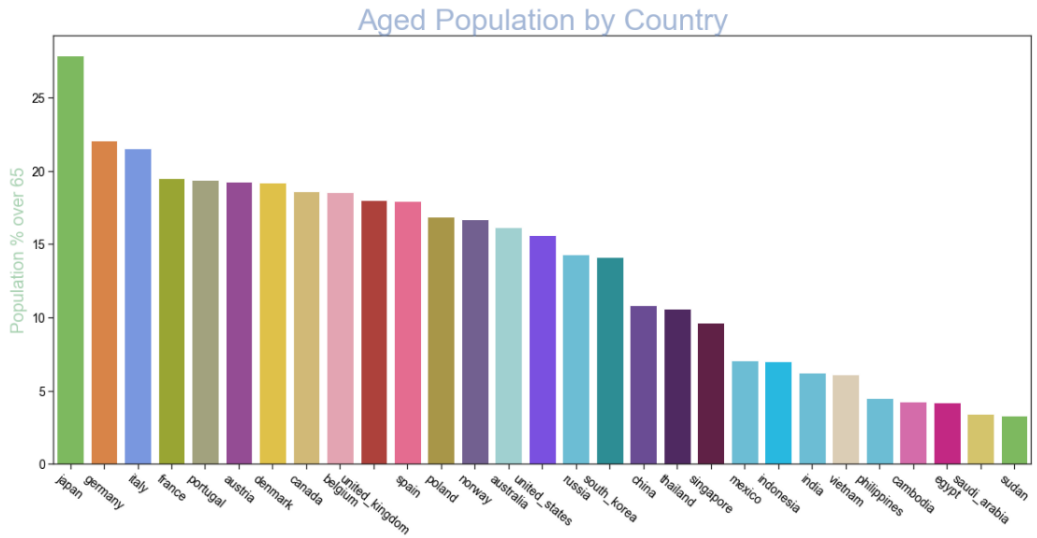
▼median age 정보 보여주기

```
import seaborn as sns
from matplotlib.gridspec import GridSpec

sns.set(style="whitegrid")
#sns.set_color_codes("Spectral")

plt.figure(3, figsize=(20,15))
the_grid = GridSpec(3, 3)
median_age1= median_age.sort_values(by='total', ascending = False)

plt.subplot(the_grid[0, 1],  title='Total Median Age')
sns.barplot(x='country name',y='total', data=median_age1, palette=pkmn_type_colors)
plt.xticks(rotation=-40)
plt.subplot(the_grid[0, 0], title='Femal Median Age')
sns.barplot(x='country name',y='female', data=median_age1, palette=pkmn_type_colors)
plt.xticks(rotation=-40)
plt.subplot(the_grid[0, 2],  title='Male Median Age')
sns.barplot(x='country name',y='male', data=median_age1, palette=pkmn_type_colors)
plt.xticks(rotation=-40)


plt.suptitle('Median Age', fontsize=30,color="b",alpha=0.5)
```

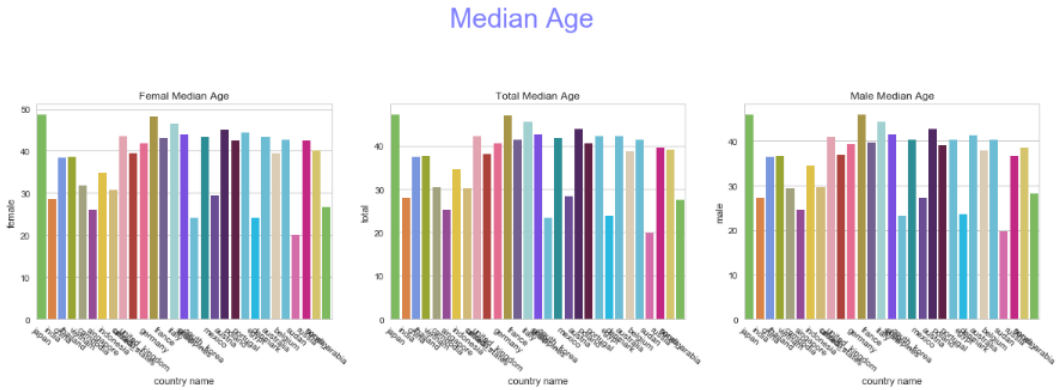
Visualization

Dataframe visualization

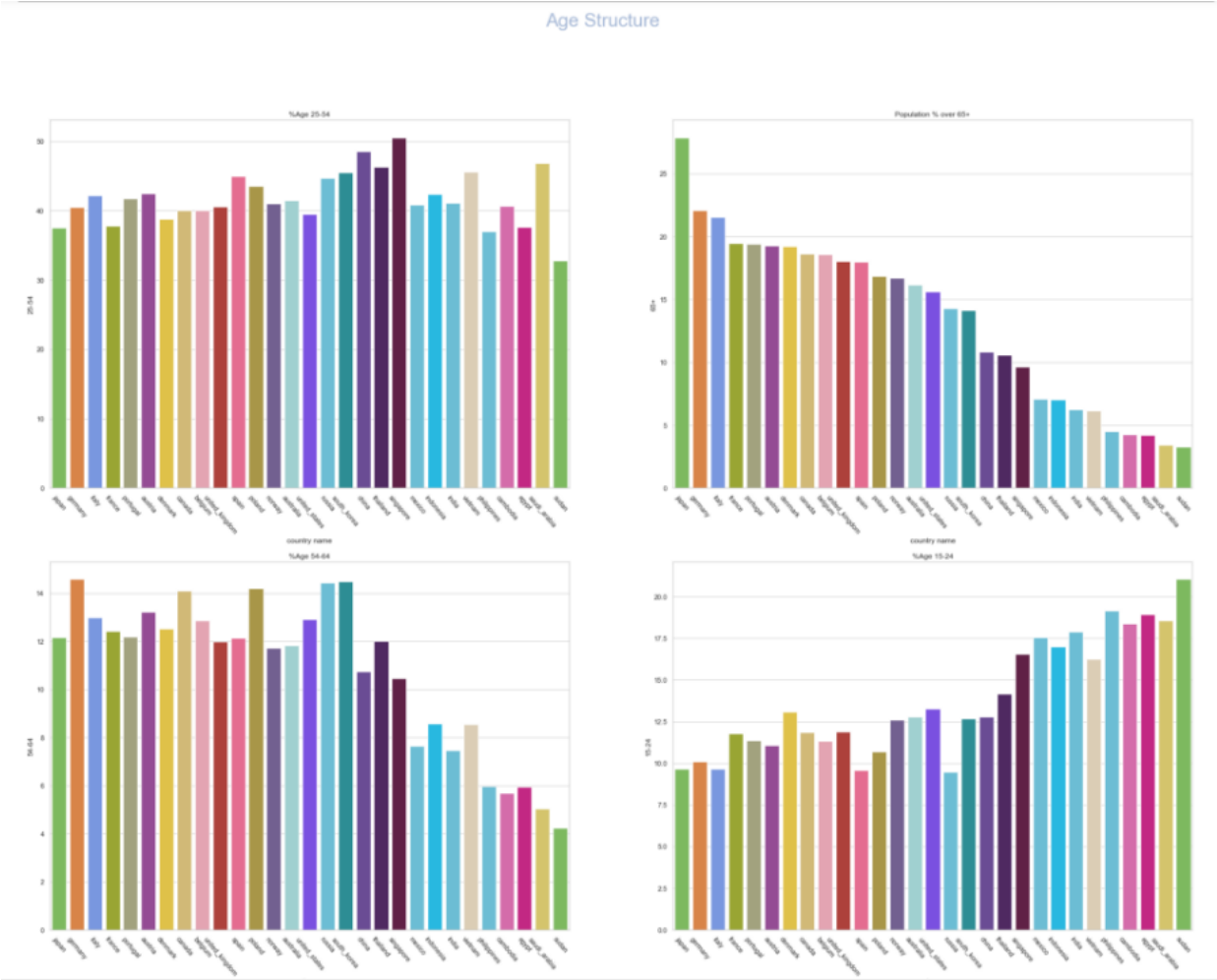
Demographic Data seaborn



▲65+population percentage sorted



각 age group 정보 subplot▼



Economic Data Unemployment Rate (plotly)

```
import plotly
plotly.tools.set_credentials_file(username='BOSUNG', api_key='pdpcFjaV9BMmS3s8Y5fz')
```

◀plotly 계정 연결

```
import csv
import pandas as pd

import plotly.plotly as py
import plotly.graph_objs as go

unemployment = pd.read_csv('unemployment.csv')
unemployment1 = unemployment.T
unemployment1.iloc[ : , 3]
year = ['2006','2007','2008','2009','2010','2011','2012','2013','2014']
```

```
unemployment1 = unemployment.T
unemployment1.iloc[ : , 3]
country= unemployment1.iloc[ 0 , : ]
unemployment2 = unemployment1.iloc[1:,:]
unemployment2
spain = unemployment2.iloc[ : , 0]
philippines= unemployment2.iloc[ : , 1]
south_korea=unemployment2.iloc[ : , 2]
mexico=unemployment2.iloc[ : , 3]
austria=unemployment2.iloc[ : , 4]
poland=unemployment2.iloc[ : , 5]
portugal=unemployment2.iloc[ : , 6]
egypt=unemployment2.iloc[ : , 7]
denmark=unemployment2.iloc[ : , 8]
australia=unemployment2.iloc[ : , 9]
unemployment
```

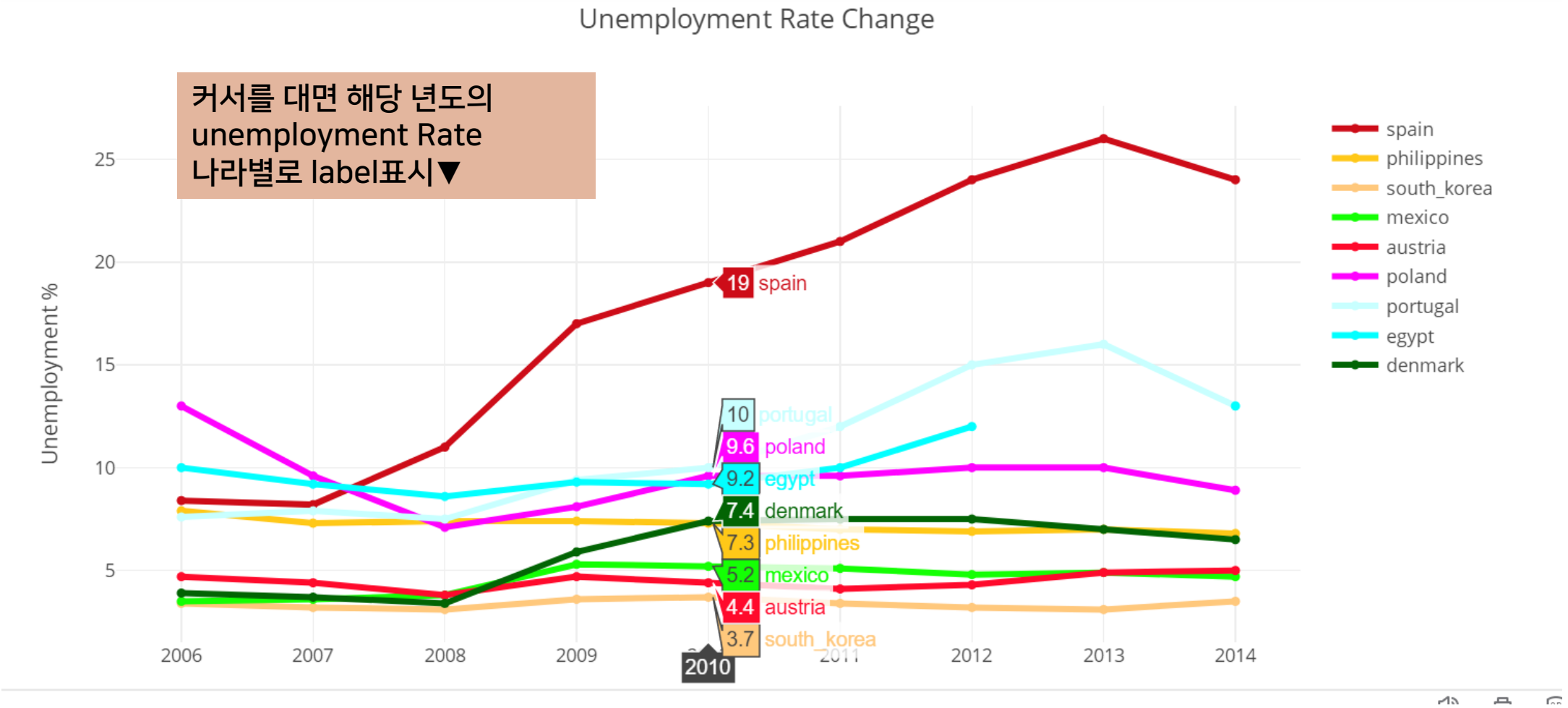
◀데이터 처리

```
trace6 = go.Scatter(
    x = year,
    y = portugal,
    name = 'portugal',
    line = dict(
        color = ('rgb(200, 800, 400)'),
        width = 4)
)

trace7 = go.Scatter(
    x = year,
    y = egypt,
    name = 'egypt',
    line = dict(
        color = ('rgb(2, 600, 400)'),
        width = 4)
)

trace8 = go.Scatter(
    x = year,
    y = denmark,
    name = 'denmark',
    line = dict(
        color = ('rgb(2, 100, 4)'),
        width = 4)
)

data = [trace0, trace1,trace2, trace3, trace4, trace5, trace6, trace7, trace8]
layout = dict(title = 'Unemployment Rate Change',
              xaxis = dict(title = 'Year'),
              yaxis = dict(title = 'Unemployment %'),

              )

fig = dict(data=data, layout=layout)
py.iplot(fig, filename='legend-names')
```

Visualization

Dataframe visualization

Economic Data Unemployment Rate (plotly)



Economic Data GDP composition (plotly)

```
import pandas as pd
GDP_comp = pd.read_csv('GDP_composition.csv')
GDP_comp
GDP_comp1 = GDP_comp.T
GDP_comp2= GDP_comp1.iloc[ 1: , ]

print("_____________________________________________________________________________________________________________")
print("Get GDP composition")
print("Category: 1.household consumption, 2.gov comsumption 3.inv in fc 4. inv in inventories 5. expoert of S&G  6.imports of S&G")

a = input("type the country name: ")
for i in range(0,30):
    if GDP_comp1.iloc[ : , i][0] == a:
        print()
        print (GDP_comp1.iloc[ : , i])
```

◀국가 이름을 input으로 받아서 해당 국가 데이터 pie chart로 나타냄(donut)

Market로 색 지정가능▶

```
fig = {
  "data": [
    {
      "values": GDP_comp2.iloc[ : , i],
      "labels": [
        "household consumption",
        "gov consumption",
        "investment in fixed capital",
        "investment in inventories",
        "exports of S&G",
        "imports of S&G",
      ]
      'marker': {'colors': ['rgb(244, 66, 241)',
                            'rgb(244, 65, 143)',
                            'rgb(226, 0, 98)',
                            'rgb(145, 21, 97)',
                            'rgb(186, 24, 75)',
                            'rgb(186, 24, 183)']},
      "domain": {"column": 0},
      "name": "GDP",
      "hoverinfo":"label+percent+name",
      "hole": .4,
      "type": "pie"
    }],
  "layout": {
        "title":"GDP Composition"+ " "+ str(a),
        "grid": {"rows": 1, "columns": 1},
        "annotations": [
            {
                "font": {
                    "size": 20
                },
                "showarrow": False,
                "text": "GDP",
                "x": 0.50,
                "y": 0.5
            }]
    }
}
py.iplot(fig, filename='donut')
```

Visualization

Dataframe visualization

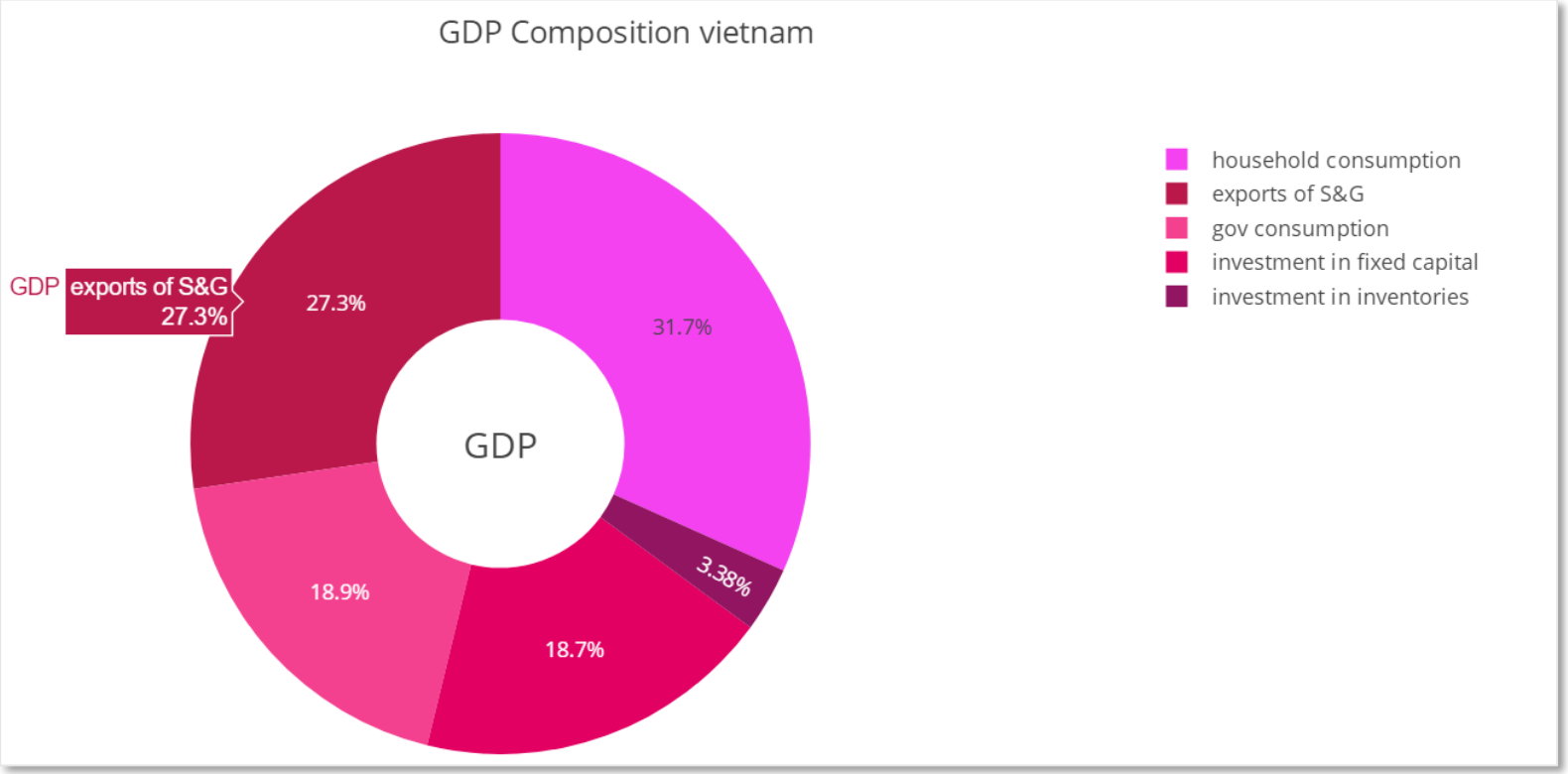
Economic Data GDP composition (plotly)

```
--------------------------------------------------------------------------------
Get GDP composition
Category: 1.household consumption, 2.gov comsumption 3.inv in fc 4. inv in inventories 5. expoert of S&G  6.imports of S&G
type the country name: vietnam
```

해당 국가이름 입력

```
country name                    vietnam
household consumption              68.5
government consumption              6.6
investment in fixed capital        24.8
investment in inventories           2.9
exports of S&G                     98.6
imports of S&G                   -101.4
Name: 4, dtype: object
```

▲해당 정보표시



마우스 위치에
따른 정보 표시▶

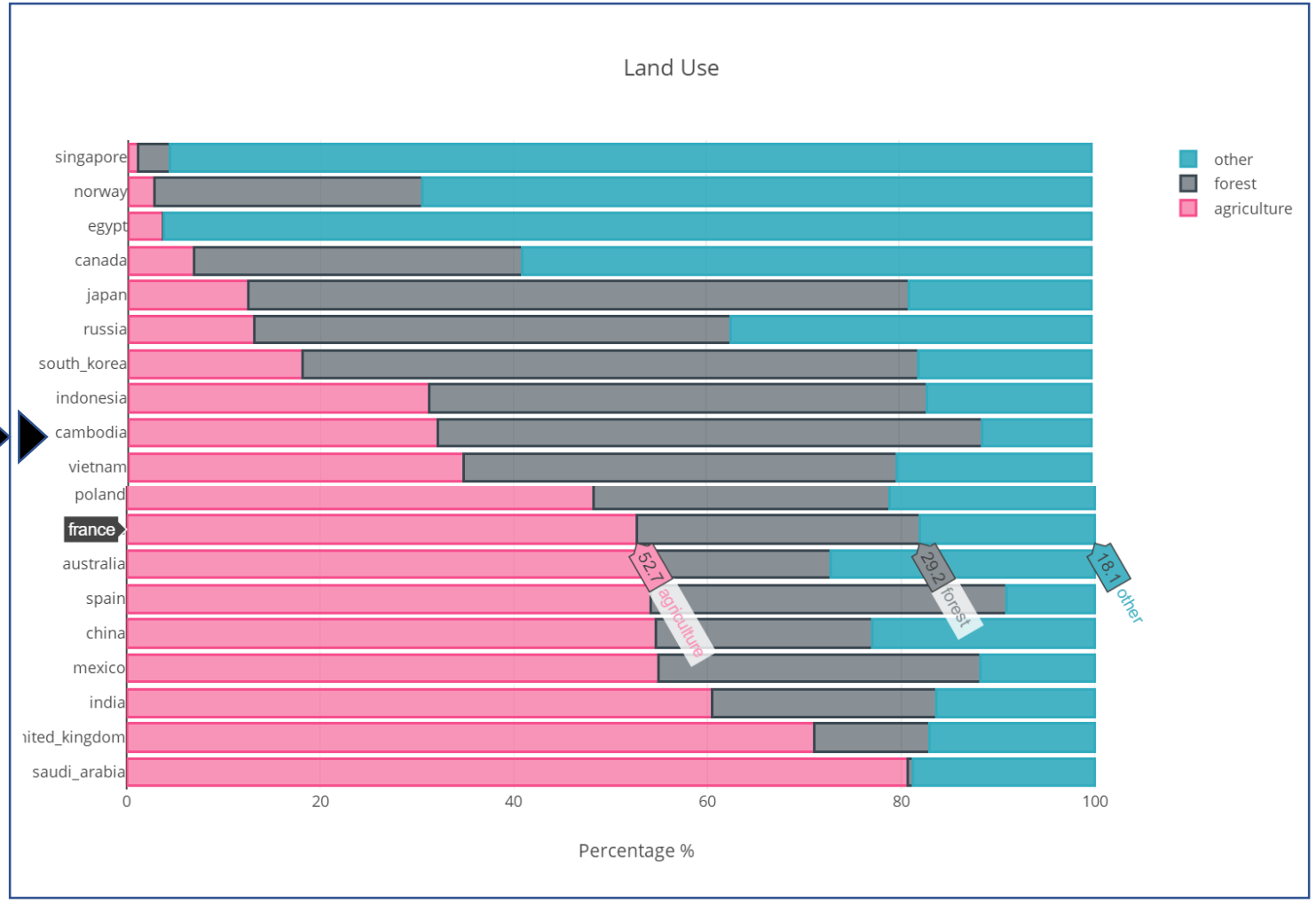
Visualization

Dataframe visualization

Geographic Data land_use (plotly)

	country name	agriculture_land	forest	other	total
29	saudi_arabia	80.7	0.5	18.8	100
10	united_kingdom	71.0	11.9	17.1	100
1	india	60.5	23.1	16.4	100
17	mexico	54.9	33.3	11.8	100
2	china	54.7	22.3	23	100
14	spain	54.1	36.8	9.1	100
23	australia	53.4	19.3	27.3	100
12	france	52.7	29.2	18.1	100
19	poland	48.2	30.6	21.2	100
11	germany	48.0	31.8	20.2	100
13	italy	47.1	31.4	21.5	100
9	united_states	44.5	33.3	22.2	100
25	belgium	44.1	22.4	33.5	100
3	thailand	41.2	37.2	21.6	100

▲agriculture_land 기준으로 sort 시킨 데이터 이용



Visualization
Dataframe visualization

Geographic Data area_comp(plotly)

```
area_comp1['land%'] = area_comp1['land'].div(area_comp1['total'])*100
area_comp1['water%'] = area_comp1['water'].div(area_comp1['total'])*100

area_comp2 = area_comp1.sort_values(by='land%', ascending = False)
area_comp2
```

▲column 두개 추가

	country name	total	land	water	land%	water%
29	saudi_arabia	2149690.0	2149690.0	0.0	100.000000	0.000000
3	thailand	513120.0	510890.0	2230.0	99.565404	0.434596
12	france	643801.0	640427.0	3374.0	99.475925	0.524075
21	egypt	1001450.0	995450.0	6000.0	99.400869	0.599131
15	philippines	300000.0	298170.0	1830.0	99.390000	0.610000
20	portugal	92090.0	91470.0	620.0	99.326746	0.673254
10	united_kingdom	243610.0	241930.0	1680.0	99.310373	0.689627
23	australia	7741220.0	7682300.0	58920.0	99.238880	0.761120
25	belgium	30528.0	30278.0	250.0	99.181080	0.818920
17	mexico	1964375.0	1943945.0	20430.0	98.959975	1.040025
14	spain	505370.0	498980.0	6390.0	98.735580	1.264420

```
y_agr= land_use2['agriculture_land']
y_land = area_comp2['land%']
x_agr = land_use2['country name']
x_land = area_comp2['country name']

trace0 = go.Bar(
    x=y_agr,
    y=x_agr,
    marker=dict(
        color='rgba(244, 66, 185, 0.6)',
        line=dict(
            color='rgba(244, 66, 185, 1.0)',
            width=1),
    ),
    name='argriculture land',
    orientation='h',
)
trace1 = go.Scatter(
    x=y_land,
    y=x_land,
    mode='lines+markers',
    line=dict(
```

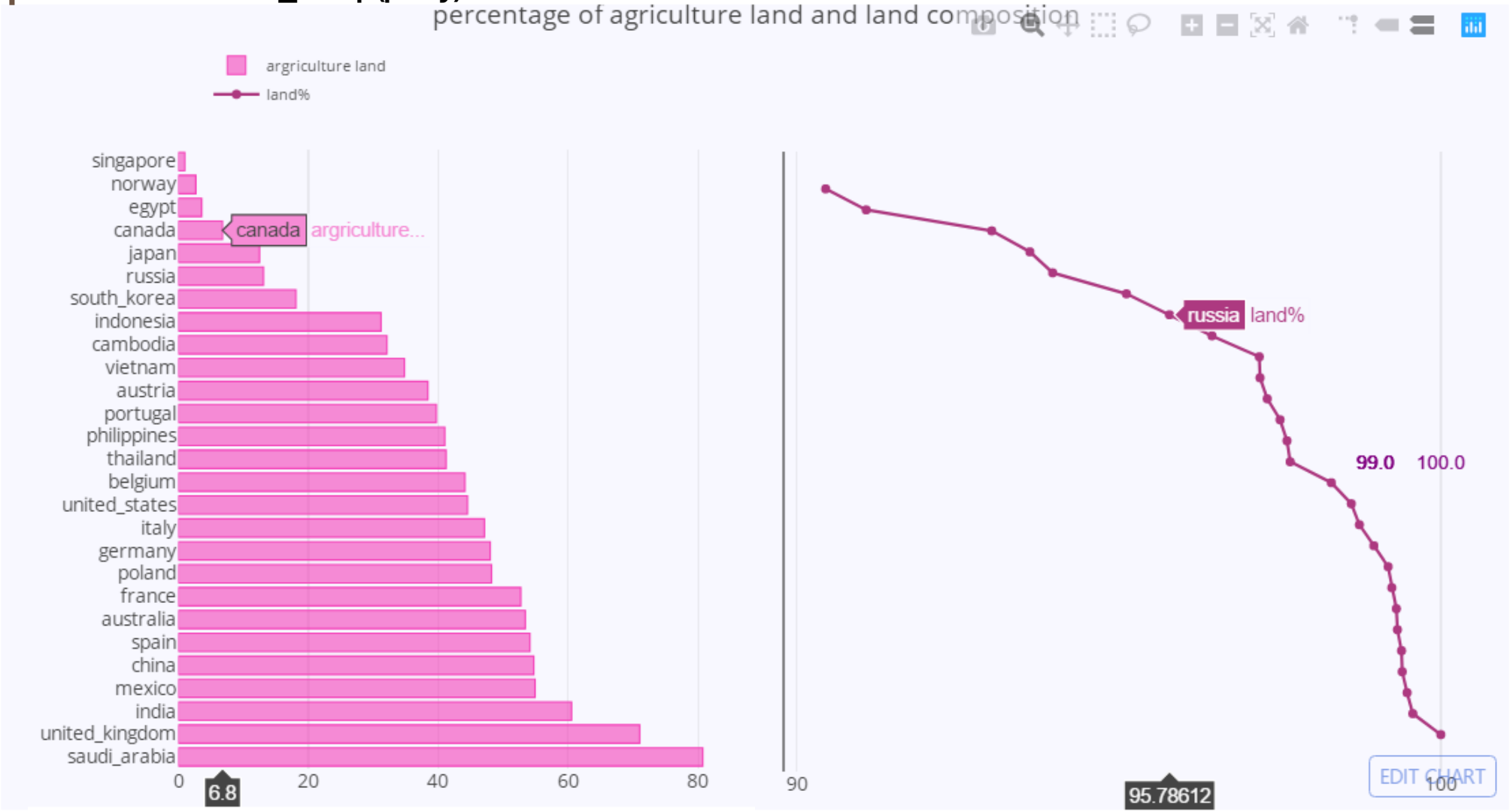
Subplot 두개로 다른 두 정보 한번에 볼 수 있게

```
# Creating two subplots
fig = tools.make_subplots(rows=1, cols=2, specs=[[{}, {}]], shared_xaxes=True,
                          shared_yaxes=False, vertical_spacing=0.001)

fig.append_trace(trace0, 1, 1)
fig.append_trace(trace1, 1, 2)

fig['layout'].update(layout)
py.iplot(fig, filename='area composition and land_use')
```

Visualization

Dataframe visualization

Geographic Data area_comp(plotly)



Visualization

Dataframe visualization

Demographic Data ploty, My_choromap

◀색깔 강도 설정

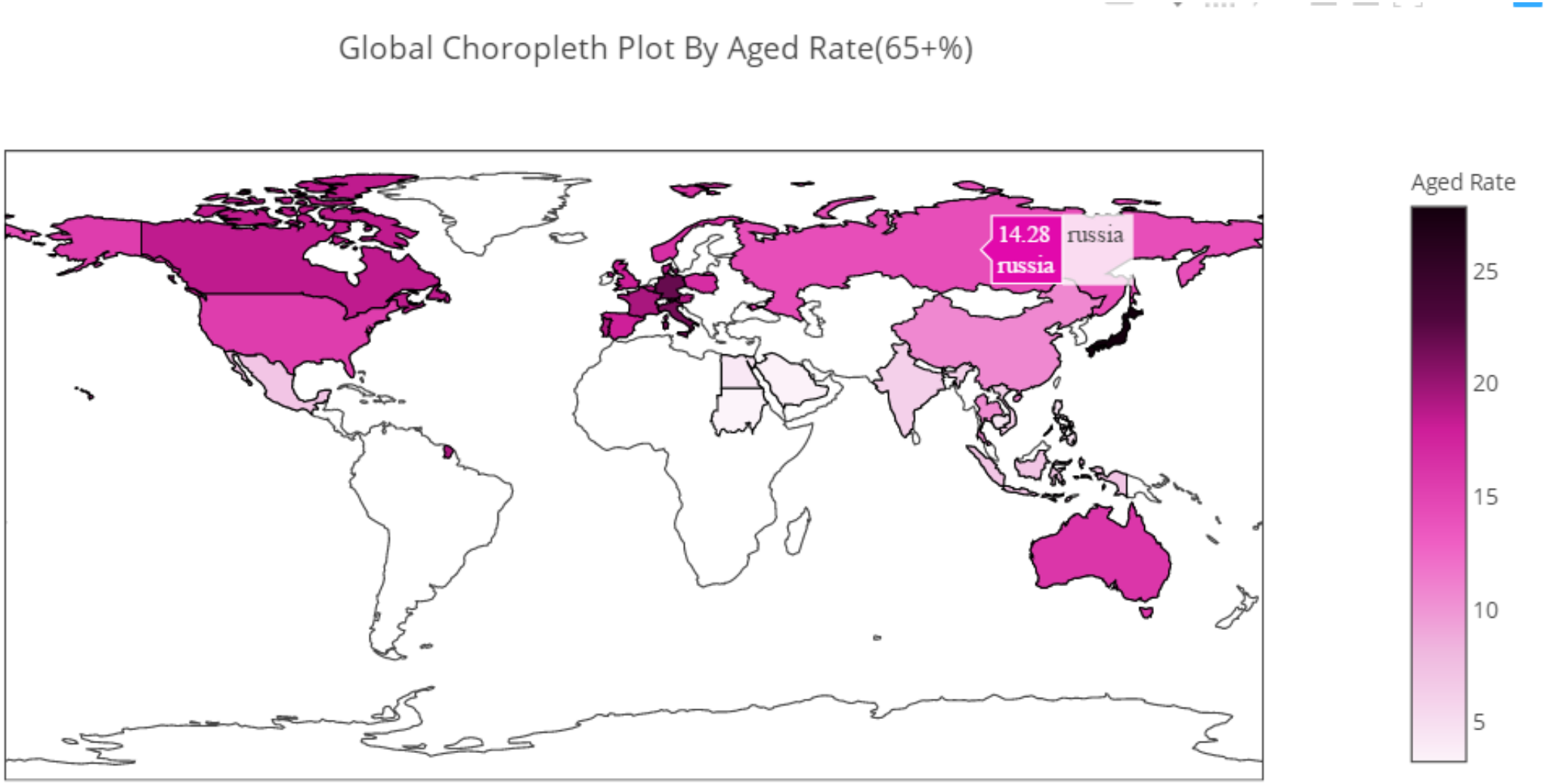
```
sc1 = [
    [0.0, 'rgb(20, 0, 14)'],
    [0.2, 'rgb(79, 7, 61)'],
    [0.4, 'rgb(206, 30, 153)'],
    [0.6, 'rgb(239, 93, 195)'],
    [0.8, 'rgb(239, 184, 223)'],
    [1.0, 'rgb(252, 244, 250)']
]
```

```
meta_data = dict(type = 'choropleth',
            locations = age_structure['country name'],
            colorscale = sc1,
            reversescale = True,
            locationmode = 'country names',
            z = age_structure['65+'],
            marker= dict(line=dict(color='black',
                         width=1)),
            text = age_structure['country name'],
            hoverlabel= dict(bgcolor= '#e209ac',
                             font=dict(family='Times New Roman',
                                       color='white')),
            colorbar = {'title':'Aged Rate', 'nticks':9}
            )

layout = dict(title = 'Global Choropleth Plot By Aged Rate(65+%)',
              geo = dict(showframe = True,
                         projection = {'type': 'equirectangular'}))

My_choromap = go.Figure(data = [meta_data], layout=layout)
py.iplot(My_choromap)
```

	country name	0-14	15-24	25-54	54-64	65+
0	japan	12.84	9.64	37.50	12.15	27.87
1	india	27.34	17.90	41.08	7.45	6.24
2	china	17.15	12.78	48.51	10.75	10.81
3	thailand	16.93	14.17	46.32	12.00	10.58
4	vietnam	23.55	16.23	45.56	8.55	6.12
5	cambodia	31.01	18.36	40.68	5.69	4.25
6	singapore	12.82	16.56	50.53	10.46	9.63
7	indonesia	25.02	16.99	42.40	8.58	7.01
8	canada	15.44	11.85	39.99	14.10	18.63
9	united_states	18.73	13.27	39.45	12.91	15.63
10	united_kingdom	17.53	11.90	40.55	11.98	18.04
11	germany	12.82	10.09	40.45	14.58	22.06
12	france	18.53	11.79	37.78	12.42	19.48
13	italy	13.65	9.66	42.16	12.99	21.53
14	spain	15.38	9.58	44.91	12.14	17.98
15	philippines	33.39	19.16	36.99	5.97	4.49
16	south_korea	13.21	12.66	45.52	14.49	14.12
17	mexico	26.93	17.54	40.81	7.64	7.09

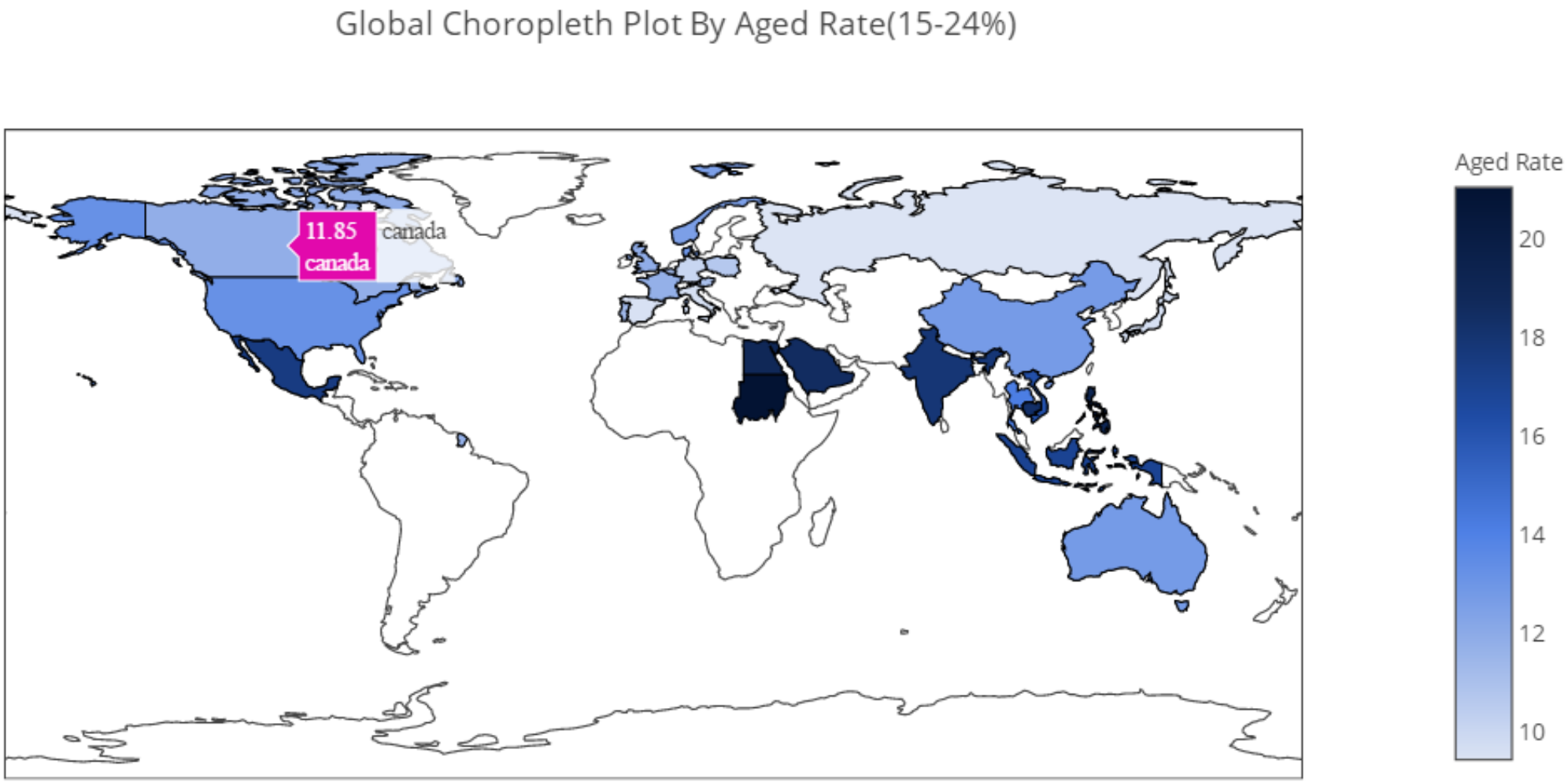
Visualization

Dataframe visualization



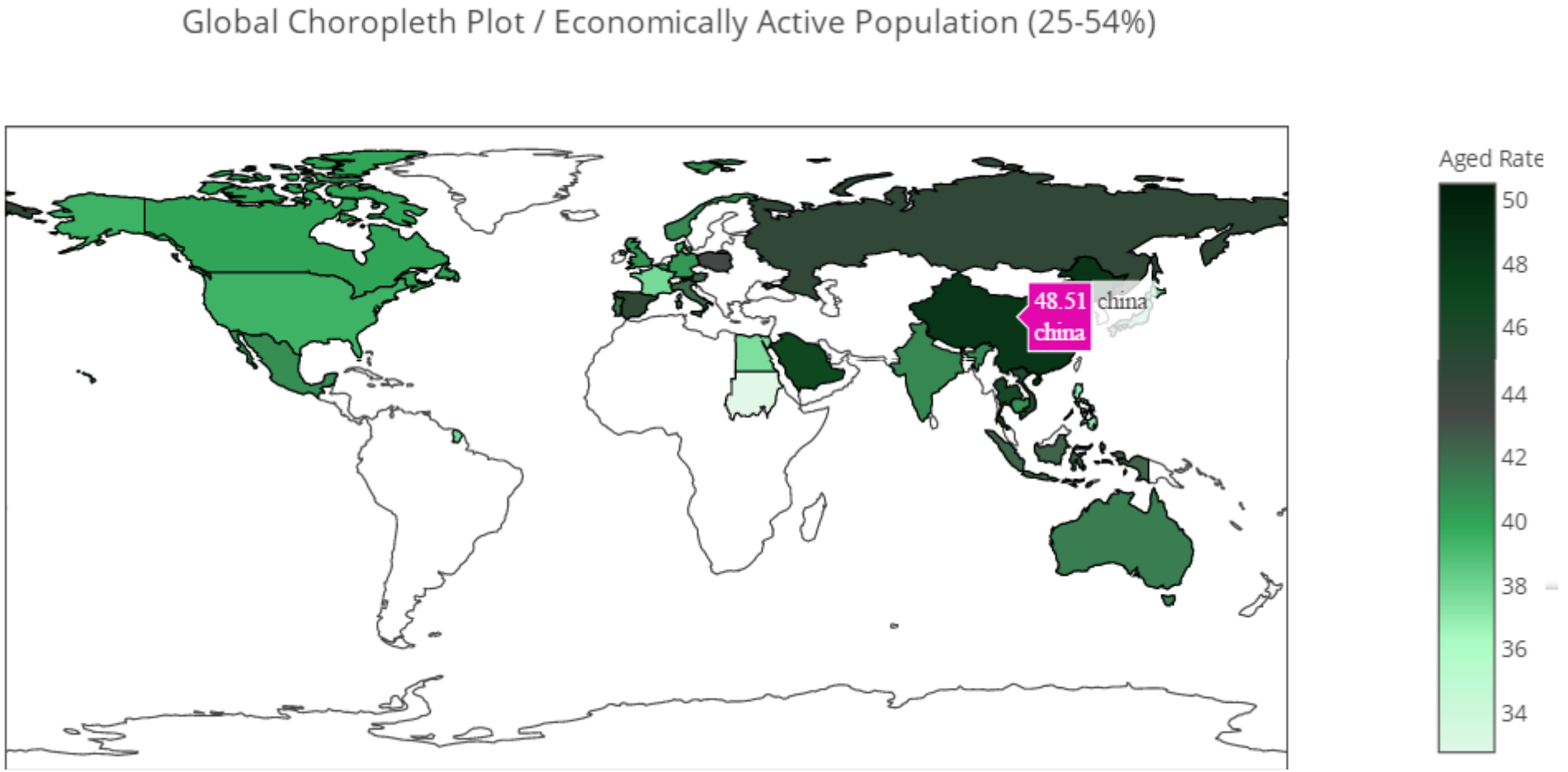
Visualization

Dataframe visualization



Visualization

Dataframe visualization



PlotlyRequestError: Hi there, you've reached the threshold of 100 combined image exports and chart saves per 24h period. If you need to raise your daily limit, please consider upgrading to a paid plan.