北京郵電大學

实验报告

题　　目：分析新冠疫情数据实验报告

课程名称：	Python程序设计
上课学期：	2021春
授课教师：	杨亚
姓　　名：	刘俊杰
学　　号：	2019213687
日　　期：	2021年12月31日

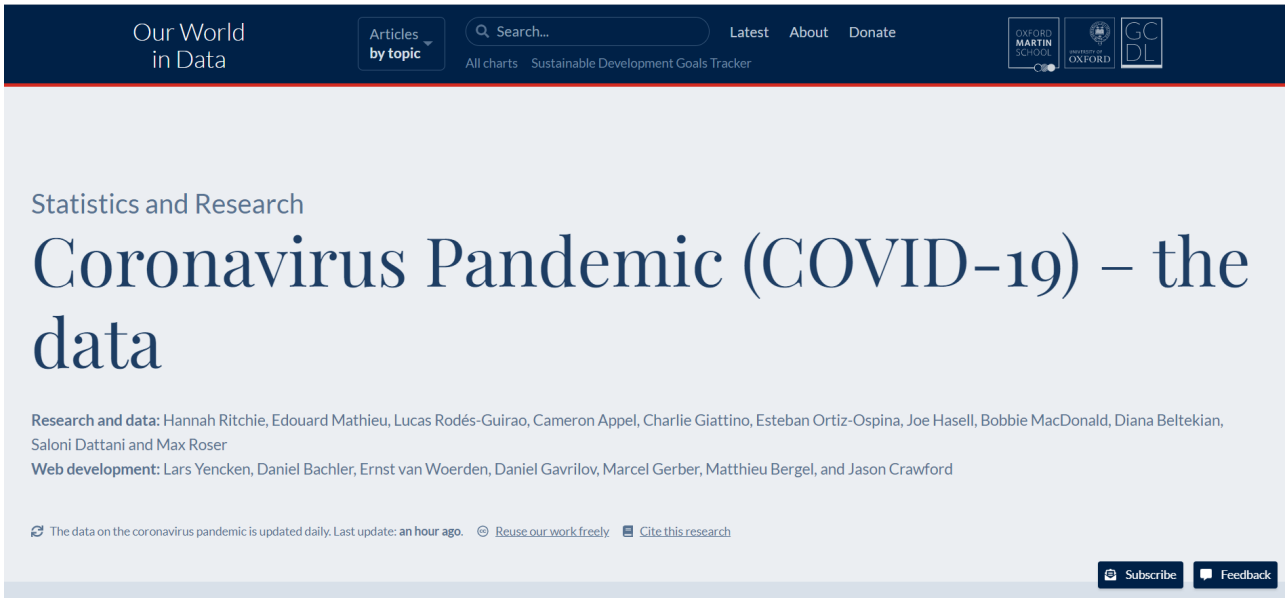
目录

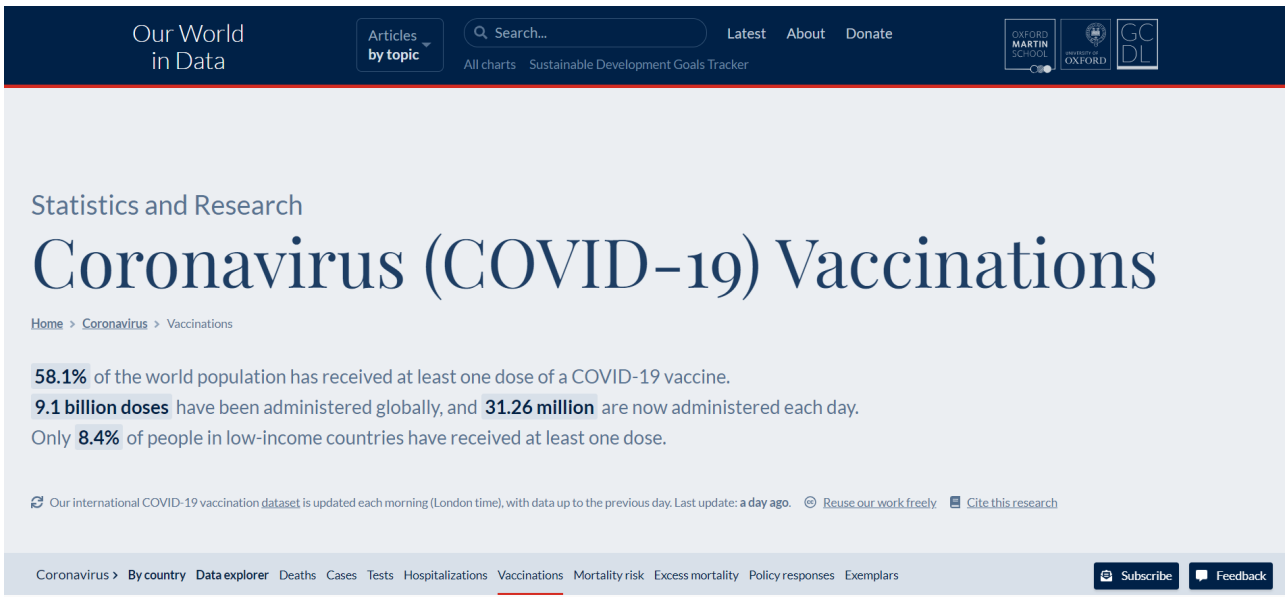
1 实验内容

找一个有全球新冠病毒数据的网站，爬取其中的数据（禁止使用数据接口直接获取数据）。
要求爬取从 2021 年 12 月 5 日开始的连续 15 天的数据，国家数不少于 100 个。

2 数据来源

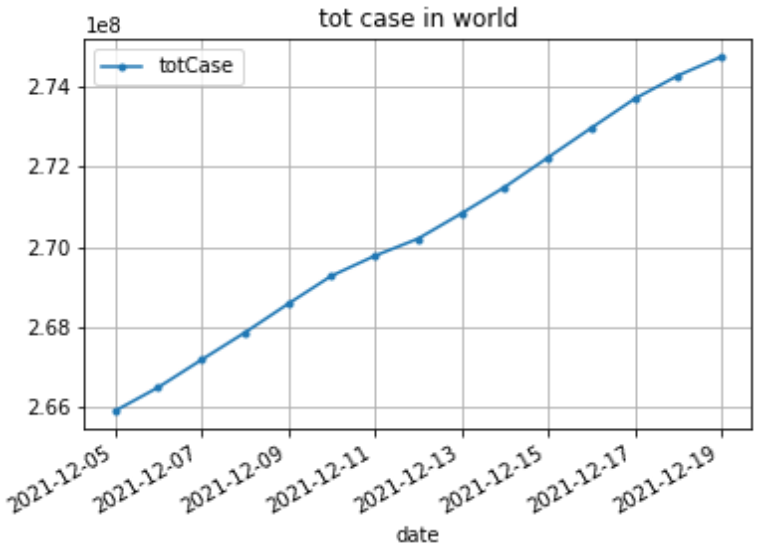
新冠病毒数据网址 URL：https://ourworldindata.org/coronavirus-data
新冠疫苗数据网址 URL：https://ourworldindata.org/covid-vaccinations?country=OWID_WRL
首页截图：





3 数据分析和展示

3.1 15 天中全球新冠疫情的总体变化趋势(totCase.py)

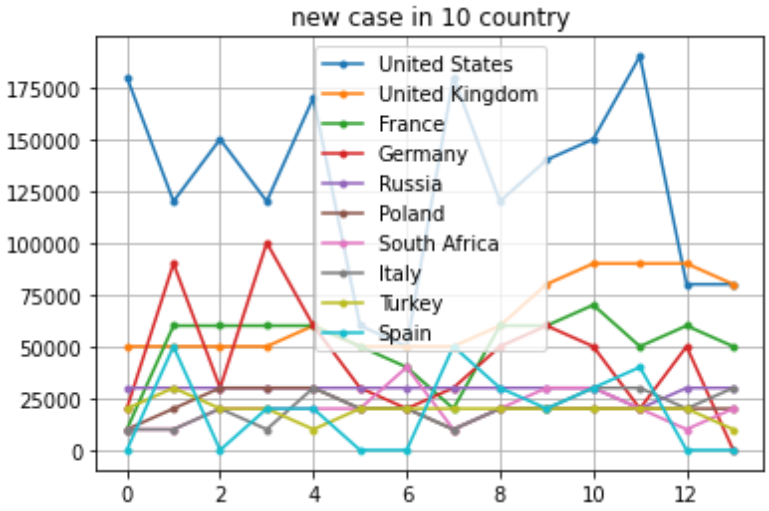


3.2 15 天中每日新增确诊数累计排名前 10 个国家的每日新增确诊数据(newCase.py)

3.2.1 15 天中每日新增确诊数累计排名前 10 个国家

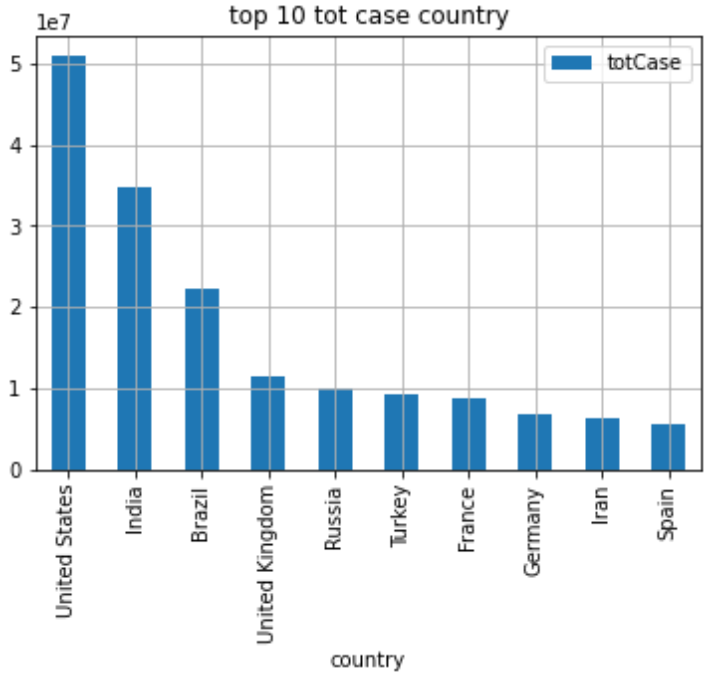
```
country,0
United States,1790000
United Kingdom,900000
France,710000
Germany,610000
Russia,410000
Poland,290000
South Africa,280000
Italy,280000
Turkey,270000
Spain,260000
```

3.2.2 每日新增确诊数据的曲线图



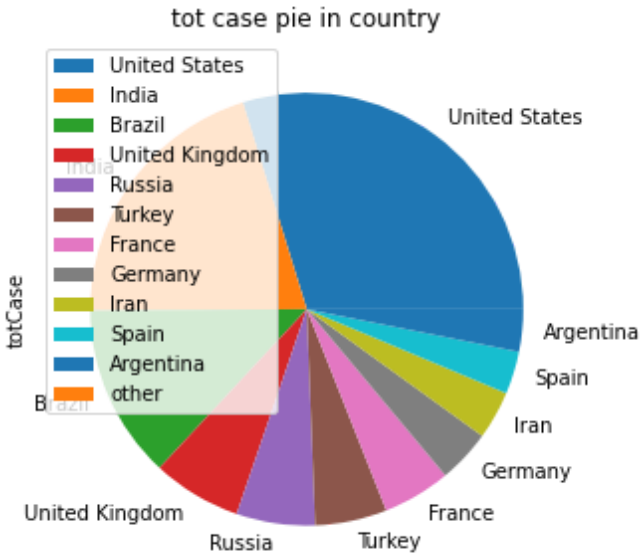
3.3 累计确诊数排名前 10 的国家名称及其数量(totCaseTop10.py)

```
,country,totCase,date
8,United States,50890000,2021-12-19
10,India,34750000,2021-12-19
11,Brazil,22220000,2021-12-19
12,United Kingdom,11380000,2021-12-19
13,Russia,10040000,2021-12-19
15,Turkey,9170000,2021-12-19
16,France,8670000,2021-12-19
17,Germany,6810000,2021-12-19
18,Iran,6170000,2021-12-19
19,Spain,5460000,2021-12-19
```

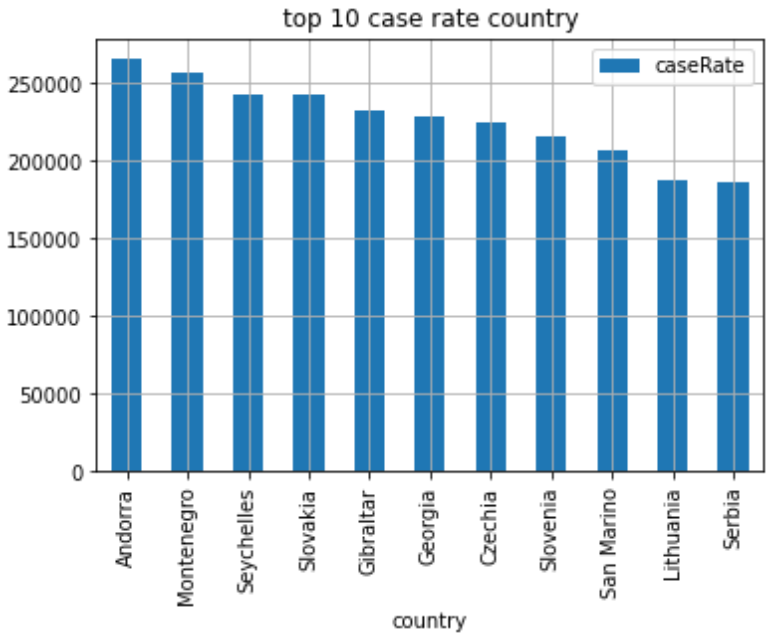


3.4 各个国家的累计确诊人数的比例(totCastProportion.py)

用饼图展示各个国家的累计确诊人数的比例（爬取的所有国家，数据较小的国家可以合并处理）



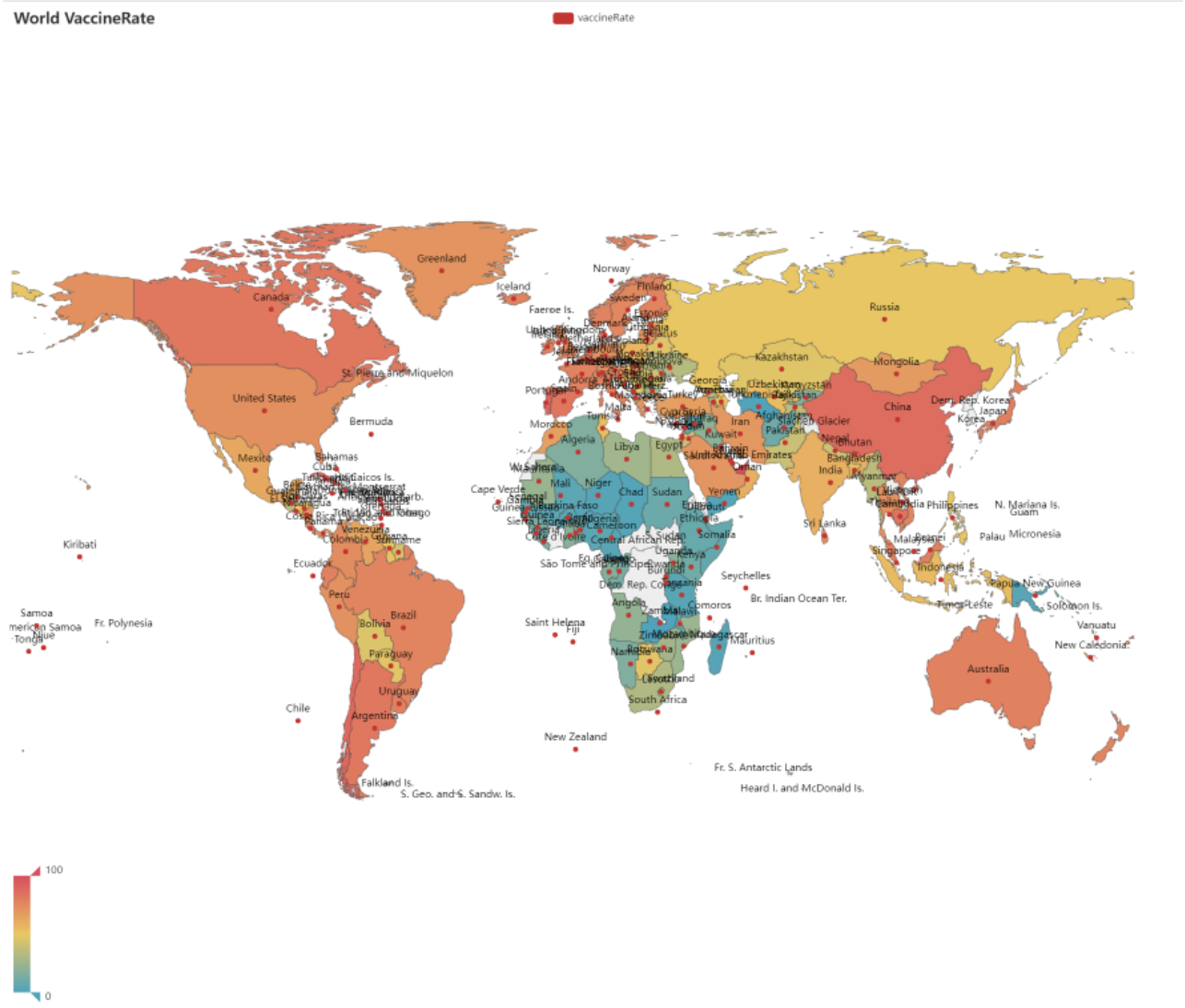
3.5 累计确诊人数占国家总人口比例最高的 10 个国家(caseRate.py)



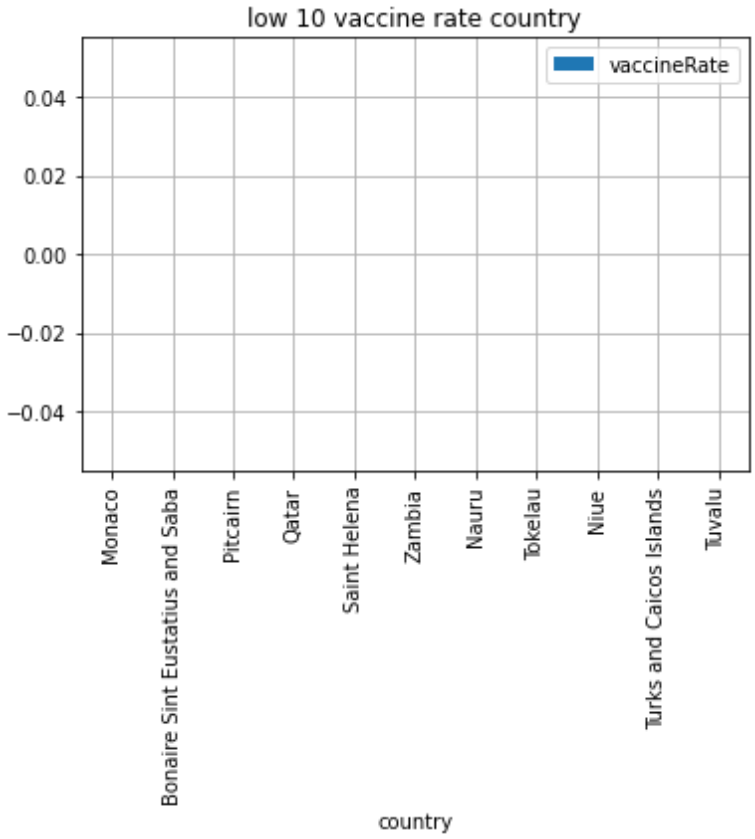
单位为每百万人确诊人数。

3.6 疫苗接种情况（vaccineMap.py）

疫苗接种情况（至少接种了一针及以上），请用地图形式展示；

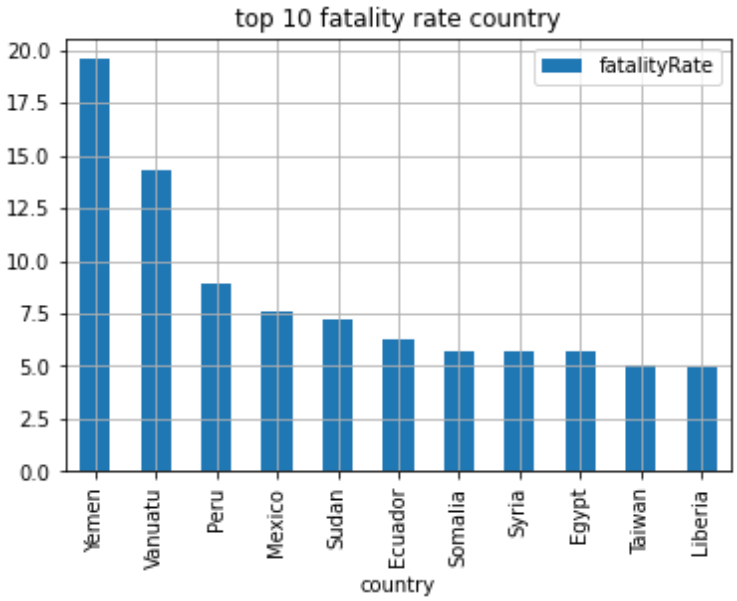


3.7 疫苗接种率（累计疫苗接种/人数国家人数 ）最低的 10 个国家 (vaccineRateLow.py)



3.8 全球 GDP 前十名国家的累计确诊人数箱型图(totCaseInCDPTop10.py)

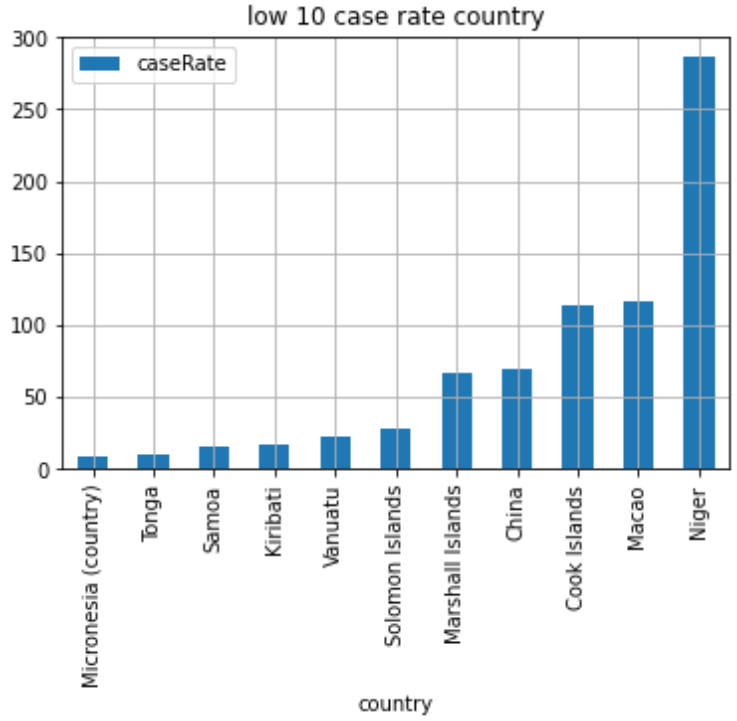
3.9 死亡率最高的 10 个国家(fatalityRate.py)



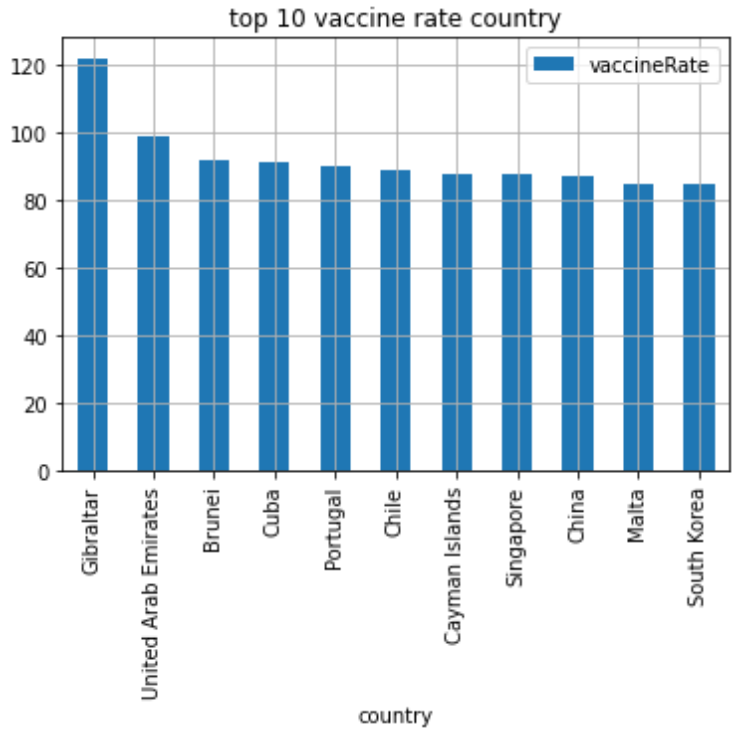
3.10 其它你希望分析和展示的数据

以上图形应包括完整的坐标、刻度、标签、图例等，如有必要请配上说明文字，对图中的内容进行解释。

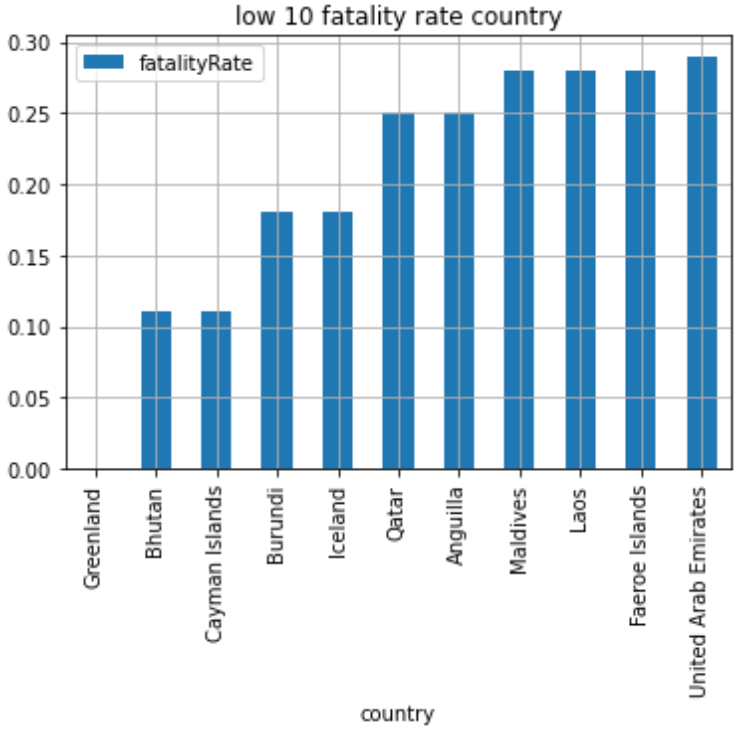
3.10.1 累计确诊人数占国家总人口比例最低的 10 个国家(caseRateLow10.py)



3.10.2 疫苗接种率（累计疫苗接种/人数国家人数）最高的 10 个国家（vaccineRateTop.py）



3.10.3　　死亡率最低的 10 个国家（fatalityRateLow.py）



4　全世界应对新冠疫情最好的 10 个国家

根据以上数据，列出全世界应对新冠疫情最好的 10 个国家，并说明你的理由。

中国确诊人数占国家总人口比例低，疫苗接种率（累计疫苗接种/人数国家人数）高，当之无愧对新冠疫情最好的 10 个国家之一。

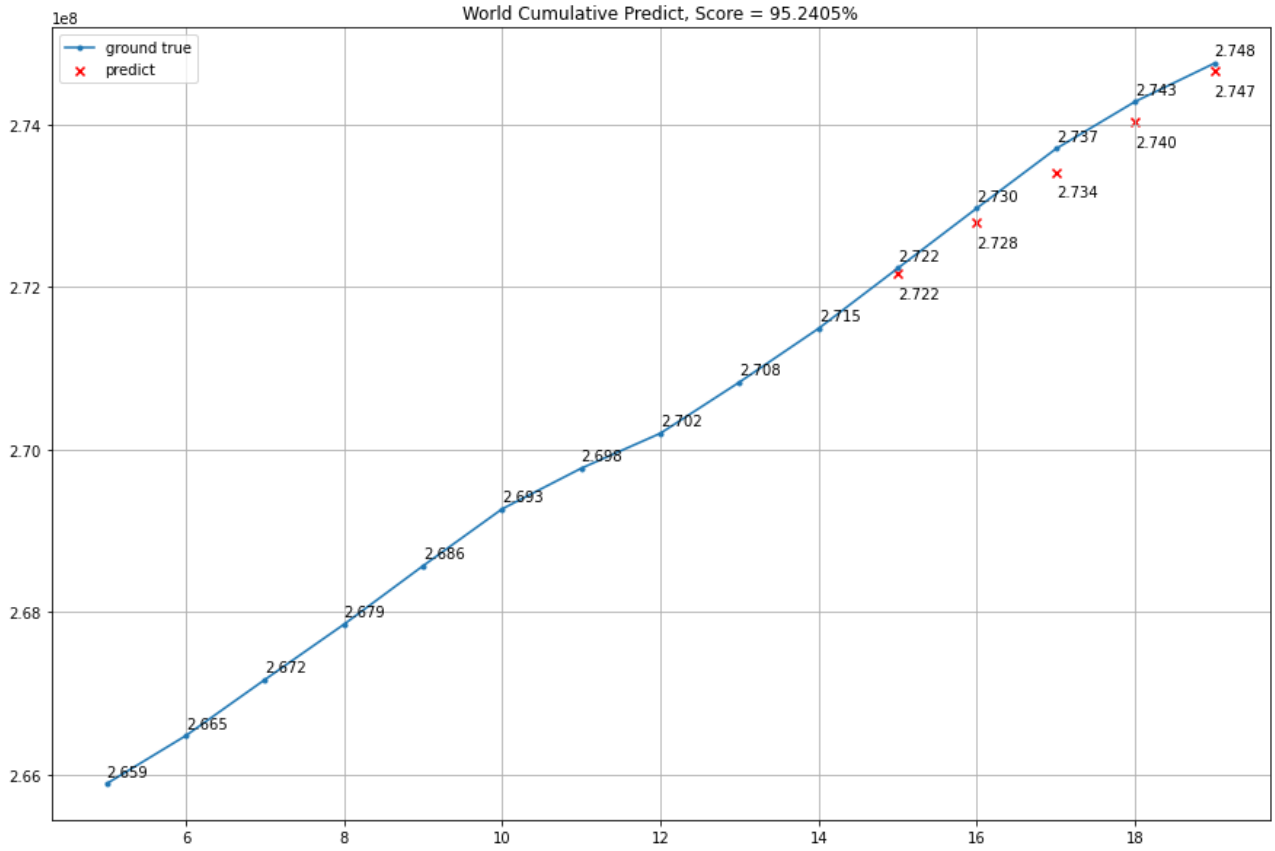
世界主要国家中还有墨西哥确诊率低，阿联酋、古巴、葡萄牙、新加坡、韩国等确诊率低，格陵兰岛、冰岛、老挝、阿联酋等死亡率低。

所以全世界应对新冠疫情最好的 10 个国家是：中国、墨西哥、阿联酋、古巴、葡萄牙、新加坡、韩国、格陵兰岛、冰岛、老挝。

5 预测(totCasePred.py)

针对全球累计确诊数，利用前 10 天采集到的数据做后 5 天的预测，并与实际数据进行对比。说明你预测的方法，并分析与实际数据的差距和原因。

使用 sklearn 中的 linear_model 线性回归模型对确诊数进行预测，真实数据为蓝色点，预测值为红色叉，可以发现预测值比真实值要小一些，产生差距的原因是线性回归模型是比较简单的模型，无法对如此复杂的全球累计确诊数进行精确的预测，现在的预测值已经非常接近真实值。



6 源程序

6.1 爬虫

6.1.1 spider.py

```
import scrapy
import re
from covid19.items import MyItem # 从 items.py 中引入 MyItem 对象

class mySpider(scrapy.spiders.Spider):
    name = "covid19" # 爬虫的名字是 covid19
    allow_domains = ['https://ourworldindata.org/explorers/'] # 允许爬取的网站域名
    # totCase
    start_urls = ['https://ourworldindata.org/explorers/'\
```

```
        'coronavirus-data-explorer?tab=table&zoomToSelection=true&time=2021-12-' + str(date).zfill(2) +
'&facet=none&pickerSort=desc&pickerMetric=total_vaccinations_per_hundred&'\
'Metric=Confirmed+cases&Interval=Cumulative&Relative+to+Population=false&'\
        'Color+by+test+positivity=false' for date in range(5, 20)] #爬取5~19日数据

    # caseRate
    #start_urls = ['https://ourworldindata.org/explorers/'\
    #    'coronavirus-data-explorer?tab=table&time=2021-12-19&facet=none&'\
    #'Metric=Confirmed+cases&Interval=Cumulative&Relative+to+Population=true&Color+by+test+positivity=false']

    # vaccineRate
    #start_urls = ['https://ourworldindata.org/explorers/'\
    #    'coronavirus-data-explorer?tab=table&time=2021-12-19&facet=none&'\
    #'Metric=People+vaccinated&Interval=Cumulative&Relative+to+Population=true&Color+by+test+positivity=false']

    # fatalityRate
    #start_urls = ['https://ourworldindata.org/explorers/'\
    #    'coronavirus-data-explorer?tab=table&time=2021-12-19&facet=none&'\
    #'uniformYAxis=0&pickerSort=asc&pickerMetric=location&Metric=Case+fatality+rate&Interval=Cumulative&Relative+to+Population=true&Color+by+test+positivity=false&country=~OWID_WRL']

    def parse(self, response):
        item = MyItem()
        for each in response.xpath('/html/body/main/div/div[3]/div/div[1]/div/table/tbody/tr'):
            item['country'] = each.xpath('td[1]/text()').get()
            # totCase

            item['totCase'] = each.xpath('td[2]/text()').get().replace(',', '') #去除逗号
            if re.search('million', item['totCase']): #将million转换为数字概念的1e6
                item['totCase'] = item['totCase'].replace(' million', '')
                item['totCase'] = int(float(item['totCase']) * 1000000)
            else:
                item['totCase'] = int(item['totCase'])

            # caseRate
```

```
            # item['caseRate'] =
float(each.xpath('td[2]/text()').get().replace(',', '')) #去除逗号
            # vaccineRate
            #item['vaccineRate'] = each.xpath('td[2]/text()').get()
            #if item['vaccineRate'] == None: # 没有接种疫苗的国家或地区不考虑
            #    continue
            #item['vaccineRate'] = item['vaccineRate'].replace('"', '')
            # fatalityRate
            #item['fatalityRate'] = each.xpath('td[2]/text()').get()
            #if item['fatalityRate'] == None: #没有死亡率的国家认为是0%
            #    item['fatalityRate'] = '0.00%'

            item['date'] = re.findall(".*time=(.*?)&.*", response.url)[0] #获
取爬取的日期
            yield item
```

6.1.2 items.py

```
# Define here the models for your scraped items
#
# See documentation in:
# https://docs.scrapy.org/en/latest/topics/items.html

import scrapy

class MyItem(scrapy.Item):
    # define the fields for your item here like:
    # name = scrapy.Field()
    country = scrapy.Field() #国家或地区名
    date = scrapy.Field() #日期
    # totCase
    totCase = scrapy.Field() #累计确诊病例
    # caseRate
    #caseRate = scrapy.Field() #确诊病例占总人口比
    # vaccineRate
    #vaccineRate = scrapy.Field() #接种至少一针疫苗人数占总人口比
    # fatalityRate
    #fatalityRate = scrapy.Field() #死亡率
    pass
```

6.1.3 middlewares.py

```
# Define here the models for your spider middleware
#
# See documentation in:
# https://docs.scrapy.org/en/latest/topics/spider-middleware.html

from scrapy import signals
```

```
7
8  # useful for handling different item types with a single interface
9  from itemadapter import is_item, ItemAdapter
10 from selenium import webdriver
11 from scrapy.http import HtmlResponse
12 from time import sleep
13
14 class Covid19SpiderMiddleware:
15     # Not all methods need to be defined. If a method is not defined,
16     # scrapy acts as if the spider middleware does not modify the
17     # passed objects.
18
19     @classmethod
20     def from_crawler(cls, crawler):
21         # This method is used by Scrapy to create your spiders.
22         s = cls()
23         crawler.signals.connect(s.spider_opened,
   signal=signals.spider_opened)
24         return s
25
26     def process_spider_input(self, response, spider):
27         # Called for each response that goes through the spider
28         # middleware and into the spider.
29
30         # Should return None or raise an exception.
31         return None
32
33     def process_spider_output(self, response, result, spider):
34         # Called with the results returned from the Spider, after
35         # it has processed the response.
36
37         # Must return an iterable of Request, or item objects.
38         for i in result:
39             yield i
40
41     def process_spider_exception(self, response, exception, spider):
42         # Called when a spider or process_spider_input() method
43         # (from other spider middleware) raises an exception.
44
45         # Should return either None or an iterable of Request or item
   objects.
46         pass
47
48     def process_start_requests(self, start_requests, spider):
49         # Called with the start requests of the spider, and works
50         # similarly to the process_spider_output() method, except
51         # that it doesn’t have a response associated.
52
```

```
        # Must return only requests (not items).
        for r in start_requests:
            yield r

    def spider_opened(self, spider):
        spider.logger.info('Spider opened: %s' % spider.name)


class Covid19DownloaderMiddleware:
    # Not all methods need to be defined. If a method is not defined,
    # scrapy acts as if the downloader middleware does not modify the
    # passed objects.

    @classmethod
    def from_crawler(cls, crawler):
        # This method is used by Scrapy to create your spiders.
        s = cls()
        crawler.signals.connect(s.spider_opened,
signal=signals.spider_opened)
        return s

    def process_request(self, request, spider):
        # Called for each request that goes through the downloader
        # middleware.

        # Must either:
        # - return None: continue processing this request
        # - or return a Response object
        # - or return a Request object
        # - or raise IgnoreRequest: process_exception() methods of
        #   installed downloader middleware will be called

        options = webdriver.ChromeOptions()
        options.add_argument('--headless')
        options.add_argument("--ignore-certificate-error")
        options.add_argument("--ignore-ssl-errors")
        driver = webdriver.Chrome(chrome_options=options)
        driver.get(request.url)
        driver.implicitly_wait(120)
        sleep(10)
        html = driver.page_source
        return HtmlResponse(url=driver.current_url, body=html,
request=request, encoding='utf-8')

    def process_response(self, request, response, spider):
        # Called with the response returned from the downloader.

        # Must either;
```

```
        # - return a Response object
        # - return a Request object
        # - or raise IgnoreRequest
        return response

    def process_exception(self, request, exception, spider):
        # Called when a download handler or a process_request()
        # (from other downloader middleware) raises an exception.

        # Must either:
        # - return None: continue processing this exception
        # - return a Response object: stops process_exception() chain
        # - return a Request object: stops process_exception() chain
        pass

    def spider_opened(self, spider):
        spider.logger.info('Spider opened: %s' % spider.name)

```

6.1.4 pipelines.py

```
from itemadapter import ItemAdapter
import json

class MyPipeline:
    def open_spider(self, spider):
        try:
            self.file = open('totCase.json', 'w', encoding='utf-8')
            #self.file = open('caseRate.json', 'w', encoding='utf-8')
            #self.file = open('vaccineRate.json', 'w', encoding='utf-8')
            #self.file = open('fatalityRate.json', 'w', encoding='utf-8')
            self.file.write('[')
        except Exception as err:
            print(err)

    def process_item(self, item, spider):
        dict_item = dict(item) #生成字典对象
        json_str = json.dumps(dict_item, ensure_ascii=False) + ",\n" #生成
json串
        self.file.write(json_str) #将json串写入到文件中
        return item

    def close_spider(self, spider):
        self.file.write(']')
        self.file.close() #关闭文件
```

6.1.5 settings.py

```
# Scrapy settings for covid19 project
#
# For simplicity, this file contains only settings considered important or
# commonly used. You can find more settings consulting the documentation:
#
#     https://docs.scrapy.org/en/latest/topics/settings.html
#     https://docs.scrapy.org/en/latest/topics/downloader-middleware.html
#     https://docs.scrapy.org/en/latest/topics/spider-middleware.html

BOT_NAME = 'covid19'

SPIDER_MODULES = ['covid19.spiders']
NEWSPIDER_MODULE = 'covid19.spiders'


# Crawl responsibly by identifying yourself (and your website) on the user-agent
#USER_AGENT = 'covid19 (+http://www.yourdomain.com)'

# Obey robots.txt rules
# ROBOTSTXT_OBEY = True
ROBOTSTXT_OBEY = False

# Configure maximum concurrent requests performed by Scrapy (default: 16)
#CONCURRENT_REQUESTS = 32

# Configure a delay for requests for the same website (default: 0)
# See https://docs.scrapy.org/en/latest/topics/settings.html#download-delay
# See also autothrottle settings and docs
DOWNLOAD_DELAY = 3
# The download delay setting will honor only one of:
#CONCURRENT_REQUESTS_PER_DOMAIN = 16
#CONCURRENT_REQUESTS_PER_IP = 16

# Disable cookies (enabled by default)
#COOKIES_ENABLED = False

# Disable Telnet Console (enabled by default)
#TELNETCONSOLE_ENABLED = False

# Override the default request headers:
#DEFAULT_REQUEST_HEADERS = {
#   'Accept':
'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
#   'Accept-Language': 'en',
#}
```

```
45
46  # Enable or disable spider middlewares
47  # See https://docs.scrapy.org/en/latest/topics/spider-middleware.html
48  # SPIDER_MIDDLEWARES = {
49  #     'covid19.middlewares.Covid19SpiderMiddleware': 543,
50  # }
51
52  # Enable or disable downloader middlewares
53  # See https://docs.scrapy.org/en/latest/topics/downloader-middleware.html
54  DOWNLOADER_MIDDLEWARES = {
55      'covid19.middlewares.Covid19DownloaderMiddleware': 543,
56  }
57
58  # Enable or disable extensions
59  # See https://docs.scrapy.org/en/latest/topics/extensions.html
60  #EXTENSIONS = {
61  #    'scrapy.extensions.telnet.TelnetConsole': None,
62  #}
63
64  # Configure item pipelines
65  # See https://docs.scrapy.org/en/latest/topics/item-pipeline.html
66  #ITEM_PIPELINES = {
67  #    'covid19.pipelines.Covid19Pipeline': 300,
68  #}
69  ITEM_PIPELINES = {
70      'covid19.pipelines.MyPipeline': 300,
71  }
72
73  # Enable and configure the AutoThrottle extension (disabled by default)
74  # See https://docs.scrapy.org/en/latest/topics/autothrottle.html
75  AUTOTHROTTLE_ENABLED = True
76  # The initial download delay
77  AUTOTHROTTLE_START_DELAY = 5
78  # The maximum download delay to be set in case of high latencies
79  AUTOTHROTTLE_MAX_DELAY = 60
80  # The average number of requests Scrapy should be sending in parallel to
81  # each remote server
82  AUTOTHROTTLE_TARGET_CONCURRENCY = 1.0
83  # Enable showing throttling stats for every response received:
84  #AUTOTHROTTLE_DEBUG = False
85
86  # Enable and configure HTTP caching (disabled by default)
87  # See https://docs.scrapy.org/en/latest/topics/downloader-
    middleware.html#httpcache-middleware-settings
88  #HTTPCACHE_ENABLED = True
89  #HTTPCACHE_EXPIRATION_SECS = 0
90  #HTTPCACHE_DIR = 'httpcache'
91  #HTTPCACHE_IGNORE_HTTP_CODES = []
```

```
#HTTPCACHE_STORAGE = 'scrapy.extensions.httpcache.FilesystemCacheStorage'

```

6.2 数据处理及展示

6.2.1 totCase.py

```
#%%
import pandas as pd
totCase = pd.read_json('./covid19/totCase.json')
totCaseWrold =  totCase.loc[totCase['country'] == 'World'].drop(columns=['country']).set_index('date')
totCaseWrold.plot(marker='.', grid=True)
# %%

```

6.2.2 newCase.py

```
#%%
import pandas as pd
totCase = pd.read_json('./covid19/totCase.json')
newCase = totCase.groupby('country')['totCase'].agg(['min','max'])
newCase = (newCase['max'] - newCase['min']).sort_values(ascending=False)
print(newCase)
newCase.to_csv('newCase.csv', index=True)
newCases = []
countrys = ['United States','United Kingdom','France','Germany','Russia','Poland','South Africa','Italy','Turkey','Spain']
for country in countrys:
    newCase = totCase.loc[totCase['country'] == country]
    newCase = newCase.drop(columns=['country']).set_index('date')
    newCase = newCase.loc[['2021-12-' + str(date).zfill(2) for date in range(6, 20)]].reset_index(drop=True). \
        subtract(newCase.loc[['2021-12-' + str(date).zfill(2) for date in range(5, 19)]].reset_index(drop=True))
    newCase = newCase.rename({'totCase':country}, axis=1)
    newCases.append(newCase)
df = pd.concat(newCases)
df.plot(title='new case in 10 country', marker='.', grid=True)
# %%

```

6.2.3 totCaseTop10.py

```
#%%
import pandas as pd
totCase = pd.read_json('./covid19/totCase.json')
totCaseTop =  totCase.loc[totCase['date'] == '2021-12-19'].sort_values(by=['totCase'], ascending=False).reset_index(drop=True)
print(totCaseTop)
totCaseTop.to_csv('totCaseTop10.csv', index=True)
Top10Country = ["United States","India","Brazil","United Kingdom","Russia","Turkey","France","Germany","Iran","Spain"]
totCaseTop10 = totCaseTop.loc[totCaseTop['country'].isin(Top10Country)]
totCaseTop10 = totCaseTop10.set_index('country')[['totCase']]
print(totCaseTop10)
totCaseTop10.plot.bar(title='top 10 tot case country', grid=True)
# %%

```

6.2.4 totCastProportion.py

```
#%%
import pandas as pd
totCase = pd.read_json('./covid19/totCase.json')
totCaseTop =  totCase.loc[totCase['date'] == '2021-12-19'].sort_values(by=['totCase'], ascending=False).reset_index(drop=True).set_index('country')[['totCase']]
dropList = ['World', 'High income', 'Upper middle income', 'Asia', 'Europe', 'Lower middle income', 'North America', 'European Union', 'South America', 'Africa']
totCaseTop = totCaseTop.drop(dropList, axis=0).reset_index()
totCaseCount = totCaseTop.loc[0:10,:]
totCaseCount = totCaseCount.set_index('country')
totCaseCount.loc['other'] = totCaseTop.loc[10:,:].sum(axis=1)
totCaseCount.plot.pie(title='tot case pie in country', y='totCase', figsize=(5, 5))
# %%

```

6.2.5 caseRate.py

```
#%%
import pandas as pd
caseRate = pd.read_json('./covid19/caseRate.json')
caseRateTop = caseRate.sort_values(by=['caseRate'],
ascending=False).set_index('country')
print(caseRateTop)
dropList = ['World', 'High income', 'Upper middle income', 'Asia', 'Europe',
'Lower middle income', 'North America', 'European Union', 'South America',
'Africa']
caseRateTop = caseRateTop.drop(dropList,
axis=0).reset_index().loc[0:10,:].set_index('country')[['caseRate']]
caseRateTop.plot.bar(title='top 10 case rate country', grid=True)
# %%

```

6.2.6 vaccineMap.py

```
#%%
import pandas as pd
from pyecharts import options as opts
from pyecharts.charts import Map
import imgkit
vaccineRate = pd.read_json('./covid19/vaccineRate.json')
print(vaccineRate)
for i in vaccineRate.index:
    vaccineRate['vaccineRate'][i] = float(vaccineRate['vaccineRate']
[i].replace('%', ''))
vaccineRate = vaccineRate.sort_values(by=['vaccineRate'],
ascending=False).set_index('country')
dropList = ['World', 'High income', 'Upper middle income', 'Asia', 'Europe',
'Lower middle income', 'North America', 'European Union', 'South America',
'Africa']
vaccineRate = vaccineRate.drop(dropList, axis=0)
[['vaccineRate']].reset_index()

countries = list([vaccineRate['country'][i] for i in vaccineRate.index])
vaccineRates = list([vaccineRate['vaccineRate'][i] for i in
vaccineRate.index])
map = Map(init_opts=opts.InitOpts(width='100%', height='1200px'))
map.add("vaccineRate", [list(z) for z in zip(countries, vaccineRates)],
'world')
map.set_global_opts(
    title_opts=opts.TitleOpts(title='World VaccineRate'), #标题
    visualmap_opts=opts.VisualMapOpts(min_=0,max_=100)) #热力图数值区间
map.set_series_opts(label_opts=opts.LabelOpts(is_show=True)) #热力图图例
map.render("map.html") #导出html
imgkit.from_file('./map.html', './map.jpg')

```

```
# %%

```

6.2.7 vaccineRateLow.py

```
#%%
import pandas as pd
vaccineRate = pd.read_json('./covid19/vaccineRate.json')
print(vaccineRate)
for i in vaccineRate.index:
    vaccineRate['vaccineRate'][i] = float(vaccineRate['vaccineRate']
[i].replace('%', ''))
vaccineRateLow = vaccineRate.sort_values(by=
['vaccineRate']).set_index('country')
dropList = ['World', 'High income', 'Upper middle income', 'Asia', 'Europe',
'Lower middle income', 'North America', 'European Union', 'South America',
'Africa']
vaccineRateLow = vaccineRateLow.drop(dropList,
axis=0).reset_index().loc[0:10,:].set_index('country')[['vaccineRate']]
vaccineRateLow.plot.bar(title='low 10 vaccine rate country', grid=True)
# %%

```

6.2.8 totCaseInCDPTop10.py

```
#%%
import pandas as pd
caseRate = pd.read_json('./covid19/totCase.json')
GDPTop10 = ['United States', 'China', 'Japan', 'Germany', 'United Kingdom',
'India', 'France', 'Italy', 'Canada', 'South Korea']
caseRate = caseRate.loc[caseRate['date'] == '2021-12-19']
caseRate = caseRate.loc[caseRate['country'].isin(GDPTop10)]
print(caseRate)
caseRate = caseRate.set_index('country')['totCase']
caseRate.plot.box(title='case rate in GDP top 10 country', grid=True)
# %%

```

6.2.9 fatalityRate.py

```
#%%
import pandas as pd
fatalityRate = pd.read_json('./covid19/fatalityRate.json')
for i in fatalityRate.index:
    fatalityRate['fatalityRate'][i] = float(fatalityRate['fatalityRate']
[i].replace('%', ''))
fatalityRateTop =  fatalityRate.sort_values(by=['fatalityRate'],
ascending=False).reset_index(drop=True).set_index('country')
dropList = ['World', 'High income', 'Upper middle income', 'Asia', 'Europe',
'Lower middle income', 'North America', 'European Union', 'South America',
'Africa']
fatalityRateTop = fatalityRateTop.drop(dropList,
axis=0).reset_index().loc[0:10,:].set_index('country')[['fatalityRate']]
fatalityRateTop.plot.bar(title='top 10 fatality rate country', grid=True)
# %%

```

6.2.10 caseRateLow10.py

```
#%%
import pandas as pd
caseRate = pd.read_json('./covid19/caseRate.json')
caseRateLow = caseRate.sort_values(by=['caseRate']).set_index('country')
dropList = ['World', 'High income', 'Upper middle income', 'Asia', 'Europe',
'Lower middle income', 'North America', 'European Union', 'South America',
'Africa']
caseRateLow = caseRateLow.drop(dropList,
axis=0).reset_index().loc[0:10,:].set_index('country')[['caseRate']]
caseRateLow.plot.bar(title='low 10 case rate country', grid=True)
# %%

```

6.2.11 vaccineRateTop.py

```
#%%
import pandas as pd
vaccineRate = pd.read_json('./covid19/vaccineRate.json')
print(vaccineRate)
for i in vaccineRate.index:
    vaccineRate['vaccineRate'][i] = float(vaccineRate['vaccineRate']
[i].replace('%', ''))
vaccineRateLow = vaccineRate.sort_values(by=['vaccineRate'],
ascending=False).set_index('country')
dropList = ['World', 'High income', 'Upper middle income', 'Asia', 'Europe',
'Lower middle income', 'North America', 'European Union', 'South America',
'Africa']
vaccineRateLow = vaccineRateLow.drop(dropList,
axis=0).reset_index().loc[0:10,:].set_index('country')[['vaccineRate']]
vaccineRateLow.plot.bar(title='top 10 vaccine rate country', grid=True)
# %%

```

6.2.12 fatalityRateLow.py

```
#%%
import pandas as pd
fatalityRate = pd.read_json('./covid19/fatalityRate.json')
for i in fatalityRate.index:
    fatalityRate['fatalityRate'][i] = float(fatalityRate['fatalityRate']
[i].replace('%', ''))
fatalityRateTop =  fatalityRate.sort_values(by=
['fatalityRate']).reset_index(drop=True).set_index('country')
dropList = ['World', 'High income', 'Upper middle income', 'Asia', 'Europe',
'Lower middle income', 'North America', 'European Union', 'South America',
'Africa']
fatalityRateTop = fatalityRateTop.drop(dropList,
axis=0).reset_index().loc[0:10,:].set_index('country')[['fatalityRate']]
fatalityRateTop.plot.bar(title='low 10 fatality rate country', grid=True)
# %%

```

6.2.13 totCasePred.py

```
#%%
import pandas as pd
import numpy as np
from sklearn import linear_model
from matplotlib import pyplot as plt
totCase = pd.read_json('./covid19/totCase.json')
for i in totCase.index:
    totCase['date'][i] = int(str(totCase['date'][i]).split(' ')[0].split('-
')[2])
```

```
totCaseWrold =  totCase.loc[totCase['country'] == 'World'].drop(columns=['country']).sort_values('date').set_index('date')
x = []
y = []
plt.figure(figsize=(15, 10))
for i in totCaseWrold.index:
    x.append(i)
    y.append(totCaseWrold['totCase'][i])
date = x.copy()
x2 = x[10:]
x = x[:10]
y2 = y[10:]
y = y[:10]
plt.grid(True)
plt.plot(date, y + y2, marker='.', label='ground true')
for a,b in zip(date, y + y2):
    plt.text(a, b+1e5, '%.3f'%(b/1e8))
x2 = [_ for _ in range(15, 20)]
model = linear_model.LinearRegression()
model.fit(np.array(x).reshape(-1, 1), y)
y3 = model.predict(np.array(x2).reshape(-1, 1))
plt.scatter(date[10:], y3, c='red', marker='x', label='predict')
for a,b in zip(date[10:], y3):
    plt.text(a, b-3e5, '%.3f'%(b/1e8))
plt.legend(loc=2)
score = model.score(np.array(x2).reshape(-1, 1), y2)
plt.title('World Cumulative Predict, Score = ' + str('%.4f'%(score*100)) + '%')
plt.show()

#totCaseWrold.plot(title='tot case in world', marker='.', grid=True)
# %%

```

7 总结

通过爬虫得到新冠病毒和疫苗的数据并保存到 json 文件，使用 Python 的 Pandas 库对数据进行清洗、提取、排序和统计，然后使用 Dataframe 结构 plot 方法对数据进行可视化。要进行更进一步的可视化，可以使用 pyecharts 库绘制热力图。要利用现有的数据对未来进行预测，可以使用 Sklearn 库进行预测，该库包含了众多经典机器学习算法，可以很方便使用。

通过上述的工作，利用网络数据，可以让我们进一步分析和了解新冠疫情的情况，也能通过现有数据预测疫情的发展情况，有利于我们更好地应对疫情。通过这次实验，进一步熟悉了数据获取和分析的工具和方法，进一步加深了对 Python 的理解，受益匪浅。