江西财经大學
JIANGXI UNIVERSITY OF FINANCE AND ECONOMICS

课程名称：Python语言与数据分析

课 程 报 告

项目名称 新冠肺炎疫情数据分析与可视化

班　　级 金融201

学　　号 0204766

姓　　名 张艺轩

任课教师 肖　泉

开课学期： 2020 至 2021 学年 第 2 学期

完成时间： 2021 年 6 月 18 日

《新冠肺炎疫情数据分析与可视化》数据分析报告

目　录

1 概述

19 世纪中期，英国伦敦西部爆发了霍乱，英国著名流行病传播控制学家约翰·斯诺（John·Snow）用一幅标注霍乱地区水泵位置和病亡人数的地图发现了霍乱“由水传播”的证据，并成功劝服伦敦政府取下了“宽街”（Broad Street）水泵的把手，疫情因而得到了控制。

而此次新冠肺炎疫情，是新中国成立以来，在我国遭遇的传播速度最快、感染范围最广、防控难度最大的一次突发公共卫生事件。数据分析与数据可视化再一次释放了不可小觑的力量。我们必须充分利用利用数据分析与数据可视化工具，结合疫情数据，全面对疫情传播在我国呈现的时间分布特点和地域分布特点。

本报告主要运用了 Python 开发环境的 Pandas 工具和 Matplotlib 数据可视化工具，对我国 2020 年 1 月初至 7 月末的各省疫情宏观数据（包括“累计确诊”“现有疑似”“累计治愈”“新增死亡”等数据项目）进行了简要分析，旨在发掘 2020 年全年疫情传播的纵向特点和横向特点。

2 数据描述

（1）数据来源：数据来源于 Github Repo DXY-COVID-19-Data，原作者为 Isaac·Lin。

（2）数据特点：该数据共有 232217 行，19 个字段。而本数据分析仅截取中国的疫情宏观数据，结合'provinceName'、'province_confirmedCount'、'province_suspectedCount’ 、

'province_curedCount'、'province_deadCount'等 5 个字段进行分析。

（3）数据清洗：

```
RangeIndex: 232217 entries, 0 to 232216
Data columns (total 19 columns):
continentName              232173 non-null object
continentEnglishName       232173 non-null object
countryName                232217 non-null object
countryEnglishName         222905 non-null object
provinceName               232217 non-null object
provinceEnglishName        222905 non-null object
province_zipCode           232217 non-null int64
province_confirmedCount    232217 non-null int64
province_suspectedCount    232214 non-null float64
province_curedCount        232217 non-null int64
province_deadCount         232217 non-null int64
updateTime                 232217 non-null datetime64[ns]
cityName                   90784 non-null object
cityEnglishName            87815 non-null object
city_zipCode               89518 non-null float64
city_confirmedCount        90784 non-null float64
city_suspectedCount        90784 non-null float64
city_curedCount            90784 non-null float64
city_deadCount             90784 non-null float64
dtypes: datetime64[ns](1), float64(6), int64(4), object(8)
```

图 2.1

```
None
各字段是否含有空值情况：
 continentName               True
continentEnglishName         True
countryName                 False
countryEnglishName           True
provinceName                False
provinceEnglishName          True
province_zipCode            False
province_confirmedCount     False
province_suspectedCount      True
province_curedCount         False
province_deadCount          False
updateTime                  False
cityName                     True
cityEnglishName              True
city_zipCode                 True
city_confirmedCount          True
city_suspectedCount          True
city_curedCount              True
city_deadCount               True
dtype: bool
```

图 2.2

由图 2.1 和图 2.2 可知，原始数据所含空值行较多，为不影响数据分析效果，须将 nan 值进行一处，同时还应剔除无用的字段。

此外在获取数据时要注意数据的类型，该文件的时间字段为文本类型，特别是日期字段的数据，整理数据时可以将其转换成时间格式，以方便后续的数据处理。执行数据清洗代码如下：

```
data['updateTime']=pd.to_datetime(data['updateTime'])
print(data.info())
print('各字段是否含有空值情况：\n',data.isna().any())
#取出中国的数据
ChinaData=data[data['countryName']=='中国']
#剔除空值行
ChinaData1=ChinaData.dropna()
#去除无用的列，无用的行
ChinaData1.drop(['continentEnglishName'],axis=1,inplace=True)
ChinaData1.drop(['provinceEnglishName'],axis=1,inplace=True)
ChinaData1.drop(['province_zipCode'],axis=1,inplace=True)
ChinaData1.drop(['cityEnglishName'],axis=1,inplace=True)
ChinaData1.drop(['city_zipCode'],axis=1,inplace=True)
print(ChinaData1.info())
print('各字段是否含有空值情况：\n',ChinaData1.isna().any())
```

去除后的数据如下所示：

```
Int64Index: 87577 entries, 163 to 232086
Data columns (total 14 columns):
continentName              87577 non-null object
countryName                87577 non-null object
countryEnglishName         87577 non-null object
provinceName               87577 non-null object
province_confirmedCount    87577 non-null int64
province_suspectedCount    87577 non-null float64
province_curedCount        87577 non-null int64
province_deadCount         87577 non-null int64
updateTime                 87577 non-null datetime64[ns]
cityName                   87577 non-null object
city_confirmedCount        87577 non-null float64
city_suspectedCount        87577 non-null float64
city_curedCount            87577 non-null float64
city_deadCount             87577 non-null float64
dtypes: datetime64[ns](1), float64(5), int64(3), object(5)
memory usage: 10.0+ MB
```

图 2.3

```
各字段是否含有空值情况:
 continentName                False
countryName                   False
countryEnglishName            False
provinceName                  False
province_confirmedCount       False
province_suspectedCount       False
province_curedCount           False
province_deadCount            False
updateTime                    False
cityName                      False
city_confirmedCount           False
city_suspectedCount           False
city_curedCount               False
city_deadCount                False
dtype: bool
```

图 2.4

剔除后的数据包含 87577 项非空行，共计 13 个字段，其中 float64 有 5 列，int64 有 3 列，object 有 5 列。为了便于后面分组统计，我们应将月份作为单独日期列出。代码如下：

```
ChinaData1['update_month']= ChinaData['updateTime'].dt.month
```

3 数据分析内容

（1）我国 2020 年 1-7 月疫情确诊人数累计确诊人数变动情况：

该项数据分析通过柱形图和折线图来观测我国 2020 年 1 月至 7 月疫情变动月确诊人数和确诊人数变动率来大致地分析全国疫情传播的纵向特点，以此更好地来判别疫情的“拐点”和“驻点”，更有预见性地预估未来疫情走向，借以更加准确地定位我国疫情防控工作的具体阶段和具体方针。该项数据未剔除湖北省数据。

（2）我国湖北省 2020 年 1 月至 7 月疫情数据概览：

众所周知，湖北省是 2020 年初新冠疫情爆发的核心地区，是年初疫情防控的重点地区。该项数据分析旨在通过用折线图的方式来呈现 2020 年 1 月至 7 月河北省累计确诊、累计治愈人数、累计死亡人

数以及新增疑似患者人数的变动情况，以此科学评价湖北省这个特殊省份疫情防控工作的巨大成效。

（3）我国不同省份治愈率差异（除湖北省外）：

防止极端值对数据分析的影响，该项数据分析去除了湖北省、西藏自治区、宁夏回族自治区和青海省等省份。该项数据分析已治愈率折线图和各省累计确诊人数与累计治愈人数的比较柱形图来分析治愈情况的差异以及背后的深层次原因。

（4）我国不同省份和地区疫情情况的比较（除湖北省外）：

该项数据分析同样去除了湖北省数据，以饼图的形式来东部、中部和西部确诊人数的差异，以此揭示我国疫情传播的地域特点和疫情地域分布的深层次原因，以此为疫情防控提出建设性的建议。

4 数据分析图表

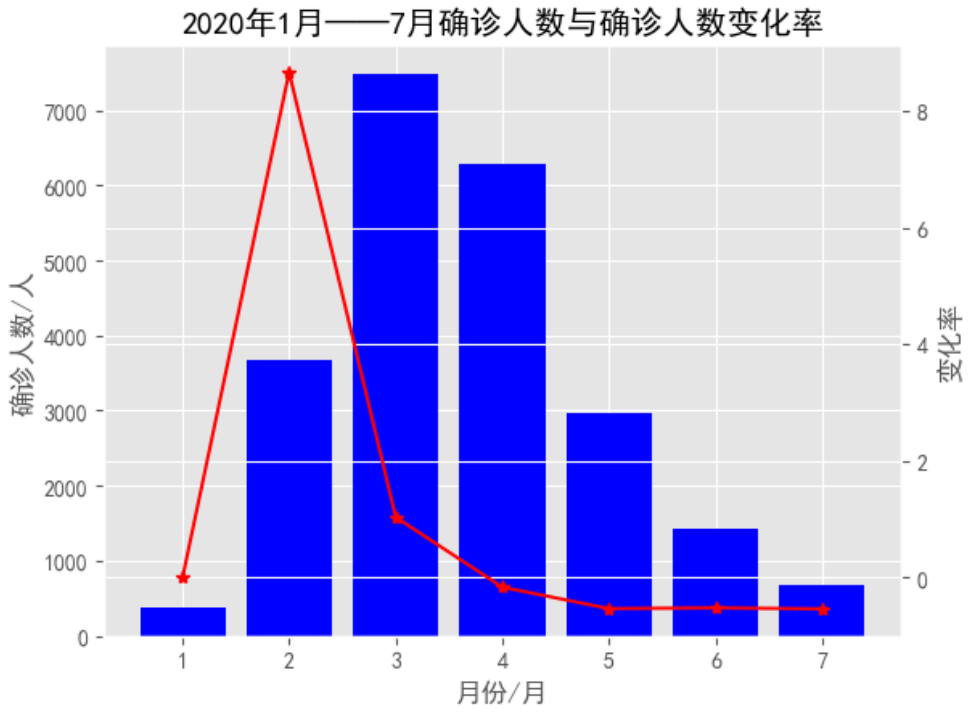

图 4.1

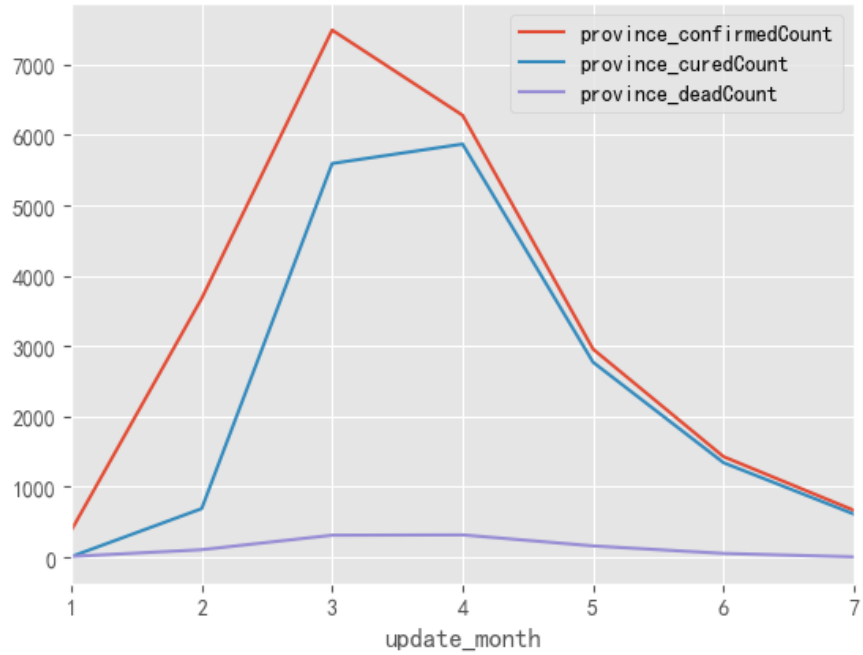

图 4.2

湖北省累计确诊、新增疑似病例、累计治愈病例、累计死亡病例统计图

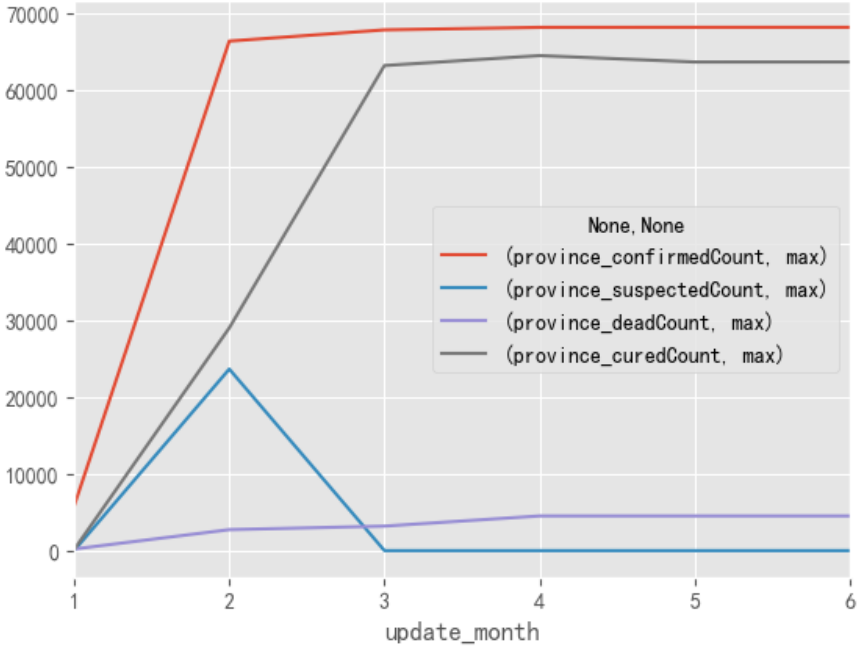

图 4.3

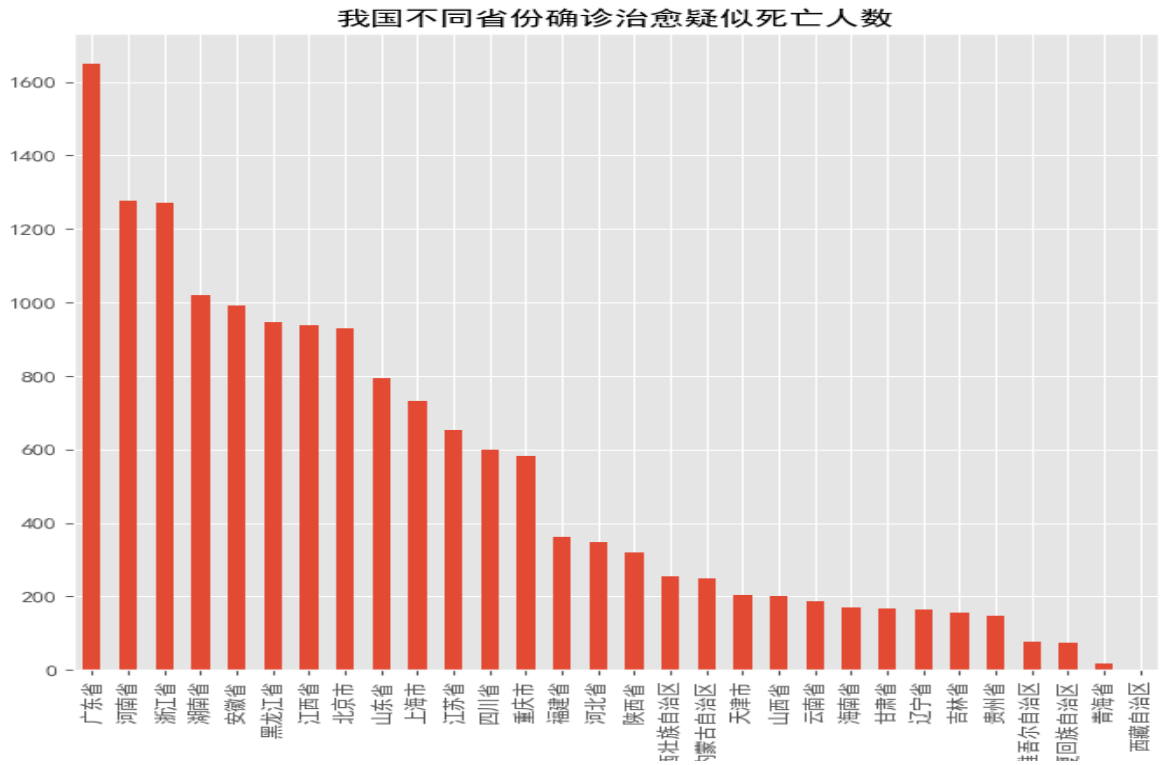
我国不同省份确诊治愈疑似死亡人数
1600
1400
1200
1000
800
600
400
200
0
广东省
河南省
浙江省
湖南省
安徽省
黑龙江省
江西省
北京市
山东省
上海市
江苏省
四川省
重庆市
福建省
河北省
陕西省
广西壮族自治区
内蒙古自治区
天津市
山西省
云南省
海南省
甘肃省
辽宁省
吉林省
贵州省
新疆维吾尔自治区
宁夏回族自治区
青海省
西藏自治区

图 4.4

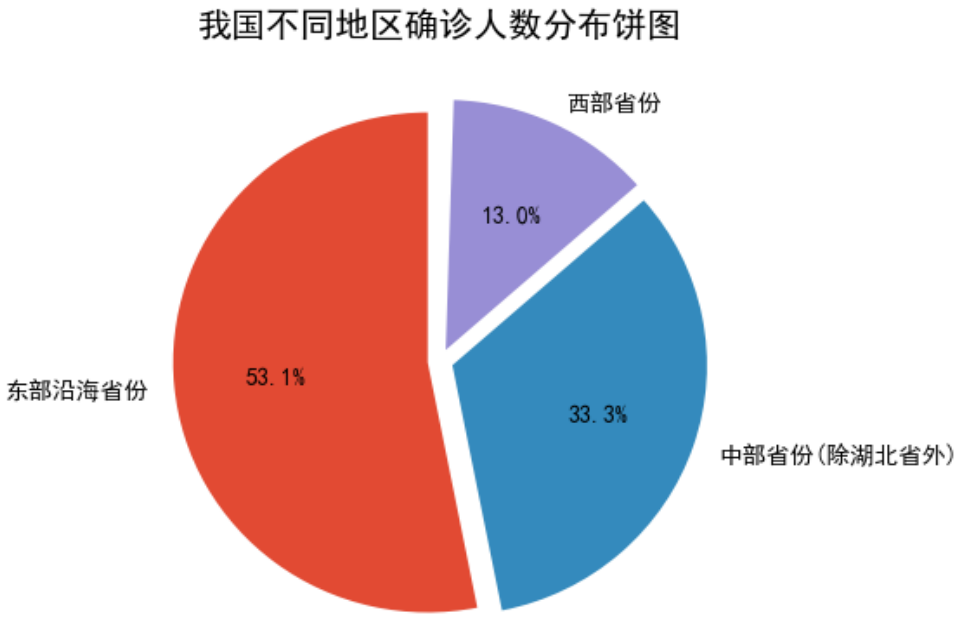
我国不同地区确诊人数分布饼图
西部省份
13.0%
33.3%
中部省份(除湖北省外)
53.1%
东部沿海省份

图 4.5

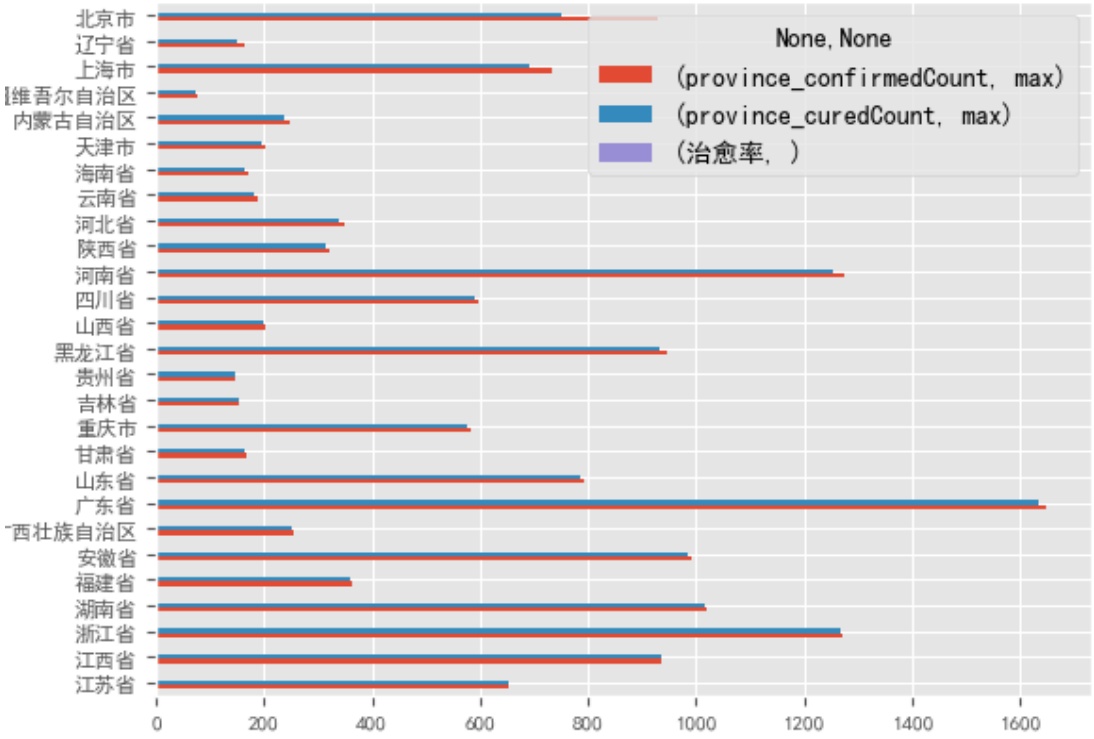
北京市
辽宁省
上海市
维吾尔自治区
内蒙古自治区
天津市
海南省
云南省
河北省
陕西省
河南省
四川省
山西省
黑龙江省
贵州省
吉林省
重庆市
甘肃省
山东省
广东省
西壮族自治区
安徽省
福建省
湖南省
浙江省
江西省
江苏省
0
200
400
600
800
1000
1200
1400
1600
None, None
(province_confirmedCount, max)
(province_curedCount, max)
(治愈率,)

图 4.6

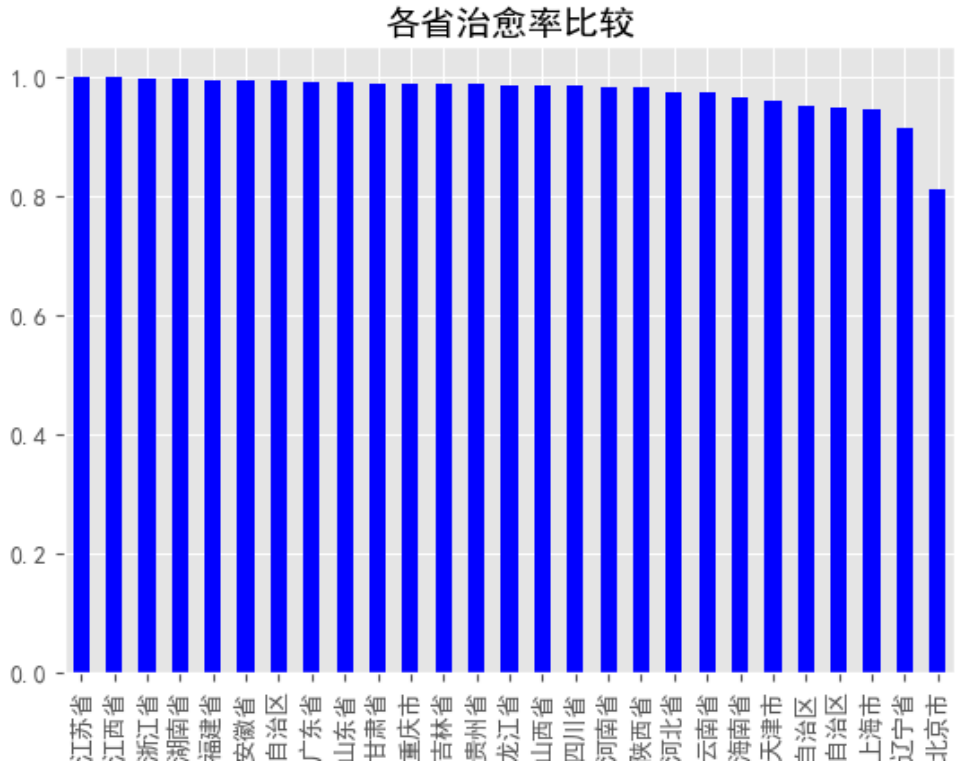
各省治愈率比较
1.0
0.8
0.6
0.4
0.2
0.0
江苏省
江西省
浙江省
湖南省
福建省
安徽省
自治区
广东省
山东省
甘肃省
重庆市
吉林省
贵州省
龙江省
山西省
四川省
河南省
陕西省
河北省
云南省
海南省
天津市
治区
自治区
上海市
辽宁省
北京市

图 4.7

5 数据分析结果

（1）我国 2020 年 1-7 月疫情确诊人数累计确诊人数变动情况：

该项数据分析通过柱形图和折线图来观测我国 2020 年 1 月至 7 月疫情变动月确诊人数和确诊人数变动率来大致地分析全国疫情传播的纵向特点。由图 4.1 和图 4.2 可知：

①1-3 月：自 12 月底疫情爆发以来，新冠肺炎在开始的 3 个月里迅速传播，无论是确诊人数，还是死亡人数，还是疑似患者都以较快的速率迅速上升，且在 1 月和 2 月，治愈人数较少，治愈人数远远赶不上确诊人数。

由图表推测可知，产生该现象的原因有：一是由于该阶段恰逢春节到来，各地之间人口流动频繁，人群聚集性感染急剧上升；二是人们在爆发初期对该种传染病缺乏科学认识，未能引起足够重视；三是一些地方政府治理防控体系尚未成熟，反应迟缓；四是在新冠肺炎爆发初期，医护工作人员对这种新型病毒的认识不够，缺乏特效疗法；且在爆发初期，医院设施、资源相对匮乏（呼吸机、床位等），均未能跟得上确诊人数的急剧上升。这都导致了 1-3 月我国疫情的急剧恶化。

②4-5 月：该阶段是新增确诊人数、新增疑似患者数、新增死亡人数快速下降，治愈人数平稳上升的阶段，由图可知，该阶段的疫情已经得到有效控制。这主要是因为：

一是各地政府防控体系日趋成熟健全，防控措施得当；二是居

民科学认识了新冠病毒，认真配合社区居家隔离工作，减少了外出和聚集。三是我国医疗设施产能大幅度提升，医疗物资供给上升。因此，疫情的增速大大放缓。疫情的高峰已经结束。

③6-7 月，该阶段疫情以较低速率平稳变动，新增确诊人员变为主要以境外输入为主，各地复工复产、国家高考均恢复正常。疫情防控工作又进入了一个新常态。

（2）我国湖北省 2020 年 1 月至 7 月疫情数据概览：

由图 4.3 可知,湖北省疫情变动趋势大体上与全国疫情变动趋势相同，新增确诊人数在 2 月底已经转为低速率增长，累计死亡人数和累计治愈人数也在 3、4 月份达到峰值，增长态势呈现水平运动的趋势。这表明湖北省疫情防控工作成效显著，也为全国的疫情形势的扭转做出了重要贡献。湖北已经由原来的疫情爆发地区转为全国最安全地区。

（3）我国不同省份和地区疫情情况的比较：

由图 4.4 和 4.5 可知，我国疫情确诊人员在人口稠密，经济发达且临近东部沿海、中俄边界等省份的分布较多；而在西北、西南等地广人稀、经济欠发达的内陆省份，则感染人数相对较少。图表呈现出的另外两个特点是南方地区疫情确诊人数多于北方地区;东部地区的疫情感染人数占据了总感染人数一半以上，而中部、西部占比仅仅 33.3%和 13.0%。这些现象侧面反映了疫情与经济发展水平、人口稠密大致成正相关关系：经济发展水平越高，人口稠密度越大，疫情累计确诊人数就越多，比如广东省。相反地，经济发展水平越低，交通

通达度越小，疫情累计确诊人数就越少，比如西藏自治区。疫情分布还与地理分布有关，东部沿海的发达地区如北京市、上海市、绥芬河等地境外输入病例较内地而言更多。以上就是通过横向对比反映出的疫情分布特点。

（4）我国不同省份治愈率差异（除湖北省外）：

防止极端值对数据分析的影响，该项数据分析去除了湖北省、西藏自治区、宁夏回族自治区和青海省等省份。该项数据分析已治愈率折线图和各省累计确诊人数与累计治愈人数的比较柱形图来分析治愈情况的差异，由图 4.6 和图 4.7 可知，治愈率较高的省份是江苏省、浙江省、甘肃省，而治愈率较低的地区是北京市、辽宁市和上海市等地。关于为何治愈率在经济不发达地区较高，在经济发达地区反而较低的原因，国家卫健委尚未给出客观的解释，本次数据分析也难以反映背后的原因。

6 总结

本次数据分析主要从我国 2020 年 1-7 月疫情确诊人数累计确诊人数变动情况；我国湖北省 2020 年 1 月至 7 月疫情数据概览；我国不同省份治愈率差异（除湖北省外）；我国不同省份和地区疫情情况的比较等 4 个方面简要探讨了发掘了 2020 年全年疫情传播的纵向特点和横向特点。借此简要分析了图表反映的现象的背后的深层次原因，简要地总结出了新冠肺炎在我国传播的大致规律。

通过本次分析，我也深刻体会到了数据分析和数据可视化在疫情防控上的不可小觑的力量和 Pandas 工具在处理 Dataframe 数据的强

大功能；也进一步熟练了 Python 开发环境的 Pandas 工具和 Matplotlib 数据可视化工具的运用。同时我在编程也意识到了自己在运用 Pandas 工具和 Matplotlib 数据可视化工具时的不足与局限：代码编写过于冗长；数据分析项目仍然有限；在数据分析初期，未能区分“新增确诊数据”和“累计确诊数据”，导致初期数据分析错误。但是，这未尝也是一种收获，能让我在发现问题之后，能够在日后的数据分析这条路上更加行稳致远。

同时，在数据分析过程中。我深深深意识到了对我们这样一个拥有 14 亿人口的发展中国家来说，能在较短时间内有效控制疫情，保障了人民基本生活，在新冠疫情防控上的取得重大战略成果，可以说是十分不易、成之惟艰。我国在新冠疫情防控上的取得重大战略成果，也体现了中国特色社会主义的强大韧性和生命力。这得益于以习近平同志为核心的党中央坚强领导；得益于习近平新时代中国特色社会主义思想科学指引；得益于我国的制度优势；得益于全党全军全国各族人民的团结奋斗。

附录-数据分析代码

```
import pandas as pd
data = pd.read_csv('DXYArea.csv', encoding='UTF-8')
data['updateTime']=pd.to_datetime(data['updateTime'])
#取出中国的数据
ChinaData=data[data['countryName']=='中国']
#剔除空值行
ChinaData1=ChinaData.dropna()
#去除无用的列，无用的行
ChinaData1.drop(['continentEnglishName'],axis=1,inplace=True)
ChinaData1.drop(['provinceEnglishName'],axis=1,inplace=True)
ChinaData1.drop(['province_zipCode'],axis=1,inplace=True)
ChinaData1.drop(['cityEnglishName'],axis=1,inplace=True)
```

```
ChinaData1.drop(['city_zipCode'],axis=1,inplace=True)
print(ChinaData1.info())
print('各字段是否含有空值情况: \n',ChinaData1.isna().any())
#单独分离出月份和具体日期
ChinaData1['update_month']=ChinaData['updateTime'].dt.month
ChinaData1['update_year']=ChinaData['updateTime'].dt.date
#逐月确诊人数增长率分析
ChinaData1_group1=ChinaData1.groupby('update_month')
a=ChinaData1_group1.mean().sort_index()
print(a)
#confirmedCount_1=sales_year[2012]/sales_year[2011]-1
rate_2=a.iloc[1,0]/a.iloc[0,0]-1
rate_3=a.iloc[2,0]/a.iloc[1,0]-1
rate_4=a.iloc[3,0]/a.iloc[2,0]-1
rate_5=a.iloc[4,0]/a.iloc[3,0]-1
rate_6=a.iloc[5,0]/a.iloc[4,0]-1
rate_7=a.iloc[6,0]/a.iloc[5,0]-1
#
print(rate_2,rate_3,rate_4,rate_5,rate_6,rate_7)
all_rate=pd.DataFrame({'confirmedCount':a['province_confirmedCount'],
\
'rate':[0,rate_2,rate_3,rate_4,rate_5,rate_6,rate_7]})
print(all_rate)
import matplotlib.pyplot as plt
import matplotlib as mpl
mpl.rcParams['font.sans-serif'] = 'SimHei'
mpl.rcParams['axes.unicode_minus'] = False
plt.style.use('ggplot')

y1=all_rate['confirmedCount'] #确诊人数列
y2=all_rate['rate'] #增长率列
x=[str(value) for value in all_rate.index.tolist()] #得到年份字符串列
表
#新建 figure 对象
fig=plt.figure()
#新建子图 1
ax1=fig.add_subplot(1,1,1)
#ax2 与 ax1 共享 x 轴
ax2=ax1.twinx()
ax1.bar(x,y1,color='blue')
ax2.plot(x,y2,marker='*',color='r')
ax1.set_xlabel('月份/月')
ax1.set_ylabel('确诊人数/人')
ax2.set_ylabel('变化率')
```

```
ax1.set_title('2020 年 1 月——7 月全国确诊人数与确诊人数变化率')
plt.show()
#各省确诊治愈死亡人数变动图;
b=a[['province_confirmedCount','province_curedCount','province_deadCo
unt']]
b.plot(title='全国确诊、治愈、疑似和死亡人数逐月变动图')
#我国不同省份确诊治愈疑似死亡人数
ChinaData1_group2=ChinaData1[['provinceName','province_confirmedCount
','province_suspectedCount','province_curedCount','province_deadCount
']].groupby('provinceName')
c= ChinaData1_group2.describe()
d=
c[[('province_confirmedCount','max'),('province_suspectedCount','max'
),('province_deadCount','max'),('province_curedCount','max')]]
d=d.drop(index=['湖北省'])
d=d.sort_values(by=[('province_confirmedCount','max'),('province_susp
ectedCount','max'),('province_deadCount','max'),('province_curedCount
','max')],ascending=False)
d.plot(kind='bar',title= '我国不同省份确诊治愈疑似死亡人数
',rot=90,figsize=(8,8))
#我国不同地区确诊人数分布饼图:
all=d[('province_confirmedCount','max')].sum(axis=0)
dongbu_all= d[(d.index=='广东省')|(d.index=='福建省')|(d.index=='海南
省')|(d.index=='浙江省')|(d.index=='江苏省')
              |(d.index=='山东省')|(d.index=='北京市')|(d.index=='天
津市')|(d.index=='河北省')
              |(d.index=='辽宁省')|(d.index=='上海市')|(d.index=='黑
龙江省')][('province_confirmedCount','max')].sum(axis=0)
dongbu_rate1=dongbu_all/all
#去除湖北省
zhongbu_all=d[(d.index=='湖南省')|(d.index=='安徽省')|(d.index=='山西
省')|(d.index=='河南省')
              |(d.index=='吉林省')|(d.index=='重庆市')|(d.index=='江
西省')][('province_confirmedCount','max')].sum(axis=0)
zhongbu_rate2=zhongbu_all/all
xibu_all=d[(d.index=='甘肃省')|(d.index=='新疆维吾尔族自治区
')|(d.index=='西藏自治区')|(d.index=='青海省')|(d.index=='宁夏回族自
治区')
              |(d.index=='陕西省')|(d.index=='四川省')|(d.index=='云
南省')|(d.index=='贵州省')
              |(d.index=='广西壮族自治区')|(d.index=='内蒙古自治区
')][('province_confirmedCount','max')].sum(axis=0)
xibu_rate3=xibu_all/all
fig=plt.figure()
```

```
rate_=[dongbu_rate1,zhongbu_rate2,xibu_rate3]
labels=['东部沿海省份','中部省份(除湖北省外)','西部省份']
explode=(0.05,0.05,0.05)
plt.pie(rate_,explode=explode,labels=labels,autopct='%.1f%%',shadow=F
alse,startangle=90)
plt.title('我国不同地区确诊人数分布饼图')
plt.show()
#'湖北省累计确诊、新增疑似病例、累计治愈病例、累计死亡病例统计图'
Hubeidata_group=ChinaData1[ChinaData1['provinceName']=='湖北省
'][['provinceName','province_confirmedCount'
                    ,'province_suspectedCount','province_curedCount',
'province_deadCount','update_month']].groupby('update_month')
Hubeidata=Hubeidata_group.describe()[[('province_confirmedCount','max
')
                                        ,('province_suspectedCount','ma
x'),('province_deadCount','max'),('province_curedCount','max')]]
Hubeidata.plot(title='湖北省累计确诊、新增疑似病例、累计治愈病例、累
计死亡病例统计图')

#各省治愈人数比较:
d=c[[('province_confirmedCount','max'),('province_curedCount','max')]
]
d=d.drop(index=['湖北省'])
d=d.drop(index=['青海省'])
d=d.drop(index=['西藏自治区'])
d=d.drop(index=['宁夏回族自治区'])
f= lambda x:x[1]/x[0]
d['治愈率']=d.apply(f,axis=1)
d=d.sort_values(by=[('治愈率')],ascending=False)
d.plot(kind='barh',fontsize=8)
fig=plt.figure()
d['治愈率'].plot(kind='bar',color='b',title='各省治愈率比较')
```