In [1]:

```
import requests
import pandas as pd
from lxml import etree

html = 'https://ncov.dxy.cn/ncovh5/view/pneumonia'
html_data = requests.get(html)
html_data.encoding = 'utf-8'
html_data = etree.HTML(html_data.text, etree.HTMLParser())
html_data = html_data.xpath(
    '//*[@id="getListByCountryTypeService2true"]/text()')
ncov_world = html_data[0][49:-12]
ncov_world = ncov_world.replace('true', 'True')
ncov_world = ncov_world.replace('false', 'False')
ncov_world = eval(ncov_world)

country = []
confirmed = []
lived = []
dead = []

for i in ncov_world:
    country.append(i['provinceName'])
    confirmed.append(i['confirmedCount'])
    lived.append(i['curedCount'])
    dead.append(i['deadCount'])

data_world = pd.DataFrame()
data_world['国家名称'] = country
data_world['确诊人数'] = confirmed
data_world['治愈人数'] = lived
data_world['死亡人数'] = dead
data_world.head(5)
```

Out[1]:

	国家名称	确诊人数	治愈人数	死亡人数
0	法国	29583616	368023	149044
1	德国	26408657	4328400	139218
2	韩国	18129313	336548	24212
3	英国	22455392	6491069	178880
4	西班牙	12360256	150376	106493

In [2]:

```
import pandas as pd
data_world = pd.read_csv('https://labfile.oss.aliyuncs.com/courses/2791/data_world.csv')
data_world.head(5)
```

Out[2]:

	国家名称	确诊人数	治愈人数	死亡人数
0	法国	27626578	368023	144130
1	德国	23376879	4328400	132929
2	韩国	16212751	336548	20889
3	英国	21819851	6491069	171560
4	西班牙	11662214	150376	103266

In [3]:

```
data_economy = pd.read_csv(
    "https://labfile.oss.aliyuncs.com/courses/2791/gpd_2016_2020.csv", index_col=0)
time_index = pd.date_range(start='2016', periods=18, freq='Q')
data_economy.index = time_index
data_economy
```

Out[3]:

	国内生产总值	第一产业增加值	第二产业增加值	第三产业增加值	农林牧渔业增加值	工业增加值	制造业增加值	建筑业增加值	批发和零售业增加值	交运输、储和政
2016-03-31	162410.0	8312.7	61106.8	92990.5	8665.5	53666.4	45784.0	7763.0	16847.5	718
2016-06-30	181408.2	12555.9	73416.5	95435.8	13045.5	60839.2	52378.3	12943.8	17679.8	829
2016-09-30	191010.6	17542.4	75400.5	98067.8	18162.2	61902.5	52468.3	13870.6	18513.0	859
2016-12-31	211566.2	21728.2	85504.1	104334.0	22577.8	68998.4	58878.4	16921.5	20684.1	896
2017-03-31	181867.7	8205.9	69315.5	104346.3	8595.8	60909.3	51419.7	8725.3	18608.9	809
2017-06-30	201950.3	12644.9	82323.0	106982.4	13204.2	68099.8	58172.1	14574.4	19473.6	939
2017-09-30	212789.3	18255.8	84574.1	109959.5	18944.2	69327.2	58632.6	15590.1	20342.9	968
2017-12-31	235428.7	22992.9	95368.0	117067.8	23915.8	76782.9	65652.1	19015.8	22731.1	994
2018-03-31	202035.7	8575.7	76598.2	116861.8	9005.8	66905.6	56631.9	10073.8	20485.5	880
2018-06-30	223962.2	13003.8	91100.6	119857.8	13662.2	75122.1	64294.9	16404.3	21374.2	1017
2018-09-30	234474.3	18226.9	93112.5	123134.9	18961.8	76239.6	64348.2	17294.5	22334.1	1058
2018-12-31	258808.9	24938.7	104023.9	129846.2	25929.0	82822.1	70662.1	21720.4	24710.0	1077
2019-03-31	218062.8	8769.4	81806.5	127486.9	9249.4	71064.5	60357.1	11143.1	21959.2	938
2019-06-30	242573.8	14437.6	97315.6	130820.6	15108.7	79820.7	68041.8	17954.2	23097.0	1086
2019-09-30	252208.7	19798.0	97790.4	134620.4	20629.0	79501.8	66823.8	18734.6	23993.6	1131
2019-12-31	278019.7	27461.6	109252.8	141305.2	28579.9	86721.6	73952.4	23072.4	26795.9	1124
2020-03-31	206504.3	10186.2	73638.0	122680.1	10708.4	64642.0	53852.0	9377.8	18749.6	786
2020-06-30	250110.1	15866.8	99120.9	135122.3	16596.4	80402.4	69258.8	19156.8	23696.1	1065

In [4]:

```
data_area = pd.read_csv('https://labfile.oss.aliyuncs.com/courses/2791/DXYArea.csv')
data_news = pd.read_csv('https://labfile.oss.aliyuncs.com/courses/2791/DXYNews.csv')
```

In [5]:

```
data_area = data_area.loc[data_area['countryName'] == data_area['provinceName']]
data_area_times = data_area[['countryName', 'province_confirmedCount',
                             'province_curedCount', 'province_deadCount', 'updateTime']]

time = pd.DatetimeIndex(data_area_times['updateTime'])
data_area_times.index = time
data_area_times = data_area_times.drop('updateTime', axis=1)
data_area_times.head(5)

data_area_times.isnull().any()
```

Out[5]:

```
countryName               False
province_confirmedCount   False
province_curedCount       False
province_deadCount        False
dtype: bool
```

In [6]:

```
data_news_times = data_news[['pubDate', 'title', 'summary']]
time = pd.DatetimeIndex(data_news_times['pubDate'])
data_news_times.index = time
data_news_times = data_news_times.drop('pubDate', axis=1)
data_news_times.head(5)
```

Out[6]:

pubDate	title	summary
2020-07-17 05:40:08	美国新增71434例新冠肺炎确诊病例，累计确诊超354万例	据美国约翰斯·霍普金斯大学统计数据显示，截至美东时间7月16日17:33时（北京时间17日0...
2020-07-17 06:06:49	巴西新冠肺炎确诊病例破201万，近六成大城市确诊病例加速增长	截至当地时间7月16日18时，巴西新增新冠肺炎确诊病例45403例，累计确诊2012151例...
2020-07-16 22:31:00	阿塞拜疆新增493例新冠肺炎确诊病例 累计确诊26165例	当地时间7月16日，阿塞拜疆国家疫情防控指挥部发布消息，在过去24小时内，阿塞拜疆新增新冠肺...
2020-07-16 22:29:48	科威特新增791例新冠肺炎确诊病例 累计确诊57668例	科威特卫生部当地时间16日下午发布通告，确认过去24小时境内新增791例新冠肺炎确诊病例，同...
2020-07-16 21:26:54	罗马尼亚新增777例新冠肺炎确诊病例 累计确诊35003例	据罗马尼亚政府7月16日公布的数据，过去24小时对19097人进行新冠病毒检测，确诊777例...

In [7]:

```
print(data_world.isnull().any())
print(data_economy.isnull().any())
print(data_area_times.isnull().any())
print(data_news_times.isnull().any())
```

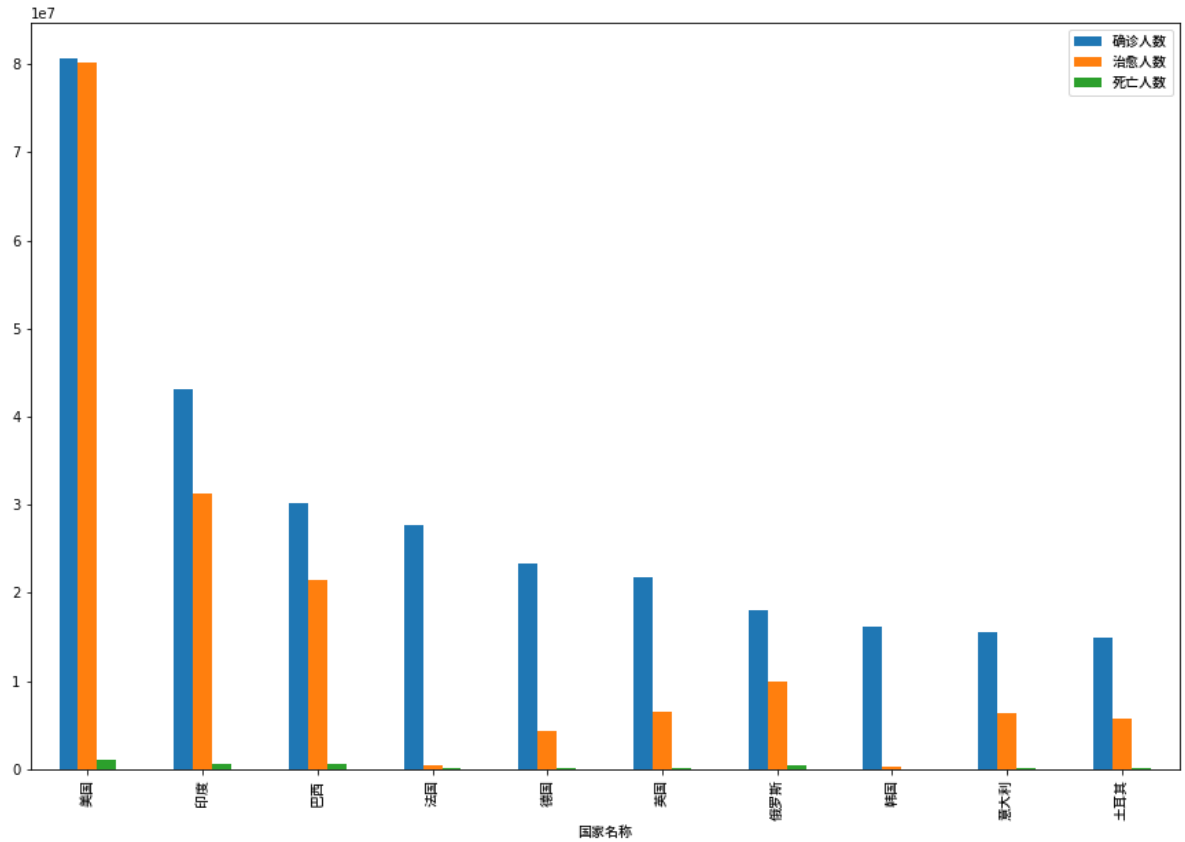
```
国家名称    False
确诊人数    False
治愈人数    False
死亡人数    False
dtype: bool
国内生产总值                 False
第一产业增加值                False
第二产业增加值                False
第三产业增加值                False
农林牧渔业增加值               False
工业增加值                  False
制造业增加值                 False
建筑业增加值                 False
批发和零售业增加值              False
交通运输、仓储和邮政业增加值         False
住宿和餐饮业增加值              False
金融业增加值                 False
房地产业增加值                False
信息传输、软件和信息技术服务业增加值    False
租赁和商务服务业增加值             False
其他行业增加值                False
dtype: bool
countryName               False
province_confirmedCount   False
province_curedCount       False
province_deadCount        False
dtype: bool
title      False
summary    False
dtype: bool
```

In [8]:

```
import matplotlib.pyplot as plt
import matplotlib
import os

%matplotlib inline

fpath = os.path.join(r"D:\数分\NotoSansCJK.otf")
myfont = matplotlib.font_manager.FontProperties(fname=fpath)
# 绘图
data_world = data_world.sort_values(by='确诊人数', ascending=False)
data_world_set = data_world[['确诊人数', '治愈人数', '死亡人数']]
data_world_set.index = data_world['国家名称']
data_world_set.head(10).plot(kind='bar', figsize=(15, 10))
plt.xlabel('国家名称', fontproperties=myfont)
plt.xticks(fontproperties=myfont)
plt.legend(fontsize=30, prop=myfont)
```

Out[8]:

<matplotlib.legend.Legend at 0x1cdf2ff88b0>



In [9]:

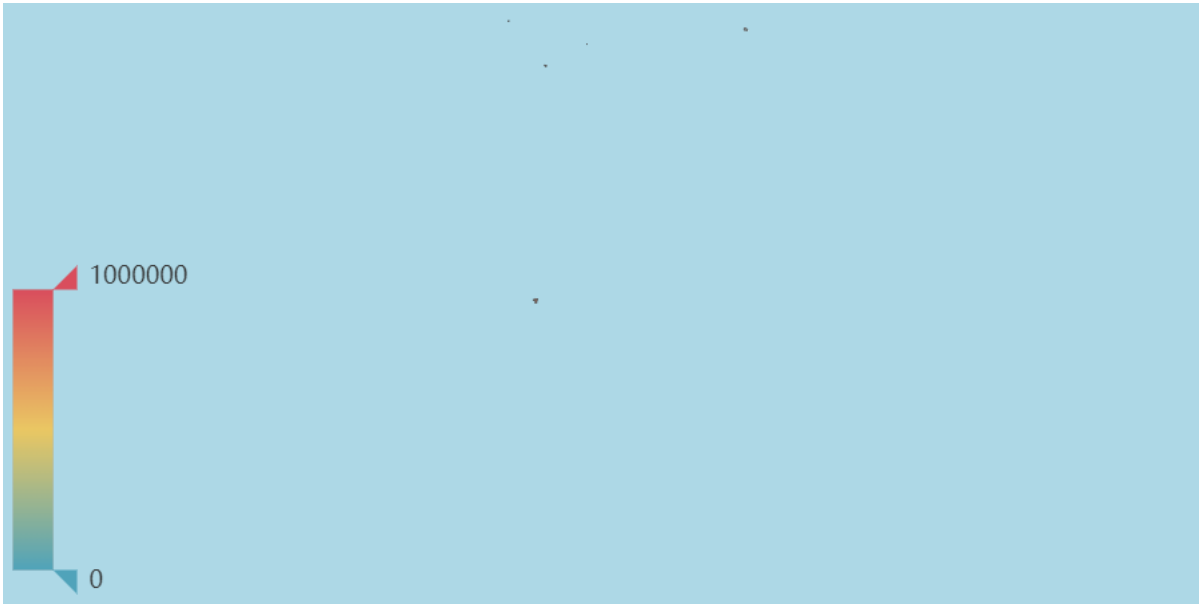
```
from pyecharts.charts import Map
from pyecharts import options as opts
from pyecharts.globals import CurrentConfig, NotebookType

CurrentConfig.NOTEBOOK_TYPE = NotebookType.JUPYTER_NOTEBOOK
name_map = {
    'Singapore Rep.': '新加坡',
    'Dominican Rep.': '多米尼加',
    'Palestine': '巴勒斯坦',
    'Bahamas': '巴哈马',
    'Timor-Leste': '东帝汶',
    'Afghanistan': '阿富汗',
    'Guinea-Bissau': '几内亚比绍',
    "Côte d'Ivoire": '科特迪瓦',
    'Siachen Glacier': '锡亚琴冰川',
    "Br. Indian Ocean Ter.": '英属印度洋领土',
    'Angola': '安哥拉',
    'Albania': '阿尔巴尼亚',
    'United Arab Emirates': '阿联酋',
    'Argentina': '阿根廷',
    'Armenia': '亚美尼亚',
    'French Southern and Antarctic Lands': '法属南半球和南极领地',
    'Australia': '澳大利亚',
    'Austria': '奥地利',
    'Azerbaijan': '阿塞拜疆',
    'Burundi': '布隆迪',
    'Belgium': '比利时',
    'Benin': '贝宁',
    'Burkina Faso': '布基纳法索',
    'Bangladesh': '孟加拉国',
    'Bulgaria': '保加利亚',
    'The Bahamas': '巴哈马',
    'Bosnia and Herz.': '波斯尼亚和黑塞哥维那',
    'Belarus': '白俄罗斯',
    'Belize': '伯利兹',
    'Bermuda': '百慕大',
    'Bolivia': '玻利维亚',
    'Brazil': '巴西',
    'Brunei': '文莱',
    'Bhutan': '不丹',
    'Botswana': '博茨瓦纳',
    'Central African Rep.': '中非',
    'Canada': '加拿大',
    'Switzerland': '瑞士',
    'Chile': '智利',
    'China': '中国',
    'Ivory Coast': '象牙海岸',
    'Cameroon': '喀麦隆',
    'Dem. Rep. Congo': '刚果民主共和国',
    'Congo': '刚果',
    'Colombia': '哥伦比亚',
    'Costa Rica': '哥斯达黎加',
    'Cuba': '古巴',
    'N. Cyprus': '北塞浦路斯',
    'Cyprus': '塞浦路斯',
    'Czech Rep.': '捷克',
    'Germany': '德国',
    'Djibouti': '吉布提',
    'Denmark': '丹麦',
```

```
    'Algeria': '阿尔及利亚',
    'Ecuador': '厄瓜多尔',
    'Egypt': '埃及',
    'Eritrea': '厄立特里亚',
    'Spain': '西班牙',
    'Estonia': '爱沙尼亚',
    'Ethiopia': '埃塞俄比亚',
    'Finland': '芬兰',
    'Fiji': '斐',
    'Falkland Islands': '福克兰群岛',
    'France': '法国',
    'Gabon': '加蓬',
    'United Kingdom': '英国',
    'Georgia': '格鲁吉亚',
    'Ghana': '加纳',
    'Guinea': '几内亚',
    'Gambia': '冈比亚',
    'Guinea Bissau': '几内亚比绍',
    'Eq. Guinea': '赤道几内亚',
    'Greece': '希腊',
    'Greenland': '格陵兰',
    'Guatemala': '危地马拉',
    'French Guiana': '法属圭亚那',
    'Guyana': '圭亚那',
    'Honduras': '洪都拉斯',
    'Croatia': '克罗地亚',
    'Haiti': '海地',
    'Hungary': '匈牙利',
    'Indonesia': '印度尼西亚',
    'India': '印度',
    'Ireland': '爱尔兰',
    'Iran': '伊朗',
    'Iraq': '伊拉克',
    'Iceland': '冰岛',
    'Israel': '以色列',
    'Italy': '意大利',
    'Jamaica': '牙买加',
    'Jordan': '约旦',
    'Japan': '日本',
    'Kazakhstan': '哈萨克斯坦',
    'Kenya': '肯尼亚',
    'Kyrgyzstan': '吉尔吉斯斯坦',
    'Cambodia': '柬埔寨',
    'Korea': '韩国',
    'Kosovo': '科索沃',
    'Kuwait': '科威特',
    'Lao PDR': '老挝',
    'Lebanon': '黎巴嫩',
    'Liberia': '利比里亚',
    'Libya': '利比亚',
    'Sri Lanka': '斯里兰卡',
    'Lesotho': '莱索托',
    'Lithuania': '立陶宛',
    'Luxembourg': '卢森堡',
    'Latvia': '拉脱维亚',
    'Morocco': '摩洛哥',
    'Moldova': '摩尔多瓦',
    'Madagascar': '马达加斯加',
    'Mexico': '墨西哥',
    'Macedonia': '马其顿',
    'Mali': '马里',
```

```
    'Myanmar': '缅甸',
    'Montenegro': '黑山',
    'Mongolia': '蒙古',
    'Mozambique': '莫桑比克',
    'Mauritania': '毛里塔尼亚',
    'Malawi': '马拉维',
    'Malaysia': '马来西亚',
    'Namibia': '纳米比亚',
    'New Caledonia': '新喀里多尼亚',
    'Niger': '尼日尔',
    'Nigeria': '尼日利亚',
    'Nicaragua': '尼加拉瓜',
    'Netherlands': '荷兰',
    'Norway': '挪威',
    'Nepal': '尼泊尔',
    'New Zealand': '新西兰',
    'Oman': '阿曼',
    'Pakistan': '巴基斯坦',
    'Panama': '巴拿马',
    'Peru': '秘鲁',
    'Philippines': '菲律宾',
    'Papua New Guinea': '巴布亚新几内亚',
    'Poland': '波兰',
    'Puerto Rico': '波多黎各',
    'Dem. Rep. Korea': '朝鲜',
    'Portugal': '葡萄牙',
    'Paraguay': '巴拉圭',
    'Qatar': '卡塔尔',
    'Romania': '罗马尼亚',
    'Russia': '俄罗斯',
    'Rwanda': '卢旺达',
    'W. Sahara': '西撒哈拉',
    'Saudi Arabia': '沙特阿拉伯',
    'Sudan': '苏丹',
    'S. Sudan': '南苏丹',
    'Senegal': '塞内加尔',
    'Solomon Is.': '所罗门群岛',
    'Sierra Leone': '塞拉利昂',
    'El Salvador': '萨尔瓦多',
    'Somaliland': '索马里兰',
    'Somalia': '索马里',
    'Serbia': '塞尔维亚',
    'Suriname': '苏里南',
    'Slovakia': '斯洛伐克',
    'Slovenia': '斯洛文尼亚',
    'Sweden': '瑞典',
    'Swaziland': '斯威士兰',
    'Syria': '叙利亚',
    'Chad': '乍得',
    'Togo': '多哥',
    'Thailand': '泰国',
    'Tajikistan': '塔吉克斯坦',
    'Turkmenistan': '土库曼斯坦',
    'East Timor': '东帝汶',
    'Trinidad and Tobago': '特里尼达和多巴哥',
    'Tunisia': '突尼斯',
    'Turkey': '土耳其',
    'Tanzania': '坦桑尼亚',
    'Uganda': '乌干达',
    'Ukraine': '乌克兰',
    'Uruguay': '乌拉圭',
```

```
    'United States': '美国',
    'Uzbekistan': '乌兹别克斯坦',
    'Venezuela': '委内瑞拉',
    'Vietnam': '越南',
    'Vanuatu': '瓦努阿图',
    'West Bank': '西岸',
    'Yemen': '也门',
    'South Africa': '南非',
    'Zambia': '赞比亚',
    'Zimbabwe': '津巴布韦',
    'Comoros': '科摩罗'
}

map = Map(init_opts=opts.InitOpts(width="1900px", height="900px",
                                  bg_color="#ADD8E6", page_title="全球疫情确诊人数"))
map.add("确诊人数", [list(z) for z in zip(data_world['国家名称'], data_world['确诊人数'])],
        is_map_symbol_show=False,
        maptype="world", label_opts=opts.LabelOpts(is_show=False), name_map=name_map,
        itemstyle_opts=opts.ItemStyleOpts(color="rgb(49,60,72)"),
        ).set_global_opts(
    visualmap_opts=opts.VisualMapOpts(max_=1000000),
)
map.render_notebook()
```

Out[9]:



In [10]:

```
country = data_area_times.sort_values('province_confirmedCount', ascending=False).drop_duplicates(
    subset='countryName', keep='first').head(6)['countryName']
country = list(country)
country
```

Out[10]:

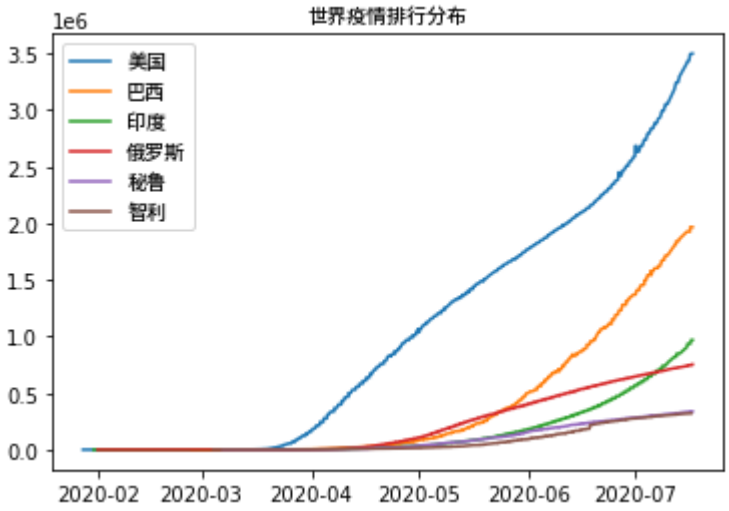
['美国', '巴西', '印度', '俄罗斯', '秘鲁', '智利']

In [11]:

```
data_America = data_area_times[data_area_times['countryName'] == '美国']
data_Brazil = data_area_times[data_area_times['countryName'] == '巴西']
data_India = data_area_times[data_area_times['countryName'] == '印度']
data_Russia = data_area_times[data_area_times['countryName'] == '俄罗斯']
data_Peru = data_area_times[data_area_times['countryName'] == '秘鲁']
data_Chile = data_area_times[data_area_times['countryName'] == '智利']

timeindex = data_area_times.index
timeindex = timeindex.floor('D')
data_area_times.index = timeindex

timeseries = pd.DataFrame(data_America.index)
timeseries.index = data_America.index
data_America = pd.concat([timeseries, data_America], axis=1)
data_America.drop_duplicates(
    subset='updateTime', keep='first', inplace=True)
data_America.drop('updateTime', axis=1, inplace=True)

timeseries = pd.DataFrame(data_Brazil.index)
timeseries.index = data_Brazil.index
data_Brazil = pd.concat([timeseries, data_Brazil], axis=1)

data_Brazil.drop_duplicates(subset='updateTime', keep='first', inplace=True)
data_Brazil.drop('updateTime', axis=1, inplace=True)

timeseries = pd.DataFrame(data_India.index)
timeseries.index = data_India.index
data_India = pd.concat([timeseries, data_India], axis=1)
data_India.drop_duplicates(subset='updateTime', keep='first', inplace=True)
data_India.drop('updateTime', axis=1, inplace=True)

timeseries = pd.DataFrame(data_Russia.index)
timeseries.index = data_Russia.index
data_Russia = pd.concat([timeseries, data_Russia], axis=1)
data_Russia.drop_duplicates(subset='updateTime', keep='first', inplace=True)
data_Russia.drop('updateTime', axis=1, inplace=True)

timeseries = pd.DataFrame(data_Peru.index)
timeseries.index = data_Peru.index
data_Peru = pd.concat([timeseries, data_Peru], axis=1)
data_Peru.drop_duplicates(subset='updateTime', keep='first', inplace=True)
data_Peru.drop('updateTime', axis=1, inplace=True)

timeseries = pd.DataFrame(data_Chile.index)
timeseries.index = data_Chile.index
data_Chile = pd.concat([timeseries, data_Chile], axis=1)
data_Chile.drop_duplicates(subset='updateTime', keep='first', inplace=True)
data_Chile.drop('updateTime', axis=1, inplace=True)

plt.title("世界疫情排行分布", fontproperties=myfont)
plt.plot(data_America['province_confirmedCount'])
plt.plot(data_Brazil['province_confirmedCount'])
plt.plot(data_India['province_confirmedCount'])
plt.plot(data_Russia['province_confirmedCount'])
plt.plot(data_Peru['province_confirmedCount'])
plt.plot(data_Chile['province_confirmedCount'])
plt.legend(country, prop=myfont)
```

Out[11]:

<matplotlib.legend.Legend at 0x1cdf9067f40>



In [12]:

```
!pip install wordcloud==1.8.0
```

```
Requirement already satisfied: wordcloud==1.8.0 in c:\programdata\anaconda3\lib\site
-packages (1.8.0)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib\site-packa
ges (from wordcloud==1.8.0) (3.3.4)
Requirement already satisfied: numpy>=1.6.1 in c:\programdata\anaconda3\lib\site-pac
kages (from wordcloud==1.8.0) (1.20.1)
Requirement already satisfied: pillow in c:\programdata\anaconda3\lib\site-packages
(from wordcloud==1.8.0) (8.2.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\lib\sit
e-packages (from matplotlib->wordcloud==1.8.0) (1.3.1)
Requirement already satisfied: python-dateutil>=2.1 in c:\programdata\anaconda3\lib
\site-packages (from matplotlib->wordcloud==1.8.0) (2.8.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\progra
mdata\anaconda3\lib\site-packages (from matplotlib->wordcloud==1.8.0) (2.4.7)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-pac
kages (from matplotlib->wordcloud==1.8.0) (0.10.0)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-packages (fr
om cycler>=0.10->matplotlib->wordcloud==1.8.0) (1.15.0)
```

In [1]:

```
pip install -i https://pypi.tuna.tsinghua.edu.cn/simple jieba
```

```
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple (https://pypi.tuna.tsinghua.edu.cn/simple)
Collecting jieba
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/c6/cb/18eeb235f833b726522d7ebed54f2278ce28ba9438e3135ab0278d9792a2/jieba-0.42.1.tar.gz (https://pypi.tuna.tsinghua.edu.cn/packages/c6/cb/18eeb235f833b726522d7ebed54f2278ce28ba9438e3135ab0278d9792a2/jieba-0.42.1.tar.gz) (19.2 MB)
Building wheels for collected packages: jieba
  Building wheel for jieba (setup.py): started
  Building wheel for jieba (setup.py): finished with status 'done'
  Created wheel for jieba: filename=jieba-0.42.1-py3-none-any.whl size=19314477 sha256=4187329243b36fb95844f9d05ede61938b2f2f0a96502e8b2907423d7d4329a9
  Stored in directory: c:\users\86150\appdata\local\pip\cache\wheels\f3\30\86\64b88bf0241f0132806c61b1e2686b44f1327bfc5642f9d77d
Successfully built jieba
Installing collected packages: jieba
Successfully installed jieba-0.42.1
Note: you may need to restart the kernel to use updated packages.
```

In [13]:

```
import jieba
import re
from wordcloud import WordCloud

def word_cut(x): return jieba.lcut(x)

news = []
reg = "[^\u4e00-\u9fa5]"
for i in data_news['title']:
    if re.sub(reg, '', i) != '':
        news.append(re.sub(reg, '', i))

words = []
counts = {}
for i in news:
    words.append(word_cut(i))
for word in words:
    for a_word in word:
        if len(a_word) == 1:
            continue
        else:
            counts[a_word] = counts.get(a_word, 0)+1
words_sort = list(counts.items())
words_sort.sort(key=lambda x: x[1], reverse=True)

newcloud = WordCloud(font_path=r"D:\数分\NotoSansCJK.otf",
                     background_color="white", width=600, height=300, max_words=50)
newcloud.generate_from_frequencies(counts)
image = newcloud.to_image()
image
```

```
Building prefix dict from the default dictionary ...
Loading model from cache C:\Users\86150\AppData\Local\Temp\jieba.cache
Loading model cost 0.651 seconds.
Prefix dict has been built successfully.
```

Out[13]:

In [15]:

```
pip install gensim==3.8.1
```

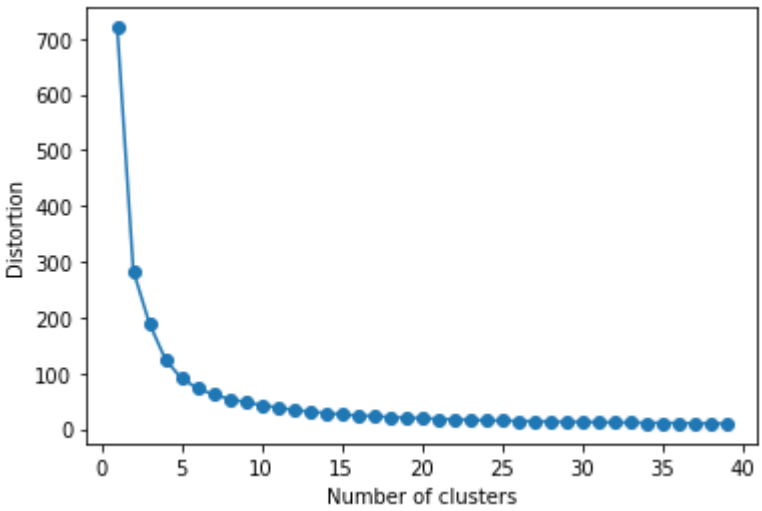
```
Collecting gensim==3.8.1
  Downloading gensim-3.8.1.tar.gz (23.4 MB)
Requirement already satisfied: numpy>=1.11.3 in c:\programdata\anaconda3\lib\site-pa
ckages (from gensim==3.8.1) (1.20.1)
Requirement already satisfied: scipy>=0.18.1 in c:\programdata\anaconda3\lib\site-pa
ckages (from gensim==3.8.1) (1.6.2)
Requirement already satisfied: six>=1.5.0 in c:\programdata\anaconda3\lib\site-packa
ges (from gensim==3.8.1) (1.15.0)
Requirement already satisfied: smart_open>=1.8.1 in c:\programdata\anaconda3\lib\sit
e-packages (from gensim==3.8.1) (6.0.0)
Building wheels for collected packages: gensim
  Building wheel for gensim (setup.py): started
  Building wheel for gensim (setup.py): finished with status 'done'
  Created wheel for gensim: filename=gensim-3.8.1-cp38-cp38-win_amd64.whl size=23720
033 sha256=334a6cb4a287f635ab075a500f055c22a976c5d0c5e84ba430ec8a55ea059d44
  Stored in directory: c:\users\86150\appdata\local\pip\cache\wheels\33\de\03\7346ae
70da7f980f78569668caf78fb2d678b176e549557c7d
Successfully built gensim
Installing collected packages: gensim
Successfully installed gensim-3.8.1
Note: you may need to restart the kernel to use updated packages.
```

In [14]:

```
from gensim.models import Word2Vec
from sklearn.cluster import KMeans
import warnings
warnings.filterwarnings('ignore')

words = []

for i in news:
    words.append(word_cut(i))
model = Word2Vec(words, sg=0, size=300, window=5, min_count=5)
keys = model.wv.vocab.keys()
wordvector = []
for key in keys:
    wordvector.append(model[key])

distortions = []
for i in range(1, 40):
    word_kmeans = KMeans(n_clusters=i,
                         init='k-means++',
                         n_init=10,
                         max_iter=300,
                         random_state=0)
    word_kmeans.fit(wordvector)
    distortions.append(word_kmeans.inertia_)

plt.plot(range(1, 40), distortions, marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Distortion')
```

Out[14]:

```
Text(0, 0.5, 'Distortion')
```



In [33]:

```
word_kmeans = KMeans(n_clusters=10)  # 聚成10类
word_kmeans.fit(wordvector)

labels = word_kmeans.labels_

for num in range(0, 10):
    text = []
    for i in range(len(keys)):
        if labels[i] == num:
            text.append(list(keys)[i])  # 分别获得10类的聚类结果
    print(text)
```

['摩洛哥', '公共', '逝者', '仪式', '受新冠', '状态', '上调', '警惕', '多国', '低',
'航空', '酒吧', '一周', '塞尔维亚', '波兰', '哈萨克斯坦', '等国', '恶化', '压力',
'加大', '超例', '回升', '阿根廷', '省份', '但', '监护', '扩散', '引', '市', '葡萄
牙', '证明', '严格', '坚决', '省区市', '工作人员', '警告', '疾病', '中心', '返回',
'毕业生', '岗位', '抗议', '发出', '主要', '方式', '欧盟', '外长', '准备', '任何',
'很', '有效', '不是', '无法', '传染病', '重开', '多项', '新西兰', '男子', '酒店',
'总干事', '尚未', '之下', '规定', '接待', '至时', '多家', '两', '行动', '民航局',
'加剧', '做', '放松', '指南', '经', '转机', '这些', '若', '体温', '正常', '重点',
'等级', '创新', '使馆', '自', '莫斯科', '升级', '落实', '欧元', '复苏', '级', '第二
阶段', '期', '外籍', '大区', '再度', '巴基斯坦', '隐瞒', '史', '密接', '所', '两天',
'中国政府', '援', '举办', '会议', '分享', '统计', '匈牙利', '执行', '秘书', '市长',
'今起', '下调', '墨西哥', '保护', '恢复正常', '采购', '呼吸机', '提升', '救治', '建
设', '座', '价格', '吴尊友', '说', '万个', '区域', '吨', '万份', '基本', '采样', '详
情', '解禁', '增幅', '多地', '省', '家中', '禁足', '圈', '使用', '规模', '卫健委日',
'交易', '流行', '澳门', '削减', '监狱', '追加', '世界卫生组织', '办事处', '超人',
'一季度', '化', '其他', '迪士尼', '因新冠', '总', '经验', '视频', '总数', '波', '事
态', '联合国', '包括', '名新冠', '副', '佛罗里达州', '过去', '罚款', '即将', '床位',
'哥伦比亚', '针对', '主流', '三个', '处于', '辽宁大连', '海鲜', '环境', '份', '中
考', '水平', '市民', '补助', '看', '吗', '其', '任务', '移动', '公务员', '进出', '大
部分', '最小', '出租车', '运营', '启用', '发', '保持', '三级', '奥组委', '考虑', '前
往', '用', '认为', '万多', '力度', '流感', '采取', '此前', '诊断', '停课', '缓解',
'有关', '世界', '逾', '投资', '生产', '瑞士', '沙特', '联合', '旅游业', '变化', '赞
比亚', '接收', '系统', '幼儿园', '启程', '现', '都', '发言人', '总领馆', '有序', '禁
令', '强调', '系', '一年', '吉林市', '亿只', '境内', '治愈率', '籍', '央视', '不明',
'民航', '运输', '半数', '封城', '举措', '得到', '四川省', '院士', '复阳', '人民',
'领导', '药物', '零', '紧张', '大会', '召开', '群体', '州长', '病毒感染', '首相',
'封闭式', '河北省', '省市', '合肥', '撤侨', '防护', '接近', '当地', '财政', '开',
'行程', '鲍里斯', '降', '可以', '销售', '黑龙江省', '普京', '也', '援鄂', '首尔',
'夜店', '白宫', '爱心', '俄', '万次', '航线', '同时', '以下', '至人', '提高', '代
表', '网络', '高三', '行业', '人士', '沪', '严禁', '乘', '城市', '陆续', '复学',
'鄂', '加快', '明确', '工资', '而', '英雄', '免疫', '入院', '指导', '危重', '收到',
'告急', '回', '共同', '降至例', '收治', '临床', '返程', '严防', '证据', '不足', '乘
客', '暂', '哈尔滨', '外', '让', '首日', '实现', '宣言', '急需', '小汤山', '派', '辽
宁省', '山西', '大臣', '撤离', '全力', '现在', '第二批', '野生动物', '天津市', '内蒙
古自治区', '神山', '江西省', '记者', '滞留', '福建省', '贵州省', '黎巴嫩', '全区',
'连降', '安徽省', '日前', '河南省', '迎接', '捐款', '火神', '山', '婴儿', '两例',
'急']
['巴西', '达', '从', '地区', '西班牙', '举行', '悼念', '持续', '英国', '万人', '东
京', '年', '发现', '受', '新加坡', '总统', '阳性', '升至', '重启', '国内', '及', '首
都', '首次', '情况', '提供', '委员会', '可能', '连续', '活动', '上升', '暴发', '卫健
委', '均', '于', '仍', '重症', '戴', '性', '实施', '国', '支持', '日本', '继续', '德
国', '其中', '因', '泰国', '法国', '开始', '最', '传播', '意大利', '呼吁', '入境',
'取消', '天津', '今日', '时', '海外', '旅客', '特朗普', '要求', '新型', '冠状病毒',
'内', '单日', '北京市', '解除', '计划', '非洲', '应', '来', '大使馆', '公民', '要',
'扩大', '美', '近', '增加', '最新', '亿', '日时', '令', '无症状', '感染者', '总理',
'新闻', '前', '市场', '目前', '最大', '完成', '假期', '关闭', '疫苗', '欧洲', '应

急', '全部', '者', '企业', '疑似病例', '医护人员', '学校', '增至', '奥运会', '居家', '发热', '卫生', '专家组', '抗体', '实行', '一级', '公主']
['新增', '确诊', '病例', '累计', '例', '境外', '输入']
['口罩', '将', '恢复', '与', '国家', '的', '在', '疫情', '防控', '病毒', '防疫', '措施', '中国', '抗疫', '物资', '为', '和', '已', '隔离', '核酸', '检测', '人员', '医院', '武汉', '不', '医疗']
['加速', '增长', '佩戴', '逐步', '香港', '到', '没有', '新疆', '最高', '曾', '呈', '卫生部长', '建议', '机构', '超万', '重庆', '伊朗', '俄罗斯', '菲律宾', '印度', '以来', '复课', '调整', '正', '发生', '至少', '阿联酋', '旅行', '封锁', '安全', '控制', '提醒', '本土', '需', '强制', '高风险', '推迟', '约', '高校', '面临', '大规模', '研究', '进一步', '我', '现有', '观察', '直播', '反弹', '全面', '年月日时', '辽宁', '大幅', '小时', '发地', '卫生部', '加拿大', '管理', '岁', '以上', '合作', '放宽', '管控', '推出', '各国', '数据', '注意', '回国', '江西', '是否', '广东', '土耳其', '回应', '开展', '成', '会', '并', '允许', '解封', '社区', '已有', '医生', '首批', '延期', '已经', '大', '禁止', '安徽', '社会', '方舱', '截至', '集中', '清零', '来自', '消费', '韩国', '相关', '有例', '日起', '结束', '加强', '做好', '密切接触', '抗击', '不会', '一天', '再次', '公共卫生', '出台', '服务', '第例', '媒体', '捐赠', '上', '事件', '援助', '海南', '时间', '机场', '四川', '显示', '中小学', '一律', '云南', '未', '天无', '支援', '赴', '好消息', '复产', '同胞', '钟南山', '疑似', '临时', '各地', '景区', '年月日', '抵达', '约翰逊', '广西', '队员', '逝世', '铁路', '儿童', '黑龙江', '五一', '山东', '牺牲', '亿元', '痊愈', '突发', '浙江', '含', '兵团', '下半旗', '烈士', '钻石']
['超', '万例', '世卫', '组织', '全球', '超过']
['美国', '起', '民众', '进入', '月', '部分', '驻', '患者', '人数', '北京', '向', '个', '中', '风险', '影响', '至', '开放', '又', '一', '病毒检测', '航班', '再', '延长', '全国', '出现', '后', '进行', '对', '应对', '天', '经济', '聚集', '感染', '发布', '人', '名', '宣布', '万', '所有', '号', '或', '限制', '有', '多', '称', '了', '工作', '政府', '是', '丨', '被', '可', '等', '新', '医疗队', '启动', '专家', '返校', '期间', '健康', '暂停', '复工', '首例', '紧急', '国际', '公布', '响应', '湖北', '开学']
['例新冠', '肺炎', '新冠', '日', '无', '报告', '治愈', '出院', '达例', '上海', '本地', '死亡', '通报']
['下周', '封闭', '必须', '外交部', '形势', '冲击', '月份', '主席', '发展', '部长', '秘鲁', '客运', '澳大利亚', '餐厅', '结果', '英国首相', '未来', '调查', '感谢', '一名', '决定', '昨日', '加州', '治疗', '就', '问题', '公司', '迪拜', '以外', '居民', '申请', '裁员', '严重', '养老院', '员工', '例为', '小区', '出行', '比利时', '高考', '今年', '万名', '失业', '宵禁', '羟', '氯喹', '最后', '推动', '留学生', '蔓延', '达到', '每日', '场所', '旅游', '游客', '同比', '学生', '数', '官员', '疾控中心', '官方', '重新', '家庭', '考生', '范围', '人群', '地', '一个', '内蒙古', '高', '两周', '社交', '距离', '康复', '购买', '线上', '就业', '南非', '州', '能力', '发放', '阴性', '症状', '排除', '导致', '月底', '边境', '通过', '关于', '信息', '每天', '多名', '纽约', '破', '食品', '生活', '运抵', '助力', '失业率', '项目', '还', '常态', '福建', '首个', '集体', '蔬菜', '快递', '家', '医疗机构', '共', '团队', '实验室', '接触', '紧急状态', '万人次', '研发', '积极', '预计', '资金', '批准', '下', '河北', '保障', '成为', '甘肃', '第二', '军队', '医学观察', '下降', '更', '发布会', '由', '江苏', '埃及', '医用', '马来西亚', '刚果', '门诊', '确认', '中方', '逼近', '张文宏', '非', '正式', '关联', '样本', '一线', '医务人员', '师生', '河南', '安排', '业务', '停止', '排查', '级别', '二级', '需要', '武汉市', '关注', '不得', '陕西', '确定', '战疫', '我国', '病人', '快速', '突破', '以', '个人', '返京', '回家', '占', '接受', '湖北省', '中央', '试剂', '国务院', '联防', '联控', '机制', '通知', '去世', '今天', '宁夏', '年级', '趋缓', '广州', '救助', '政策', '高峰', '外出', '阶段', '口岸', '包机', '重要', '测试', '轨迹', '吉林', '吉林省', '纽约州', '部门', '只', '工作者', '预约', '避免', '表明', '护士', '试剂盒', '共有', '免费', '绥芬河', '邮轮', '表示', '除', '医护', '重庆市', '诊疗', '驰援', '金银', '潭', '全省', '贵州', '湖南', '云南省', '山东省', '志哀', '深切', '广东省', '黄冈', '西藏', '青海']
['阿塞拜疆', '科威特', '塞内加尔', '白俄罗斯', '越南', '国际航班', '引发', '好', '营业', '下跌', '乌克兰', '一个月', '保加利亚', '多州', '乌兹别克斯坦', '住院', '严峻', '例均', '大厅', '希腊', '两个', '展开', '筛查', '工人', '冠', '一例', '北美', '危机', '减少', '奥地利', '洛杉矶', '加纳', '阿曼', '地方', '关键', '心理', '出席', '须', '帮助', '肯尼亚', '死于', '倍', '老人', '荷兰', '如何', '圭亚那', '案例', '具备', '给', '吉尔吉斯斯坦', '挑战', '数超', '序列', '捷克', '名单', '近万', '外卖',

'预测', '卫健委月', '比', '印尼', '上海市', '正在', '考试', '智利', '比赛', '进京', '量', '多数', '纳入', '明显', '海滩', '经济衰退', '厄瓜多尔', '疾控', '主任', '尚', '条', '建', '教育部', '蛋白质', '布', '全', '团结', '全员', '型', '全体', '沈阳', '默哀', '变', '斯里兰卡', '纽约市', '防止', '处以', '停运', '生命', '也门', '贫民窟', '通告', '参加', '就诊', '作用', '供应', '不断', '突尼斯', '以色列', '参与', '复航', '共计', '流动', '分批', '孟加拉国', '牡丹江', '临床试验', '赤道几内亚', '金', '近例', '乌拉圭', '破万', '尼日利亚', '次', '啦', '航空公司', '马里', '苏丹', '亚洲', '阿尔及利亚', '胜利', '近万人', '轻症', '武汉协和医院', '叙利亚', '卡塔尔', '伊拉克', '各', '错峰', '缅甸', '发改委', '毛里求斯', '柳叶刀', '数量', '检疫', '埃塞俄比亚', '舒兰市', '供应链', '西藏自治区', '日本政府', '江苏省', '情况通报', '补贴', '津巴布韦', '办理', '巴林', '亚美尼亚', '同一', '过万', '网友', '两万', '传染', '医务', '堂食', '构成', '上班', '浙江省', '文莱', '南京', '喀麦隆', '出征', '吉布提', '斯洛伐克', '凯旋', '卢森堡', '病情', '订正', '利比亚', '布基纳法索', '资助', '第一批', '例例', '雷', '蒙古国', '青海省', '至例', '千例', '增例', '立陶宛', '安道尔', '坦桑尼亚', '阿尔巴尼亚', '山西省', '格鲁吉亚', '老挝', '湖南省', '塞浦路斯', '台湾', '黄石', '有名', '襄阳', '深圳', '春节假期']

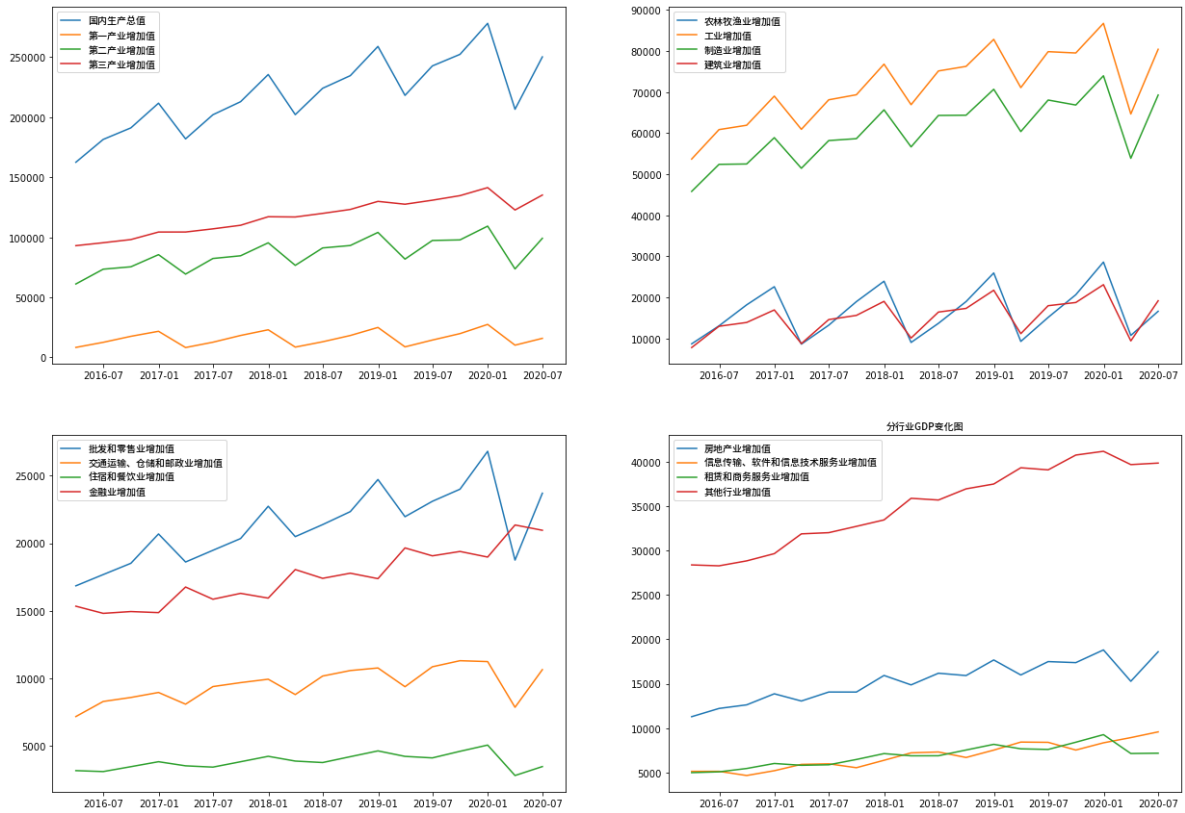
In [15]:

```
sum_GDP = ['国内生产总值', '第一产业增加值', '第二产业增加值', '第三产业增加值']
industry_GDP = ['农林牧渔业增加值', '工业增加值', '制造业增加值', '建筑业增加值']
industry2_GDP = ['批发和零售业增加值', '交通运输、仓储和邮政业增加值', '住宿和餐饮业增加值', '金融业
industry3_GDP = ['房地产业增加值', '信息传输、软件和信息技术服务业增加值',
                 '租赁和商务服务业增加值', '其他行业增加值']

fig = plt.figure()
fig, axes = plt.subplots(2, 2, figsize=(21, 15))

axes[0][0].plot(data_economy[sum_GDP])
axes[0][0].legend(sum_GDP, prop=myfont)
axes[0][1].plot(data_economy[industry_GDP])
axes[0][1].legend(industry_GDP, prop=myfont)
axes[1][0].plot(data_economy[industry2_GDP])
axes[1][0].legend(industry2_GDP, prop=myfont)
axes[1][1].plot(data_economy[industry3_GDP])
axes[1][1].legend(industry3_GDP, prop=myfont)

plt.title('分行业GDP变化图', fontproperties=myfont)
```

Out[15]:

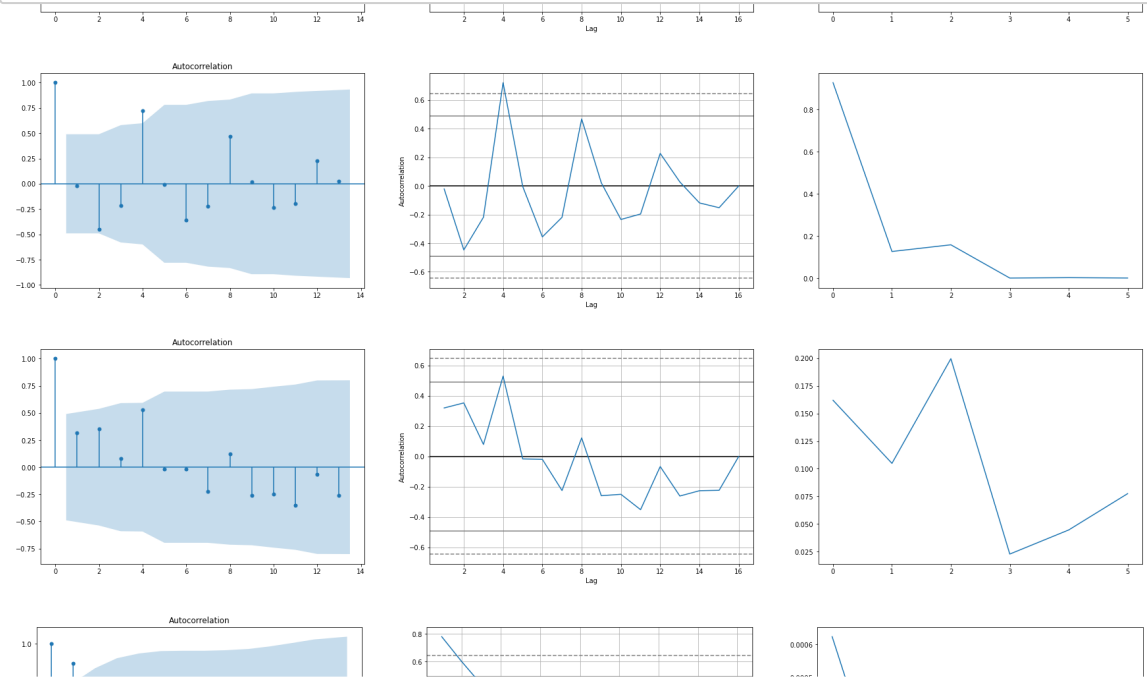
```
Text(0.5, 1.0, '分行业GDP变化图')
```

```
<Figure size 432x288 with 0 Axes>
```



In [35]:

```
from statsmodels.graphics.tsaplots import plot_acf
from pandas.plotting import autocorrelation_plot
from statsmodels.sandbox.stats.diagnostic import acorr_ljungbox


GDP_type = ['国内生产总值', '第一产业增加值', '第二产业增加值', '第三产业增加值',
            '农林牧渔业增加值', '工业增加值', '制造业增加值', '建筑业增加值', '批发和零售业增加值',
            '交通运输、仓储和邮政业增加值', '住宿和餐饮业增加值', '金融业增加值',
            '房地产业增加值', '信息传输、软件和信息技术服务业增加值', '租赁和商务服务业增加值', '其他

for i in GDP_type:
    each_data = data_economy[i][:-2]
    plt.figure(figsize=(30, 6))
    ax1 = plt.subplot(1, 3, 1)
    ax2 = plt.subplot(1, 3, 2)
    ax3 = plt.subplot(1, 3, 3)
    LB2, P2 = acorr_ljungbox(each_data)  # 进行纯随机性检验
    plot_acf(each_data, ax=ax1)
    autocorrelation_plot(each_data, ax=ax2)  # 进行平稳性检验
    ax3.plot(P2)
```



In [16]:

```
from statsmodels.tsa.arima_model import ARMA
from statsmodels.tsa.stattools import arma_order_select_ic

warnings.filterwarnings('ignore')
data_arma = pd.DataFrame(data_economy['国内生产总值'][:-2])
a, b = arma_order_select_ic(data_arma, ic='hqic')['hqic_min_order']
arma = ARMA(data_arma, order=(a, b)).fit()
rate1 = list(data_economy['国内生产总值'][-2] /
             arma.forecast(steps=1)[0])
rate1
```

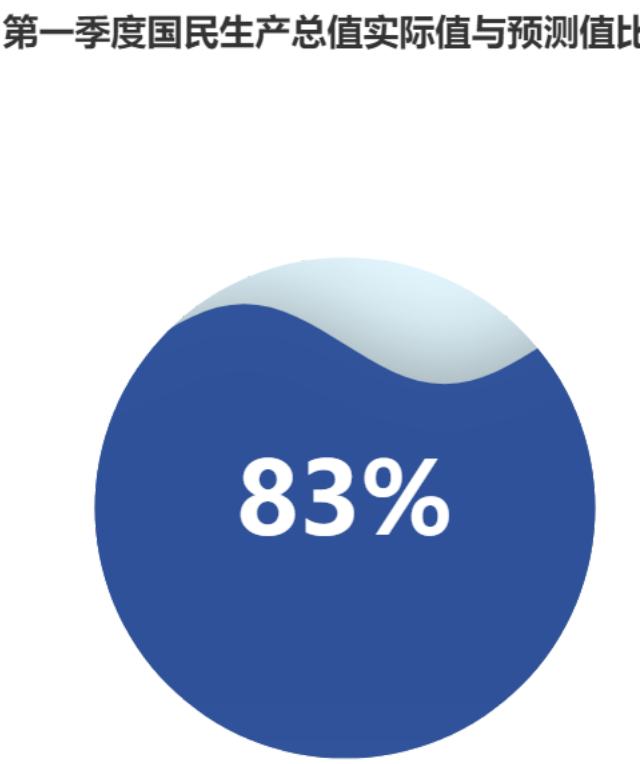
Out[16]:

```
[0.8273103019180329]
```

In [17]:

```
from pyecharts import options as opts
from pyecharts.charts import Liquid

c = (
    Liquid()
    .add("实际值/预测值", rate1, is_outline_show=False)
    .set_global_opts(title_opts=opts.TitleOpts(title="第一季度国民生产总值实际值与预测值比例",
                                               pos_left="center"))
)
c.render_notebook()
```

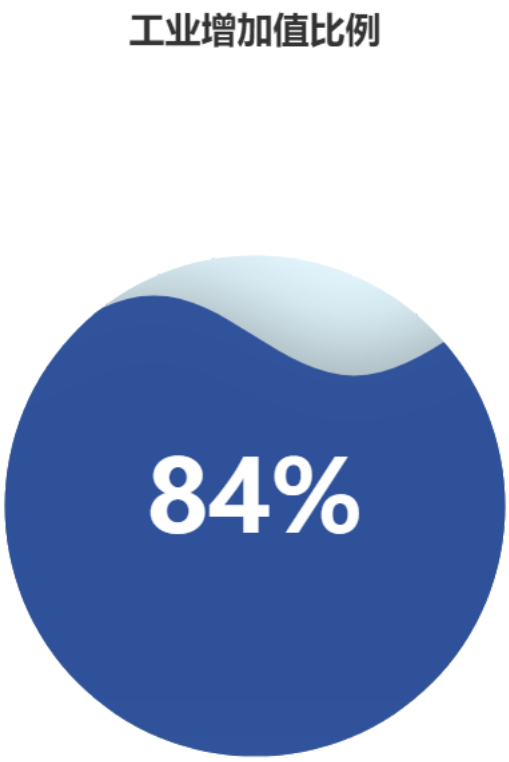
Out[17]:



In [18]:

```
warnings.filterwarnings('ignore')
data_arma = pd.DataFrame(data_economy['工业增加值'][:-2])
a, b = arma_order_select_ic(data_arma, ic='hqic')['hqic_min_order']
arma = ARMA(data_arma, order=(a, b)).fit()
rate2 = list(data_economy['工业增加值'][-2]/arma.forecast(steps=1)[0])
c = (
    Liquid()
    .add("实际值/预测值", rate2, is_outline_show=False)
    .set_global_opts(title_opts=opts.TitleOpts(title="工业增加值比例", pos_left="center"))
)
c.render_notebook()
```

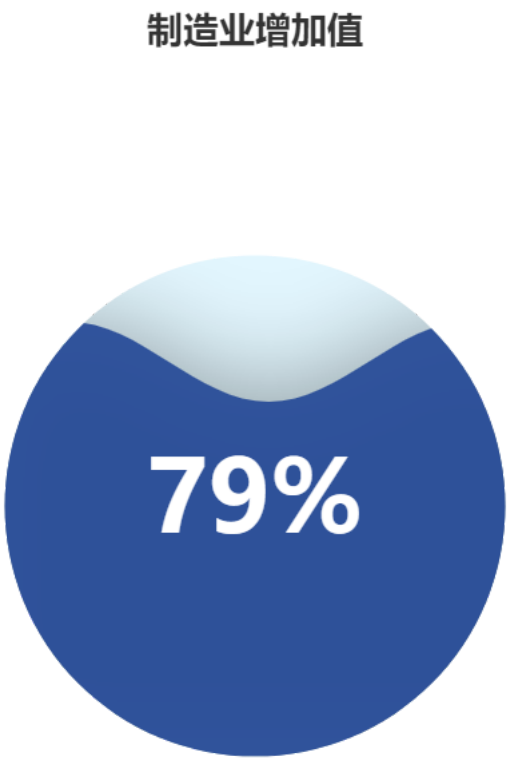
Out[18]:



In [19]:

```
warnings.filterwarnings('ignore')
data_arma = pd.DataFrame(data_economy['制造业增加值'][:-2])
a, b = arma_order_select_ic(data_arma, ic='hqic')['hqic_min_order']
arma = ARMA(data_arma, order=(a, b)).fit()
rate3 = list(data_economy['制造业增加值'][-2]/arma.forecast(steps=1)[0])
c = (
    Liquid()
    .add("实际值/预测值", rate3, is_outline_show=False)
    .set_global_opts(title_opts=opts.TitleOpts(title="制造业增加值", pos_left="center"))
)
c.render_notebook()
```

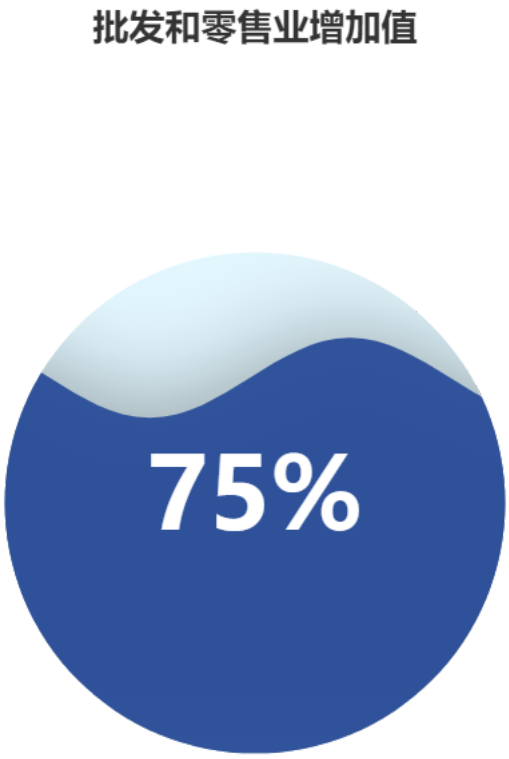
Out[19]:



In [20]:

```
data_arma = pd.DataFrame(data_economy['批发和零售业增加值'][:-2])
a, b = arma_order_select_ic(data_arma, ic='hqic')['hqic_min_order']
arma = ARMA(data_arma, order=(a, b)).fit()
rate4 = list(data_economy['批发和零售业增加值'][-2]/arma.forecast(steps=1)[0])
c = (
    Liquid()
    .add("实际值/预测值", rate4, is_outline_show=False)
    .set_global_opts(title_opts=opts.TitleOpts(title="批发和零售业增加值", pos_left="center"))
)
c.render_notebook()
```

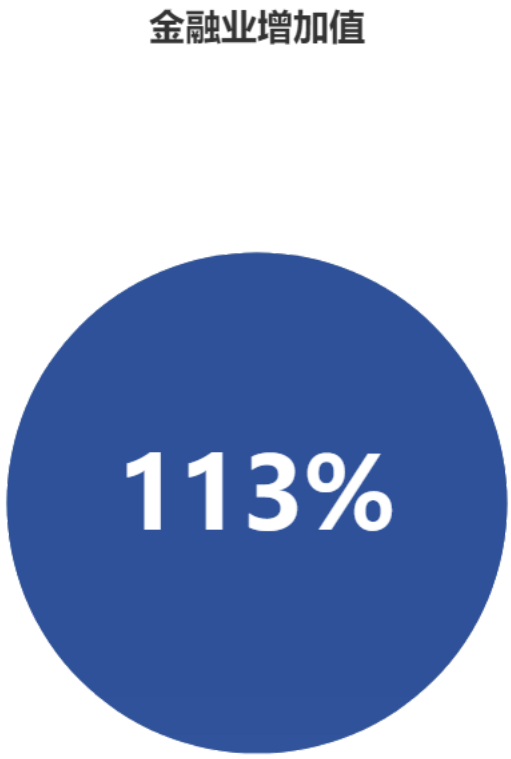
Out[20]:



In [21]:

```
data_arma = pd.DataFrame(data_economy['金融业增加值'][:-2])
a, b = arma_order_select_ic(data_arma, ic='hqic')['hqic_min_order']
arma = ARMA(data_arma, order=(a, b)).fit()
rate = list(data_economy['金融业增加值'][-2]/arma.forecast(steps=1)[0])
c = (
    Liquid()
    .add("实际值/预测值", rate, is_outline_show=False)
    .set_global_opts(title_opts=opts.TitleOpts(title="金融业增加值", pos_left="center"))
)
c.render_notebook()
```

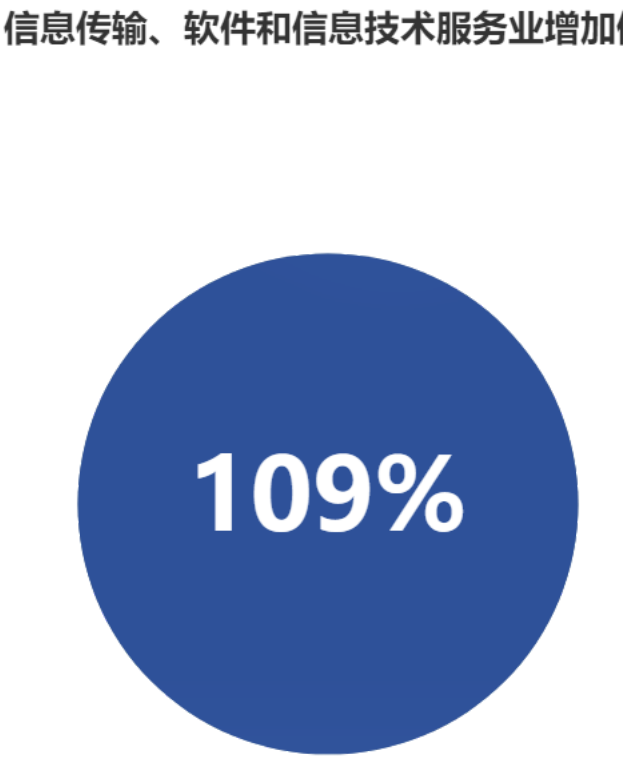
Out[21]:



In [22]:

```
data_arma = pd.DataFrame(data_economy['信息传输、软件和信息技术服务业增加值'][:-2])
a, b = arma_order_select_ic(data_arma, ic='hqic')['hqic_min_order']
arma = ARMA(data_arma, order=(a, b)).fit()
rate = list(data_economy['信息传输、软件和信息技术服务业增加值'][-2]/arma.forecast(steps=1)[0])
c = (
    Liquid()
    .add("实际值/预测值", rate, is_outline_show=False)
    .set_global_opts(title_opts=opts.TitleOpts(title="信息传输、软件和信息技术服务业增加值",
                                               pos_left="center"))
)
c.render_notebook()
```

Out[22]:



In [ ]:

In [ ]: