# 基于IDF的全球糖尿病数据分析

## Global diabetes data analysis based on IDF

**汇报时间：2024年5月**
**汇报人：伍鑫 乔妍妍 杨若妍**

# 01

# 小　组　分　工

伍鑫编年史

# 小组分工

**伍鑫**

- 项目规划与协调：
  - 作为组长，制定项目计划，明确各阶段目标，分配团队成员的任务。
  - 保持与团队成员的沟通，确保信息畅通，并向相关人员汇报项目状态。
  - 分析IDF网站数据，确定要爬取的数据类型和内容。
- 爬虫开发：开发和测试爬虫脚本，以获取所需数据；实现错误处理和异常检测机制，确保爬取过程的鲁棒性。
- 数据清洗与预处理：清洗爬取的数据，去除无用信息和噪声；格式化数据，确保一致性，便于后续分析。
- 数据分析与可视化：使用统计学方法和数据挖掘技术分析数据；利用图表和可视化工具展示分析结果。
- 机器学习部分：随机森林预测。

**乔妍妍**

**杨若妍**

- 现象分析：
  - 对图表展示的现象进行深入分析，比如糖尿病患病率的变化趋势、不同国家间的差异、影响因素等。
  - 结合背景知识和现有文献，对图表中的突出现象提出初步解释。
- 报告编写：
  - 编写分析报告，详细说明图表制作的过程、现象描述、原因分析以及结论；
  - 在报告中加入图表解释和数据来源，确保报告的逻辑连贯性和可信度。。
- 调整与优化：根据反馈对图表和分析进行调整和优化；提高图表的精确度和分析的深度，确保结果的科学性和实用性。
- 机器学习部分：梯度提升回归模型与预测。

# CONTENT

# 02

# 选题背景介绍

# 研究与分析意义

- 网站名称：国际糖尿病联盟（IDF）糖尿病地图集
- 网址：https://diabetesatlas.org/data/en/country/

2021年全球约有5.37亿成年人患有糖尿病，而预计到2030年，这一数字将增至6.43亿。同时，2021年约有670万人死于糖尿病。**糖尿病已成为全球性的重大公共卫生问题**，对人们的健康和生命安全造成了严重威胁。

研究该网站，可以帮助我们预测糖尿病的未来发展趋势。通过模型预测，我们可以提前了解糖尿病的流行趋势，为制定**预防和控制**策略提供参考。

同时，我们结合了肥胖网和numbeo网站，帮助我们理解糖尿病的发病机制，找出影响糖尿病患病率的因素。深入研究这些因素，我们可以找到预防和控制糖尿病的有效途径，为制定针对性的**干预措施**提供科学依据。

**5.37亿 (2021)**

**2021 年 全球糖尿病情况**

大约5.37 亿成年人（20-79 岁）患有糖尿病。同时，其中有2.4亿未确诊的糖尿病患者，也就是说几乎一半人不知道自己患有这种疾病。

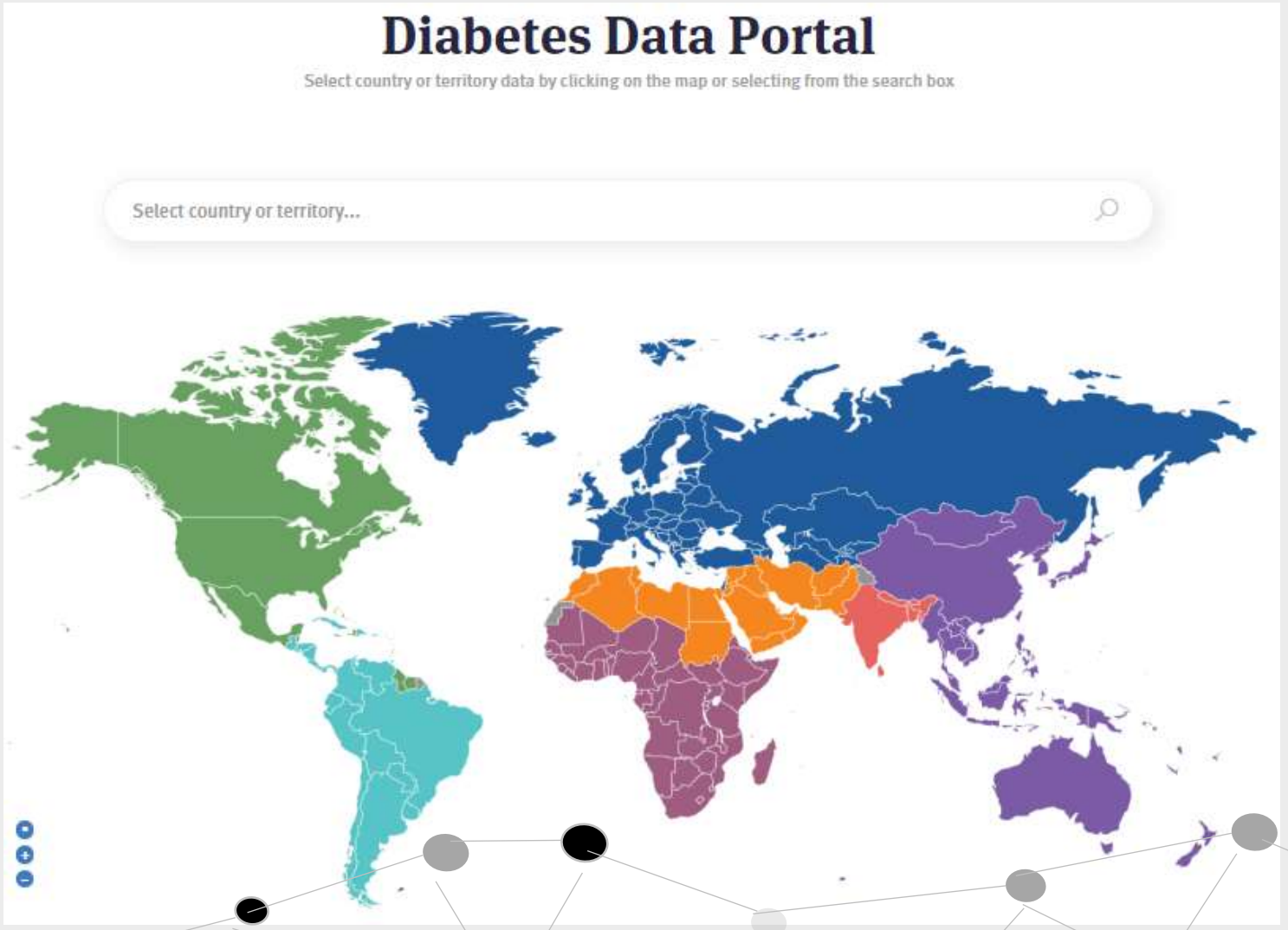
**670万 死亡人数**

**死亡人数**

2021年,糖尿病造成了670万人死亡,平均每5秒1人死亡。

# 网页介绍

- 网站名称：国际糖尿病联盟（IDF）糖尿病地图集
- 网址：https://diabetesatlas.org/data/en/country/




- **内容：**该网站提供了全球范围内的糖尿病统计数据，包括不同国家和地区的糖尿病患病率、死亡率、医疗支出以及相关的分析和预测。
- **功 能：** 通过这个网站，研究人员、医生、政策制定者以及公众都可以获取到最新的全球糖尿病统计数据，从而对糖尿病的流行病学特征有一个清晰的了解，并为预防、治疗和控制糖尿病提供决策依据。

# 选择原因

- 网站名称：国际糖尿病联盟（IDF）糖尿病地图集
- 网址：https://diabetesatlas.org/data/en/country/


- **数据具有权威性：**该网站由国际糖尿病联盟（IDF）维护，作为一个全球性的组织，其发布的数据具有较高的权威性和可信度。
- **信息覆盖全面：**网站提供的数据具有广泛性和多样性，涵盖了全球多个国家和地区，包含了多种糖尿病相关指标，如患病率、死亡率、医疗花费等。因此该网站十分适合进行糖尿病数据分析，为分析和研究提供了广泛的视角和深度。
- **数据格式 具 有 规范性：**该网站以图表和地图形式展示数据，清晰直观。
- **使用友好：**界面设计简洁，数据检索和爬取操作相对简单。

# 补充两种患病情况

## IGT(Impaired glucose tolerance)

IGT是指糖耐量减低，是一种代谢异常状态，表现为血糖水平高于正常范围，但还不足以被诊断为糖尿病。它表明个体存在发展2型糖尿病的风险增加，以及其他心血管疾病和代谢综合症的风险也可能增加，是一个重要的临床标志。因此，对IGT的早期识别和干预对于预防糖尿病和改善公共健康具有重要意义。

这些估计对于评估特定人群中IGT的普遍程度及其潜在的公共卫生影响至关重要。

## IFG(Impaired fasting glucose)

IFG是指空腹血糖受损，是指空腹血糖水平略高于正常范围，但尚未达到糖尿病的诊断标准。这种状态表明个体存在发展糖尿病的风险，同时也可能增加心血管疾病和其他代谢性疾病的风险。IFG是一个重要的健康警示，提醒人们关注生活方式和饮食习惯的调整，以预防糖尿病和相关并发症的发生。

对IFG的早期识别和干预对于改善个体健康状况和降低糖尿病的患病率具有重要意义。

# 03

# 数　据　获　取

**获取方法及代码**

**数据量**

**数据清洗**

# 01.获取方法及代码

- 函数1：AllCountry

```
def AllCountry():
    Country_url =  'https://diabetesatlas.org/data/en/'
    Country_html = requests.get(Country_url)
    Country_html.encoding = 'utf-8'
    Country_Soup = BeautifulSoup(Country_html.text,'lxml')
    tit = Country_Soup.find('select',attrs={'id':'idf-country-list'}).findAll('option')
    for i in tit[1:]:
        a=(i["value"])
        list.append(a)
        Country_Url.append('https://diabetesatlas.org/data/en/country/'+a.replace('|','/')+'.html')
```

定义AllCountry的函数，以便在网站中获取所有国家的名称和对应的链接。

其中，使用了BeautifulSoup解析HTML，提取每个国家的标识符，并构建对应的详细页面URL，最后将这些URL添加到列表中。

- 函数2：AllCountryNews01

```
def AllCountryNews01():
#获取各个人数等数据
    print(Country_Url)
    sum=0
    for url in Country_Url:
        sum=sum+1
        print(sum)
        html = requests.get(url)
        html.encoding = 'utf-8'
        Soup = BeautifulSoup(html.text,'lxml')
        tit1 = Soup.find('h1')
        CountryName.append(tit1.text)
        tit = Soup.find('table',attrs={'id':'idf-country-data'}).findAll('td')
```

遍历Country_Url列表中的每个URL（预期是包含各国糖尿病数据的网页链接），对每个URL发起网络请求获取HTML内容，并使用BeautifulSoup进行解析。

然后，从每个网页中查找<h1>标签获取国家名称，并添加到CountryName列表中。同时，它还使用一个变量sum

# 02.数据量


从网站中爬取出CSV文件，包含222个国家，共38×222个数据。我们从2011、2021、2030、2045四个年份出发，分别对各国家全国总人口、全国糖尿病患病率、患IGT型糖尿病的人数及患病率、患IFG型糖尿病的人数及患病率、20-79岁人群对糖尿病的金钱总投入与个人花费的数据进行爬取。

**因为从IDF网址获得的关于影响因素的数据不足，因此又选择了两个网站进行爬取。**

**numbeo网站：**肥胖率、超重。

**全球肥胖观察网站：**生活质量指数、采购功率指数、安全指数、健康护理指数、生活成本指数、污染指数

在最初的CSV文件中，我们发现**在数据展示时，部分国家名在爬下来的数据data01和Map中有所差别，因此要把爬下来的个别名字进行替换，**且**网站中将香港、澳门单独列举出了**，因此我们对得到的数据进行进一步清理，得出清理后的数据表格“dataAfterClean.csv”。

# 03.数据清洗

## 处理网页中可能存在的'-'等无效数据

```
#数据清理
# (一) (2011-2045) 该国家患病人数
#818.3 1606.7  2172.8  3530.7
ListYearPerson['2011'].append(Clean(tit[1].text))
ListYearPerson['2021'].append(Clean(tit[2].text))
ListYearPerson['2030'].append(Clean(tit[3].text))
ListYearPerson['2045'].append(Clean(tit[4].text))
# (二) (2011-2045) 该国家年龄调整比较患病率
ListYearPersonPrevalence['2011'].append(Clean(tit[6].text))
ListYearPersonPrevalence['2021'].append(Clean(tit[7].text))
ListYearPersonPrevalence['2030'].append(Clean(tit[8].text))
ListYearPersonPrevalence['2045'].append(Clean(tit[9].text))
# (三) (2011-2045) IGT患病人数
IGT['2011'].append(Clean(tit[21].text))
IGT['2021'].append(Clean(tit[22].text))
IGT['2030'].append(Clean(tit[23].text))
IGT['2045'].append(Clean(tit[24].text))
# (四) (2011-2045) 该国家IGT年龄调整比较患病率
IGTPrevalence['2011'].append(Clean(tit[26].text))
IGTPrevalence['2021'].append(Clean(tit[27].text))
IGTPrevalence['2030'].append(Clean(tit[28].text))
IGTPrevalence['2045'].append(Clean(tit[29].text))
# (五) (2011-2045) IFG患病人数
IFG['2011'].append(Clean(tit[31].text))
IFG['2021'].append(Clean(tit[32].text))
IFG['2030'].append(Clean(tit[33].text))
IFG['2045'].append(Clean(tit[34].text))
# (六) (2011-2045) 该国家IFG年龄调整比较患病率
IFGPrevalence['2011'].append(Clean(tit[36].text))
IFGPrevalence['2021'].append(Clean(tit[37].text))
IFGPrevalence['2030'].append(Clean(tit[38].text))
IFGPrevalence['2045'].append(Clean(tit[39].text))
print(ListYearPerson,ListYearPersonPrevalence,IGT,IGTPrevalence,IFG,IFGPrevalence)
list2=[]
list3=[]
for i in tit[92:95]:
    list2.append(Clean(i.text))
for i in tit[102:105]:
    list3.append(Clean(i.text))
```

## 数据清理及整理

针对各国家在2011-2045年间的患病人数、年龄调整比较患病率等数据，从tit列表中提取文本，并经过Clean函数处理后，分别添加到对应的字典（如ListYearPerson、ListYearPersonPrevalence等）的相应年份键下，并打印出这些字典的内容

最后，代码从tit列表中提取特定索引范围的元素，经过清理后添加到两个新列表list2和list3中，这些可能与糖尿病相关的卫生支出有关

# 03.数据清洗

## 替换个别国家名

```python
def AllCountryNews02():
    Country = []
    OverWeight = []
    Obesity = []
    data = pd.read_csv('dataafterClean.csv')
    print(data["Country"])
    Country_url = 'https://data.worldobesity.org/tables/prevalence-of-adult-overweight-obesity-2/'
    Country_html = requests.get(Country_url)
    Country_html.encoding = 'utf-8'
    Country_Soup = BeautifulSoup(Country_html.text, 'lxml')
    tit = Country_Soup.find('table', attrs={'id': 'results', 'class': 'results'}).findAll('td')
    for i in range(247):
        Country.append(tit[0 + i * 10].text)
        OverWeight.append(tit[8 + i * 10].text)
        Obesity.append(tit[9 + i * 10].text)
    Clean02(Country)
    Clean02(OverWeight)
    Clean02(Obesity)
    print(type(Obesity[0]))
    for i in data["Country"]:
        if i in Country:
            if i not in OverWeightObesity['Country']:
                OverWeightObesity['Country'].append(i)
                OverWeightObesity['Overweight'].append(float(OverWeight[Country.index(i)]))
                OverWeightObesity['Obesity'].append(float(Obesity[Country.index(i)]))
        else:
            print(i)
            OverWeightObesity['Country'].append(i)
            OverWeightObesity['OverWeight'].append(0)
            OverWeightObesity['Obesity'].append(0)
    countryEnd = ['Bosnia and Herz.', 'Central African Rep.', 'Channel Islands', 'Curaçao', 'Czech Rep',
                  'Dem. Rep. Congo',
                  'Dominican Rep', 'Faroe Islands', 'Guinea-Bissau', 'Lao PDR',
                  'Netherlands Antilles', 'Korea', 'Russia',
                  'South Sudan', 'State of Palestine', 'S. Sudan', 'United Republic of Tanzania']
    country = ['Bosnia and Herzegovina', 'Central African Republic', 'Chad', 'Croatia', 'Czechia',
               'Democratic Republic of Congo',
               'Dominican Republic', 'Finland', 'Guinea', 'Laos',
               'Netherlands', 'South Korea', 'Russian Federation',
               'Sudan', 'Palestine', 'Sweden', 'Tanzania']
    Clean03(Country, OverWeight, Obesity, countryEnd, country)
```

- **爬虫：**首先从一个CSV文件和一个网页中提取关于国家、超重和肥胖的数据，并整理到一个名为OverWeightObesity的字典中。
- **读取：**读取CSV文件并打印出其中的国家列表，接着，从指定的网页上获取HTML内容，并解析出表格中的国家、超重和肥胖数据。
- **数据清洗：**对获取数据进行清洗（调用Clean02函数），并检查CSV文件中的每个国家名是否存在于网页数据中。若存在，则将相应的超重和肥胖数据添加到OverWeightObesity字典中；如果不存在，则将超重和肥胖数据设为0
- **替换个别国家名：**对数据data01和Map中不同的国家名称进行了处理，修正数据。

# 04

# 数据展示与绘图

01 MAP

02 折线统计图

03 折线统计图

通过分析统计图表并且得出相关结论。

# 01 MAP

**绘制地图，并生成html页面，直观地同时展示了二百多个国家的数据。**

## 绘制函数

**绘制地图形式优势：**

- **直观、易懂：使二百多个国家的数据直观表示的最好方法。**
  - **因为国家数众多，地图形式远比柱状图优越。**
  - **基于用户对于现实世界空间认知的能力，用户不需要具备专业的统计或数据分析背景就能理解地图所传达的信息。**
- **容易识别：在地图上，数据可以通过颜色、地理位置等视觉元素来表示，这使得用户可以迅速发现数据中的突出点和异常值。**

```python
def MapWord(dataCountry,dataShuJu,title,subtitle,range_text_2,pieces,html01):
    print(pieces[-1])
    c=(
        Map(init_opts=opts.InitOpts(width="1400px",height='600px',theme='vintage'))#图表大小
        .add(
            "",#系列名称
            [list(z) for z in zip(dataCountry,dataShuJu)],#使用数据
            "world",#地图格式world_世界地图，China_中国地图
            is_map_symbol_show=False,)#是否显示红色小圆点
        .set_series_opts(label_opts=opts.LabelOpts(is_show=False))#标签不显示（国家名称不显示）
        .set_global_opts(
            title_opts=opts.TitleOpts(
                title=title,#主标题
                pos_left='20%',#位置
                subtitle=subtitle#副标题
            ),  # 位置
            visualmap_opts=opts.VisualMapOpts(#图列设置
                type_='color',#映射方式，color或者size
                max_=pieces[-1],
                min_=pieces[0],
                range_text=['',range_text_2],#图例的文字
                orient='vertical',#图例的方式水平，水平horizontal  竖直vertical
                is_inverse=False,#是否反转
            ),
        )
    )
    c.render(html01)
data=pd.read_csv('dataAfterClean.csv')
pieces=[0,150000]
```

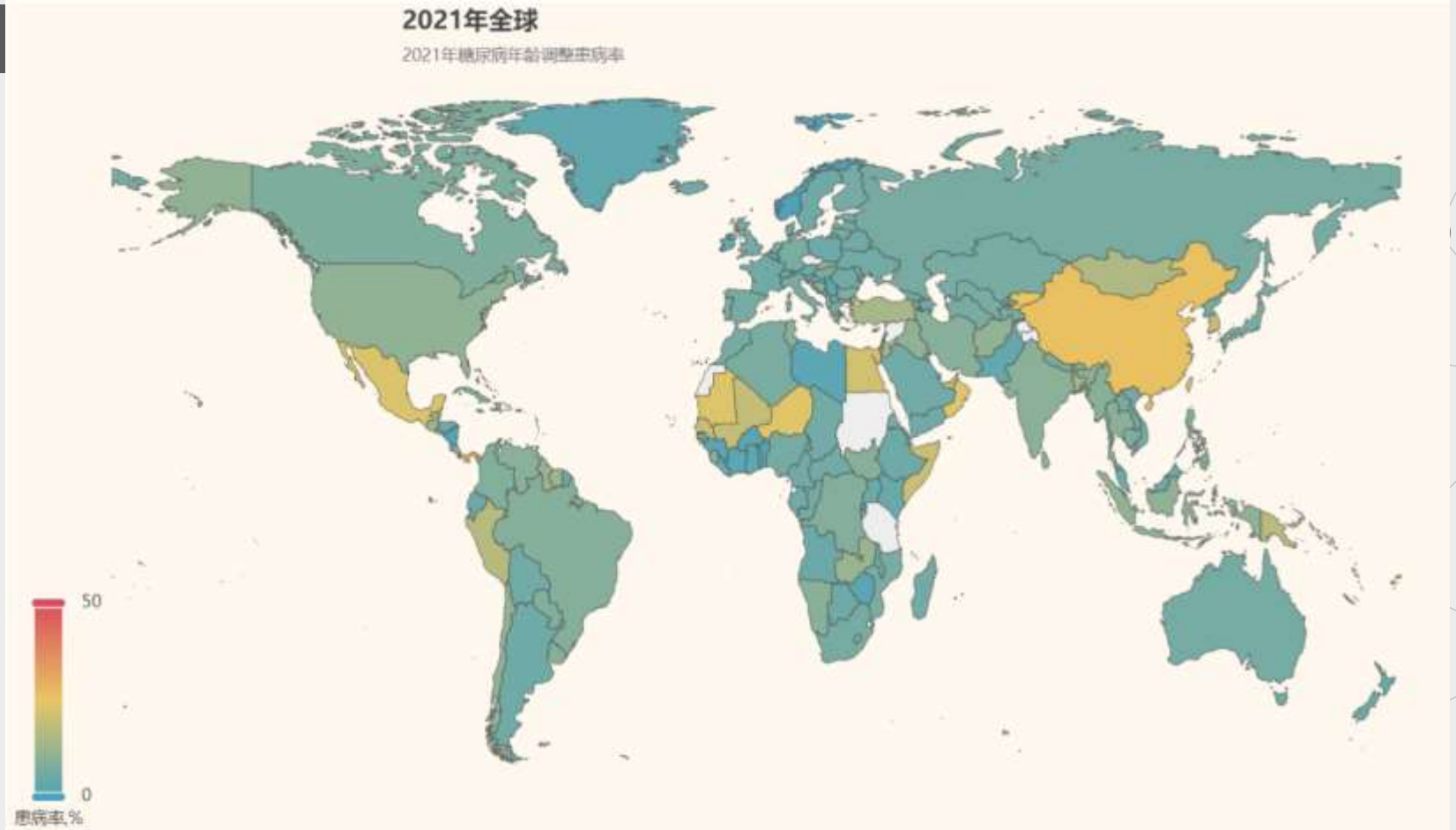
# 01 MAP

**绘制地图，并生成P2021.html。**

## A.2021年年龄调整患病率

**此地图展示了2021年各国患糖尿病年龄调整患病率。**

- **如图中举例，智利年龄调整患病率约为10.8%**
- **根据示例图，颜色代表了患病率的高低，可以直观得看出数据异常，从此图可以得出最高值——在亚洲，中国糖尿病患病率较高。**




```
MapWord(data['Country'],data['ListYearPersonPrevalence_2021'],"2021年全球","2021年糖尿病年龄调整患病率",'患病率,%',pieces,"P2021.html")
```

# 01 MAP

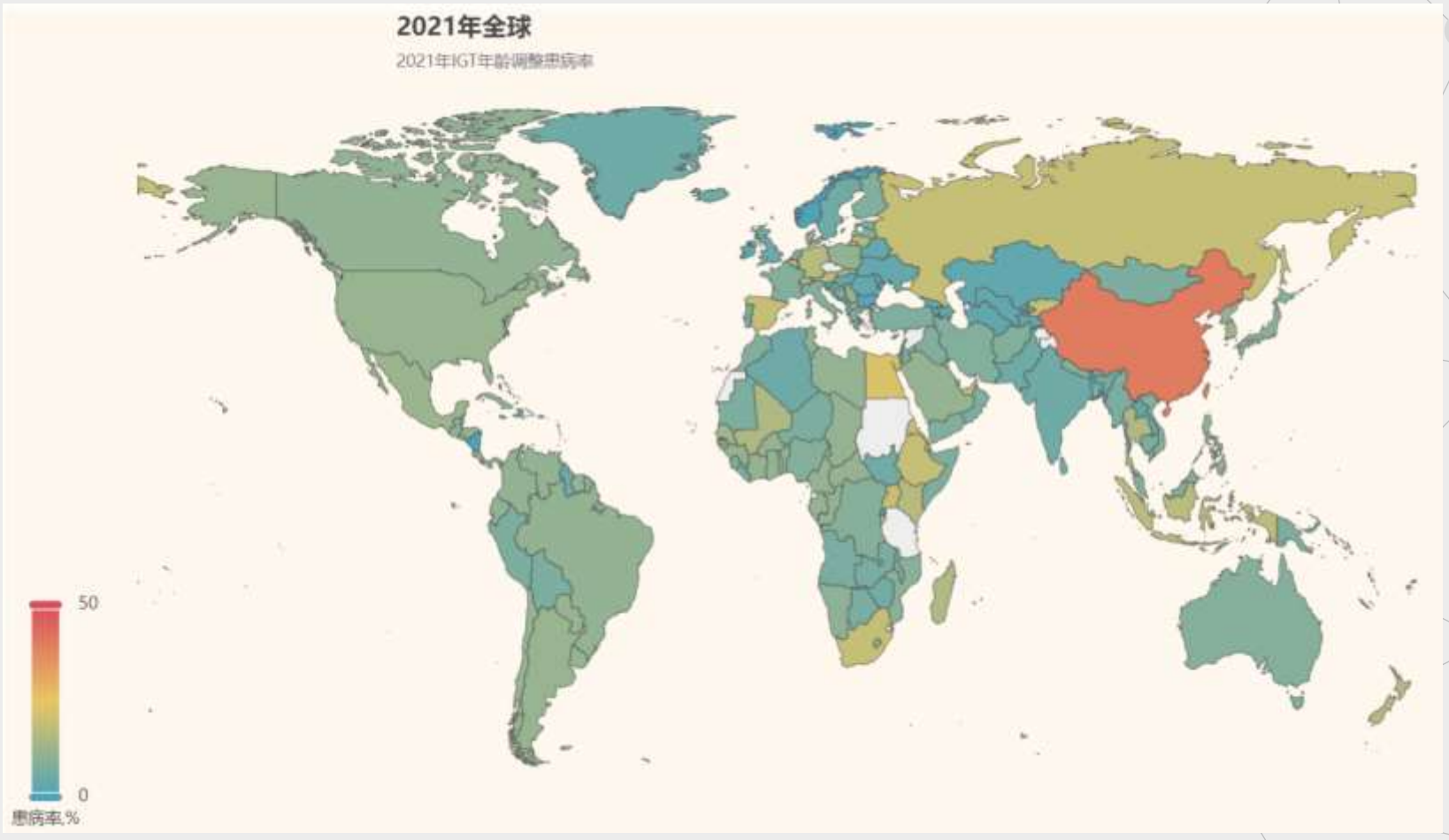
**绘制地图，并生成IGT_2021_prevalence.html。**

## B.2021年IGT患病率

**此地图展示了2021年各国IGT年龄调整患病率(%)**

- **年龄调整患病率是指在年龄段20~79岁人群中患IGT的人数占全国总人数的比例。许多疾病和健康状况的患病率会随着年龄的增长而增加。此比率可以消除年龄结构差异的影响，使得不同群体的患病率可以在同一标准下进行比较。**

- **根据示例图，颜色代表了数值的高低，可以直观得出异常数据值：中国关于此类型糖尿病患病人数比率最多，为40.4%**




```
MapWord(data['Country'],data['IGTPrevalence_2021'],"2021年全球","2021年IGT年龄调整患病率",'患病率,%',pieces,"IGT_2021_prevalence.html")
```

# 01 MAP

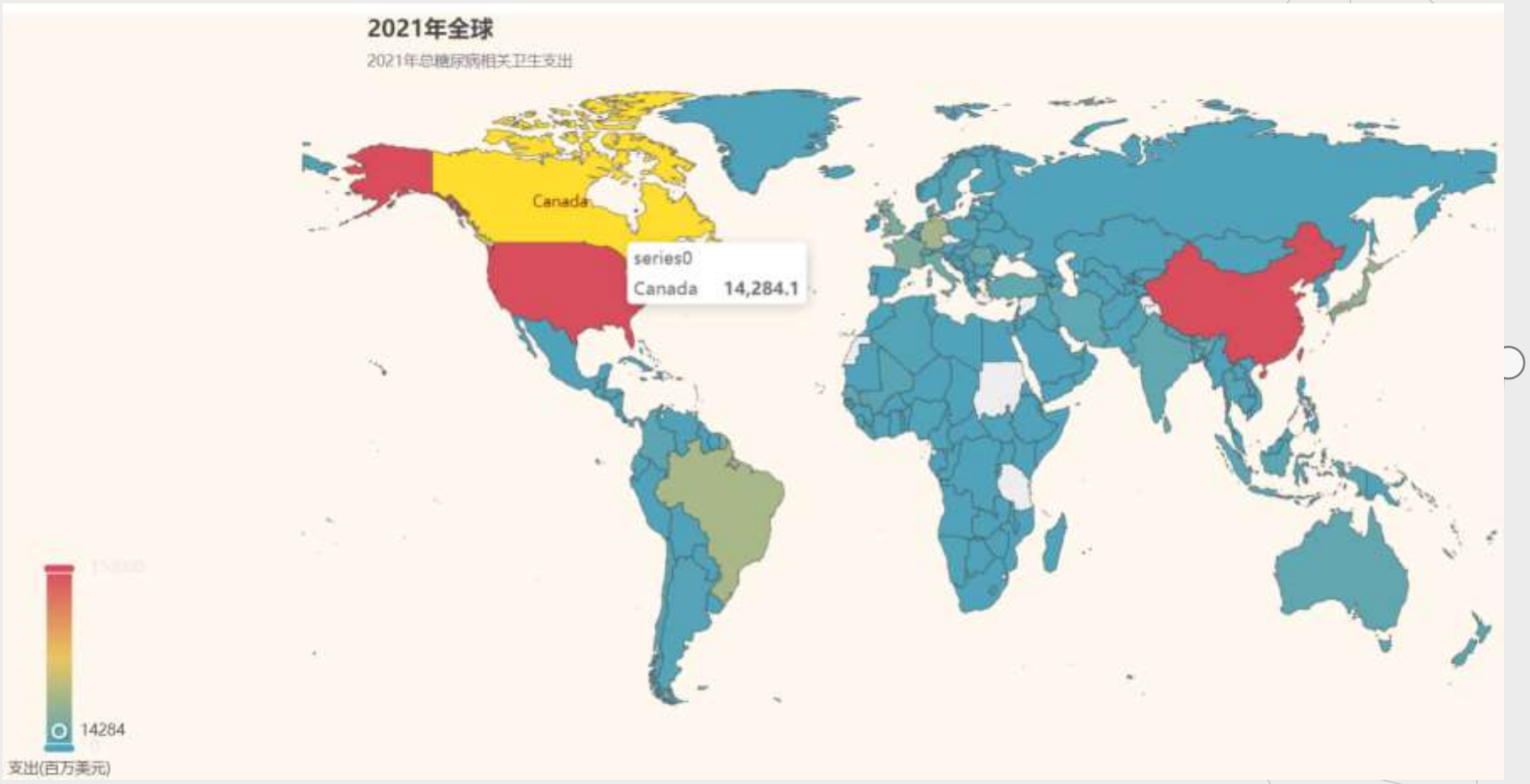
**绘制地图，并生成MONEY_total_2021.html。**

## C.2021年总糖尿病相关卫生支出

**此地图展示了2021年总糖尿病相关卫生支出(百万美元)**

- **如图中举例，加拿大在2021年关于此疾病相关总投入约142亿美元**

- **根据示例图，颜色代表了相关支出的多少，可以直观得出异常数据值：中国、美国在糖尿病相关卫生支出在全球明显最多，具体数值可以在html页面中得出。**




```
MapWord(data['Country'],data['Money20-79_2021'],"2021年全球","2021年总糖尿病相关卫生支出",'支出(百万美元)',pieces,"MONEY_total_2021.html")
```

# 02 折线统计图

#Change('','China')
##type：类型：①''：年龄调整比较患病率②'IFG'：IGT③'IGT':IGT
##Country--国家名称：'China'

## 绘制函数

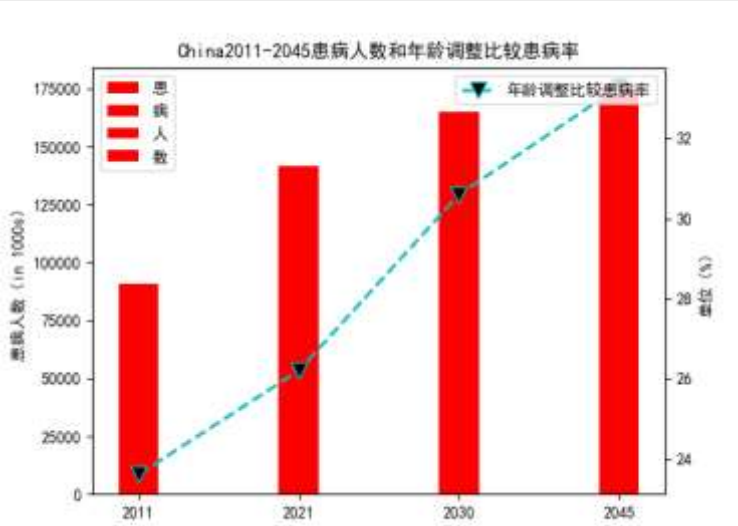
绘制折线统计图优势：

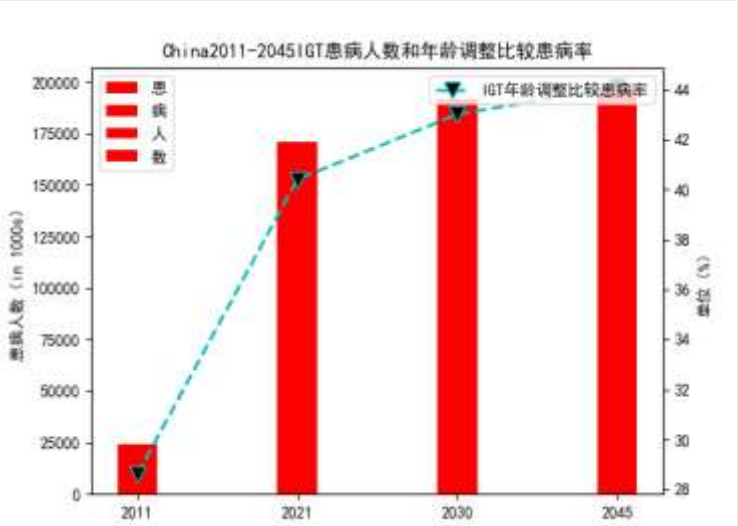
- **明显清晰：** 清晰地体现出糖尿病患病率与患病人数随时间的变化。
- **容易对比：** 选择不同国家，横轴为时间，纵轴可以选择三种数据进行绘图进行对比观察：
  - 同一国家两种疾病数据的差别
  - 不同国家同一疾病数据的差别
- **易得出结论：** 通过观察糖尿病患病率的折线图，可以评估过去实施的公共卫生政策和医疗干预措施的效果。

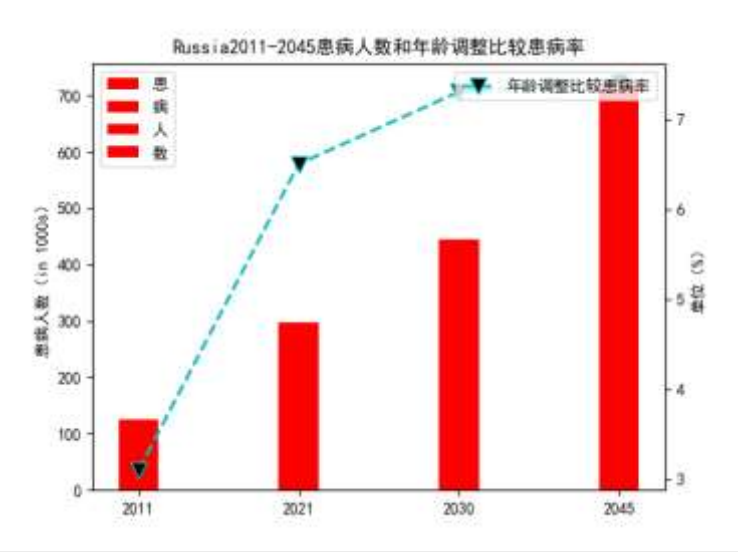
```
def Change(type,Country):
    data=pd.read_csv('dataAfterClean.csv')
    row_data=data[data['Country']==Country]
    print(row_data.iloc[0])
    datas=[]
    if type=='':
        datas=row_data.iloc[0].tolist()[1:9]
    elif type=='IGT':
        datas=row_data.iloc[0].tolist()[9:17]
    elif type=='IFG':
        datas= row_data.iloc[0].tolist()[17:25]
    else:
        print('error01')
    print(datas)
    rcParams['font.family']=rcParams['font.sans-serif']='SimHei'
    ax=plt.figure().add_subplot()
    labels=[2011,2021,2030,2045]
    cordx=range(len(labels))#x轴刻度的位置
    F1=ax.bar(x=cordx,height=datas[0:4],width=0.25,color='red')
    ax.legend(F1,'患病人数')
    ax.set_ylabel("患病人数（in 1000s）")
    ax.set_xticks(cordx)
    ax.set_xticklabels(labels)
    print(len(labels))
    ax.set_title(Country+"2011-2045"+type+"患病人数和年龄调整比较患病率")
```

# 02 折线统计图

取前24列数据，对中国、俄罗斯的患病率与患病人数、患病率进行对比。





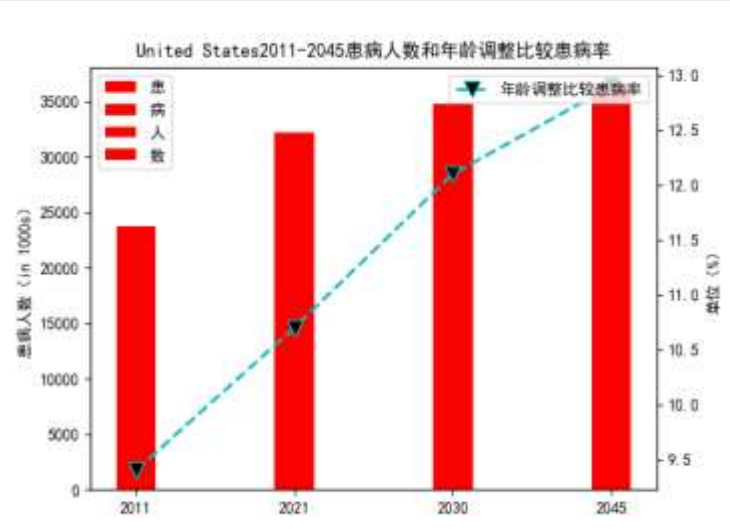
**分析图表，得出结论并分析原因**

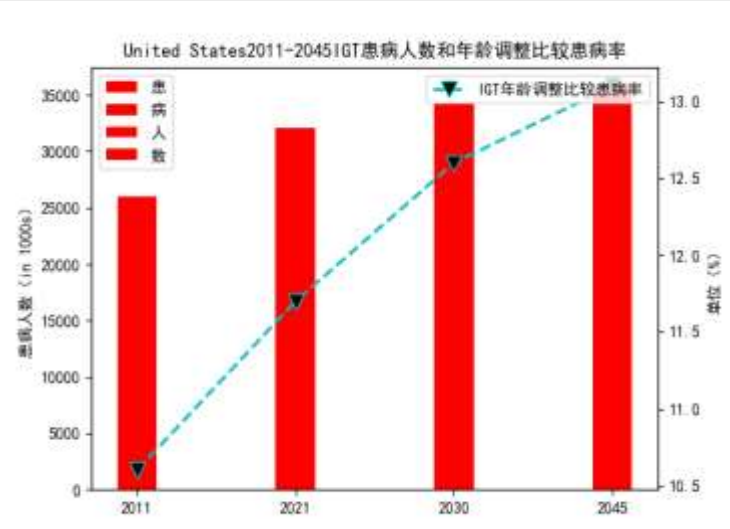
**中国患病人数、患病率都比俄罗斯高**，这种现象可能由多种因素共同导致：

- **人口结构**：俄罗斯总人口数量少于中国，糖尿病患病人数相对较低；中国的人口老龄化现象比俄罗斯更为严重，老年人胰岛素敏感性降低，是患糖尿病的高风险人群。
- **饮食文化不同**：中国传统饮食可能含有更高的碳水化合物，而俄罗斯饮食偏向高脂和高热量的食物。
- **经济发展水平**：中国的经济发展速度较快，城市化进程可能导致人们生活方式的快速变化，比如减少体力活动和增加不健康饮食习惯。
- **医疗资源分布**：俄罗斯的医疗资源分布相对均匀，而中国则可能存在城乡差异，这可能影响糖尿病等慢性病的诊断和管理。

# 02 折线统计图

取前24列数据，对美国、印度的患病率与患病人数、患病率进行对比。






**分析图表，得出结论并分析原因**

**相比美国，印度人口基数庞大，因此患病人数较多、两国患病率差别不大**，这种现象可能由多种因素共同导致：

- **人口结构**：印度人口超过13亿，而美国大约为3.3亿。即使糖尿病患病率较低，印度庞大的人口基数也会导致糖尿病患者总数很高。
- **环境因素**：环境污染、水质问题和不良的生活环境可能对印度人的健康产生负面影响，进而提高糖尿病的患病率。
- **药物获取**：美国的药品价格较高，尤其是胰岛素和其他糖尿病药物；而印度则生产大量**廉价仿制药**，使得药物成本更低。
- **技术水平**：美国在糖尿病的诊疗技术上通常处于领先地位，包括使用先进的胰岛素泵和血糖监测系统；而印度可能在这些高端技术的普及上存在滞后。

# 05

# 建 模 与 预 测

- 因果关系分析：利用回归分析（线性回归）来探究多个预测变量对结果变量的影响。
  - 热力图
  - 散点图
- 预测模型：使用机器学习方法（随机森林、梯度增强机）来预测未来趋势。

# 01 热力图

将相关系数矩阵以热力图形式可视化

## 热力图绘制函数

```
# 将相关系数矩阵以热力图的形式可视化
def ReLiShow(cols):
    cm = np.corrcoef(df[cols].values.T)
    hm= sns.heatmap(cm, cbar=True, square=True, fmt='.2f',
                    annot=True, annot_kws={'size': 15}, yticklabels=cols,
                xticklabels=cols)
    plt.suptitle(cols[0], color='red', fontsize=20)
    plt.show()
#   cbar=True 表示显示颜色条，square=True 表示将热力图的宽高设置为相等，
#   annot_kws={'size':15} 表示热力图上的数值字体大小为15
```

热力图（Heatmap）是一种数据可视化技术，它通过颜色的深浅来表示数据的密度或者强度。

在此处关于糖尿病的研究中，这种可视化方法有助于理解不同变量是如何相互作用并可能影响糖尿病的发展和管理的。

**优点：**

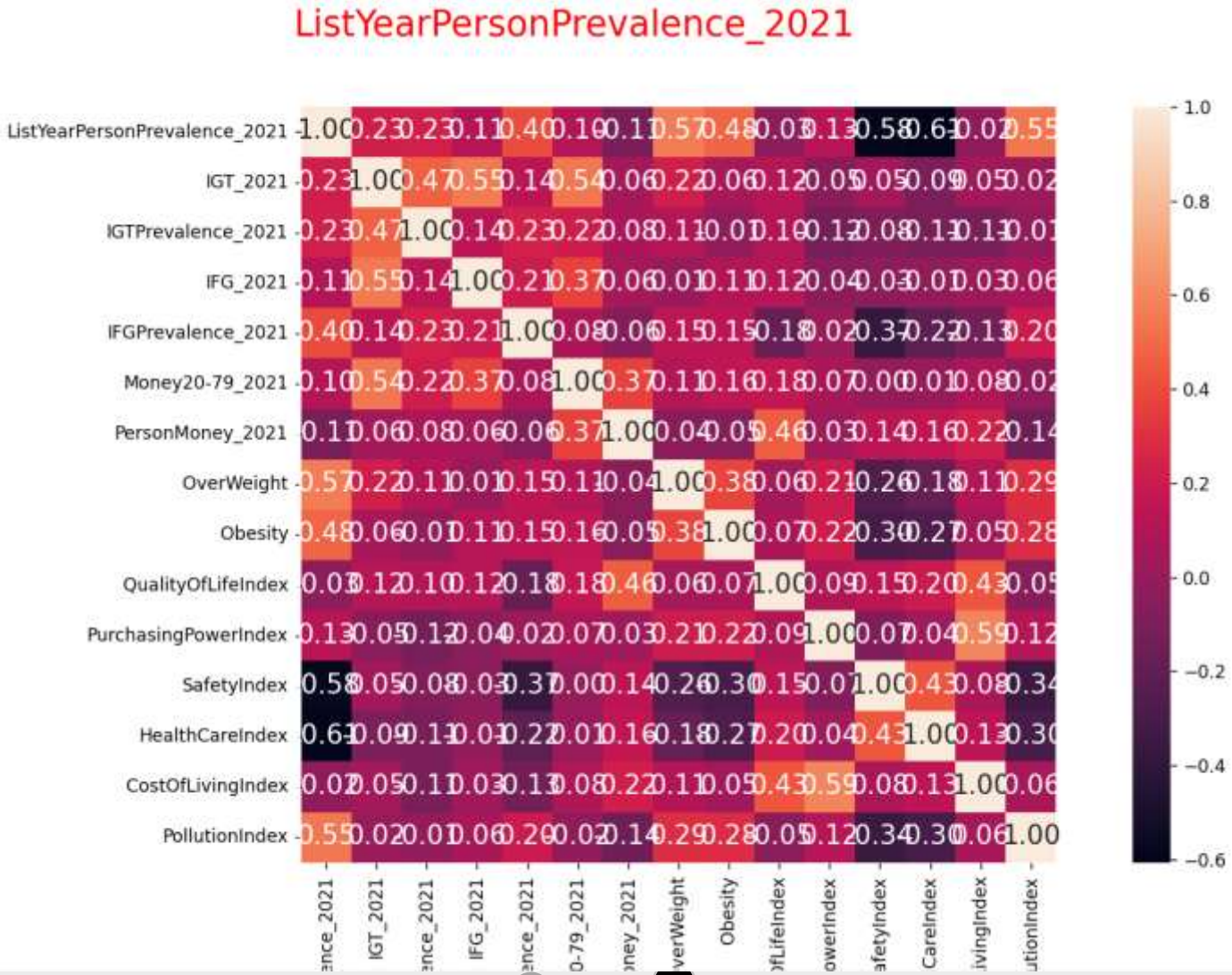
- **直观性：**热力图可以直观地显示出变量之间的相关性强度和方向。通过颜色的深浅，一眼就能看出哪些变量之间存在强相关或弱相关。
- **全面性：**热力图能够同时展示多个变量之间的相互关系，对于多变量分析特别有用。
- **利用空间直觉：**用二维空间展现数据，使得相关性的展示更符合人的空间直觉，有助于捕捉全局模式。
- **突出显示模式：**明显的颜色变化可以突出显示那些具有显著正相关或负相关的变量对，从而帮助研究人员识别关键的关联性。

# 01 热力图

ListYearPersonPrevalence_2021与其的相关系数

```
#(二)ListYearPersonPrevalence_2021与其的相关系数，
print('(二)ListYearPersonPrevalence_2021与其的相关系数')
RelaShow(cols1)
```

FIGURE 02




此热力图表示了2021年各因素数值与2021年患病率的相关系数。

与部分因素有明显的正负相关性：

- 与肥胖指数有**0.48的正相关性**。
- 与超重率有**0.57的正相关性**。
- 与社会安全系数有**0.58的负相关性**，高的社会安全系数会对居民的生活质量和健康状况产生积极影响。
- 与HealthCareIndex（卫生保健指数）有**0.61的负相关性**，这个指数越高通常意味着该地区的医疗卫生服务更好。
- 与污染指数有**0.55的正相关性**。
- 可见这些因素有一定几率影响患病率，但并不是很强的线性关系。

# 02 散点图

使用二百多个国家数据，根据热力图选择对患病率相关性较高的五个因素在散点图中进行组合，观察趋势。

```
def SanDianShow(type,X,Y,function):
        X1=[]
        if type==0:
                for i in X:
                        if i>10000:
                                X1.append(10000)
                        else:
                                X1.append(i)
        elif type==1:
                X1=X
        else:
                print("Error")
        plt.scatter(x=X1,  # 横坐标
            y=Y,  # 纵坐标
            c='red',  # 点的颜色
            )
#(四)ListYearPersonPrevalence_2021与其相关系数高的参数的散点图
print('(四)ListYearPersonPrevalence_2021与其相关系数高的参数的散点图')
for i in range(5):
    plt.subplot(321+i)
    plt.title(ListYearPersonPrevalence_2021_cols[i+1],color='green',fontsize=10)
    SanDianShow(1,df[ListYearPersonPrevalence_2021_cols[0]],
                df[ListYearPersonPrevalence_2021_cols[i+1]],ListYearPersonPrevalence_2021_cols[i+1])
plt.suptitle(ListYearPersonPrevalence_2021_cols[0],color='red',fontsize=20)
plt.show()
```

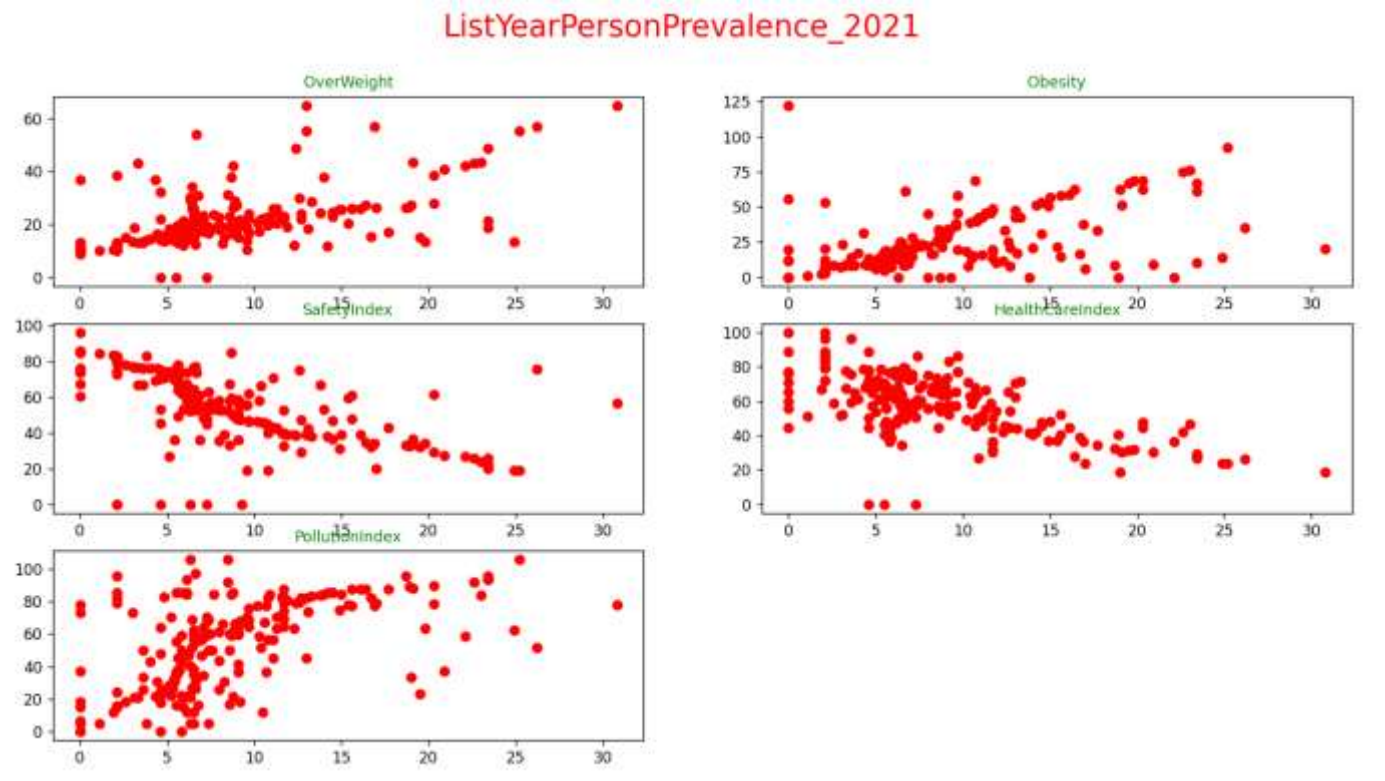
## 散点图绘制函数

散点图展示了两个变量之间关系，每个数据点表示这两个变量的一个组合。

- 上升趋势：
  - 如果散点图呈现出明显的从左下角到右上角的斜向上的线性或其他形状的趋势，那么可以说随着一个变量的增加，另一个变量也倾向于增加。这表明两个变量之间存在正相关性。
- 下降趋势：
  - 相反地，如果散点图中的点从右上角延伸到左下角，那么可以说随着一个变量的增加，另一个变量倾向于减少。这表明两个变量之间呈现负相关性。

# 02 散点图

ListYearPersonPrevalence_2021与其相关系数高的参数的散点图。




## FIGURE 02
## 结论分析

根据散点图可见，各图都有明显的上升或下降趋势。但**存在不均匀分布和偏离线性曲线的离散点**。

- 糖尿病患病率与超重、肥胖有正相关关系，但有少数离散点在曲线外，可能为特殊情况。
- 安全系数与健康指数都呈现负相关性。
- 污染指数与患病率呈现正相关以上的关系，表现为在污染指数较高的国家，患病率几乎没有偏低的情况。可见污染越重，患病可能性越高。

根据散点图，可以验证前面热力图的相关性会受到的影响因素：因为部分国家值为空白，产生较多0值，计算相关性有影响。

# 03 梯度提升回归器（GBR）预测

使用梯度提升回归模型（Gradient Boosting Regressor）来训练数据并进行预测，通过比较预测结果和实际结果来评估模型的效果。

```
data = pd.read_csv('dataAfterClean.csv')
# 选择特征
features = ['OverWeight', 'Obesity', 'SafetyIndex', 'HealthCareIndex', 'PollutionIndex']
target = 'ListYearPersonPrevalence_2021'    # 想要预测的目标变量。
# 删除缺失值
data.dropna(subset=features + [target], inplace=True)
# 划分数据集为训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(
    data[features], data[target], test_size=0.2, random_state=42)
```

```
# 模型
gbr = GradientBoostingRegressor()
# 训练模型
gbr.fit(X_train, y_train)
# 进行预测
predictions = gbr.predict(X_test)  # 特征 测试集
```

```
# 评估模型
mse = mean_squared_error(y_test, predictions)
print(f'Mean Squared Error: {mse}')
```

## 训练GBR模型函数

**通过迭代方式构建一系列简单模型（通常是决策树），并将它们结合起来创建一个更强大的预测模型。每个新模型的目的是修正前一个模型的误差，从而逐步提高整体预测性能。**

**# test_size=0.2：**这个参数意味着**将数据集按照80/20的比例划分为训练集和测试集**,测试集将包含原始数据集的20%。

**#gbr.fit(X_train，y_train)：**这行代码使用训练集的特征X_train和目标值y_train来训练模型。在训练阶段，模型会尝试找到特征和目标之间的关系，并调整其参数以最小化预测值和真实值之间的差异。

**#predictions = gbr.predict(X_test)：**使用训练好的模型gbr对测试集X_test进行预测。

**#评估模型：**将模型的预测结果predictions与实际结果y_test进行比较，计算均方误差（Mean Squared Error，MSE）以评估模型的性能。

# 03 梯度提升回归器（GBR）预测

性能分析与预测。

```
Mean Squared Error: 7.669896264801103
```

```python
# 有一个新的数据点，包含上述五个因素值
ListYearPersonPrevalence_2021_YuCe = [22, 42, 43, 29, 84]

# 使用训练好的GBR模型对该数据点进行预测
prediction = gbr.predict([ListYearPersonPrevalence_2021_YuCe])

# 打印出预测结果
print("预测的ListYearPersonPrevalence_2021值:", prediction)
```

```
预测的ListYearPersonPrevalence_2021值: [13.24134172]
```

| ListYearPersonPrevalence_2021 |
| --- |
| 10.9 |

## 结果分析

**#评估模型：** MSE值为**7.66**，模型性能一般。

**#实际值预测：** 将某一个国家的五个因素值输入后进行预测，预测的患病率为13.24%，靠近均值，实际患病率为10.9%，准确度不高。

# 04 随机森林

使用随机森林分类器对提供的数据进行分类，并且展示随机森林模型中各个特征的重要性。

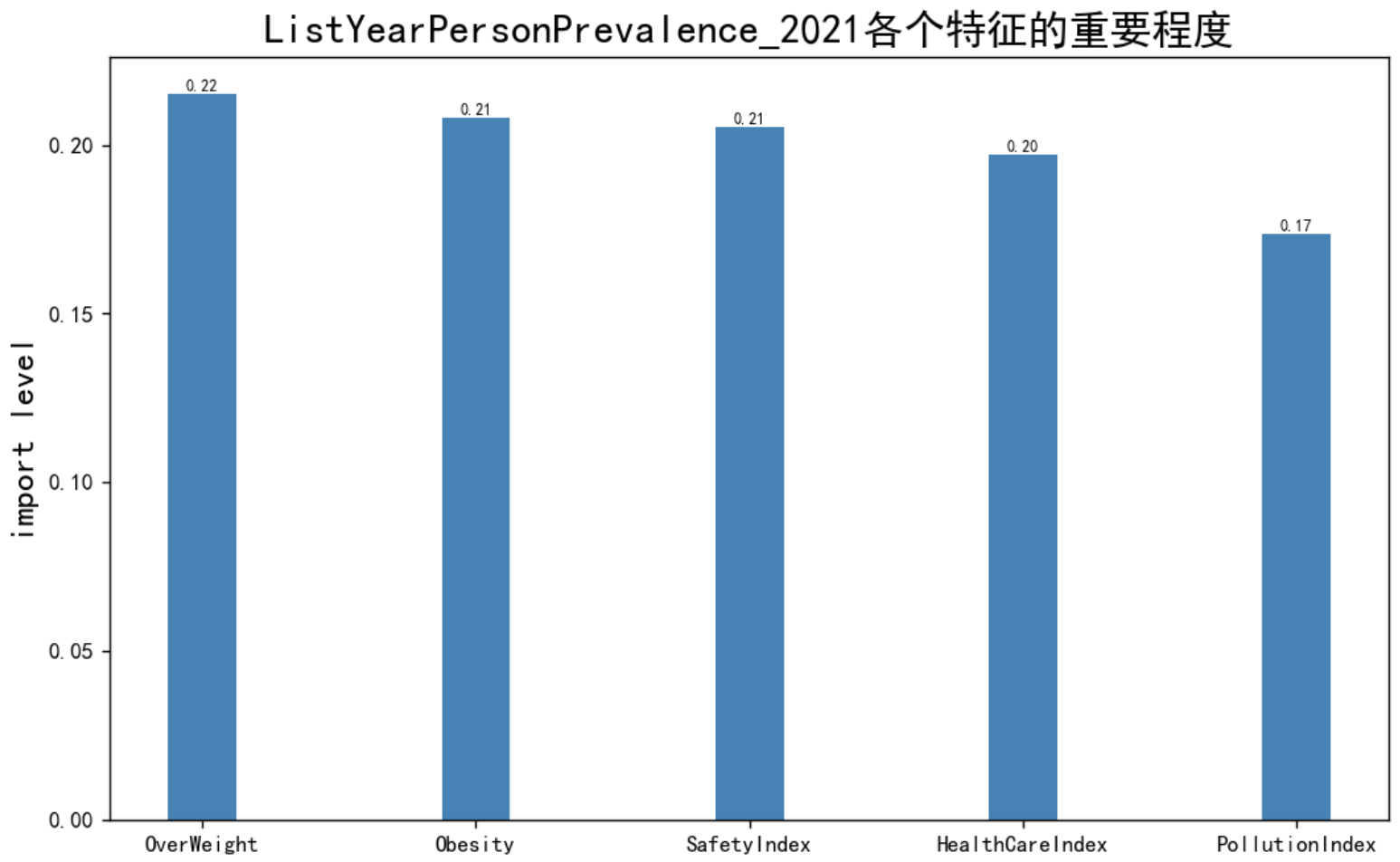
```
data = pd.read_csv('dataAfterClean.csv')
# 选择特征
features = ['OverWeight', 'Obesity', 'SafetyIndex', 'HealthCareIndex', 'PollutionIndex']
target = 'ListYearPersonPrevalence_2021'
# 删除缺失值
data.dropna(subset=features + [target], inplace=True)
# 划分数据集为训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(
    data[features], data[target], test_size=0.2, random_state=42)
# 创建随机森林回归器实例
forest = RandomForestRegressor(n_estimators=100, random_state=0, n_jobs=-1)  # 一般n_estima
# 训练模型
forest.fit(X_train, y_train)
# 使用模型对测试集进行预测
y_pred = forest.predict(X_test)
# 计算测试集上的MSE
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```

## 随机森林函数

功能：

- 使用随机森林分类器训练模型， 对提供的数据进行分类，评估特征的重要性。显示一个条形图，其中包含每个特征的重要度分数。
- 与GBR部分参数设置相同，如划分比例。
- 对传入的待预测新实例进行预测，并返回预测结果。

# 04 随机森林




## 各个特征重要程度

**特征重要程度Importance：**

有助于解释模型的预测。知道哪些特征对模型影响最大，可以帮助我们理解模型是如何做出预测的，理解模型的透明度和可信度。

经过随机森林模型的分析，**我们发现五个因素的特征对预测结果都很重要**，这些因素在模型预测中的贡献是相对均衡的，发挥了同等重要的作用。

# 04 随机森林—预测

```
Mean Squared Error: 6.299096244444433
```

```
#(六)使用随机森林对ListYearPersonPrevalence_2021的数据进行预测和得出其因素的重要性统计图
#其中的5个数据分别为：'OverWeight','Obesity','SafetyIndex','HealthCareIndex','PollutionIndex'
ListYearPersonPrevalence_2021_YuCe=[22,42,43,29,84]
print('(六)使用随机森林对ListYearPersonPrevalence_2021的数据进行预测和得出其因素的重要性统计图')
a2=SuiJiSenLin(ListYearPersonPrevalence_2021_cols,ListYearPersonPrevalence_2021_YuCe)
print("根据随机森林得出年龄调整比较患病率:",a2)
```

```
根据随机森林得出年龄调整比较患病率： 10.9
```

| ListYearPersonPrevalence_2021 |
| --- |
| 10.9 |

## 结果分析

**#MSE性能估计：计算所得MSE值为6.3，相比GBR误差较小。**

**#实际值预测：**将某一个国家的五个因素值输入后进行预测，预测的患病率与实际患病率相近，都为10.9%，准确度优于GBR模型。

# 展示结束！

## 感 谢 大 家 的 观 看

16组—伍鑫 乔妍妍 杨若妍