对环境问题和气温变化的分析

一．问题的引出

近年来，环境问题变得日渐严峻，一些关于环境指标新的名词出现在人们的视野中：“全球变暖”、“AQI 指数”、“大气污染物”，这些词语都是当下人们对于环境恶劣变化开始关注的衡量指标。

记得从 2015 年 2 月 28 日，柴静推出其自费拍摄的雾霾深度调查《穹顶之下》开始，人们对于环境变化才开始关注起来，当人们终于开始抬头望向天空，他们才发现，原来的晴空万里已经被浓烟所笼罩，原来星星遍布的夜晚已经被霓虹灯隐藏的难以寻觅。

现代化的工业不仅仅带来了 AQI 值的普遍升高，也带来了全球变暖的严峻问题。本可视化数据着重于对于环境污染指标的分析和全球变暖的预测。

二．数据来源

本数据从两个方面获取：

（1）引用的‘GlobalLandTemperaturesByCountry.csv’文件从 kaggle 数据网站上获取，其中包含 1743 年到 2013 年各国历年温度情况，ChinaTemperatures.csv 为其中节选，便于后续对中国情况的单独分析。

（2）引用的‘城市.XLSX’文件从 https://www.aqistudy.cn/historydata/上获取，截止 2020 年 12 月，上面可以获取 2013-12 到 2020-12 的各个城市污染指标数据。

三．数据分析可视化

1.预处理

（1）所用库的导入

```
from pyecharts.charts import *
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
import random
import pandas as pd
from matplotlib.pyplot import *
import pyecharts.options as opts
import matplotlib.pyplot as plt
import xlrd
```

（2）颜色函数定义

```
# 随机获取颜色
def randomcolor(kind):
```

```
    colors = []
    for i in range(kind):
        colArr = ['1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B',
'C', 'D', 'E', 'F']
        color = ""
        for i in range(6):
            color += colArr[random.randint(0, 14)]
        colors.append("#" + color)
    return colors
```

（3）读取各个数据

```
# 预定义和文件读取
all_country = pd.read_csv("GlobalLandTemperaturesByCountry.csv")
China_date = pd.read_csv('ChinaTemperatures.csv')
name = list(pd.read_excel('城市.XLSX')['城市'].drop_duplicates())
data = xlrd.open_workbook('城市.XLSX')
```

2.城市污染分析

（1）数据处理

```
table = data.sheets()[0]
dic1 = {k: [] for k in name}
for i in range(1, table.nrows):
    x = table.row_values(i)
    dic1[x[0]].append((x[1], x[2], x[5], x[6], x[7], x[8], x[9], x[10]))
# 城市作为键，日期和 AQI 作为值
lisx = list(range(0, 85))
lisy = []
lisdate = []
```

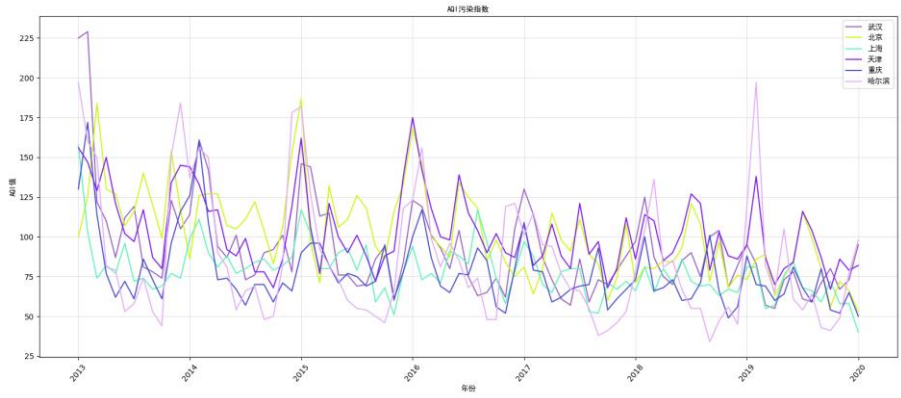
（2）绘制六大城市 AQI 污染值数对比

```
plt.figure(figsize=(20,8),dpi=100)
for i in range(2013, 2021):
    lisdate.append('{}'.format(i))
color_series = randomcolor(8)
color_number = 0
for j in name:
    for i in lisx:
        lisy.append(dic1[j][i][1])
    plt.plot(lisx, lisy, color=color_series[color_number], label=j)
    lisy = []  # 清空为下次作图
    color_number = color_number + 1  # 变换随机颜色

plt.xticks(list(range(0, 85, 12)), lisdate, rotation=50)
```

```
plt.title("AQI 污染指数", fontproperties='simHei')
plt.xlabel("年份", fontproperties='simHei')
plt.ylabel("AQI 值", fontproperties='simHei')
plt.grid(alpha=0.4)
plt.legend(prop="simHei", loc='upper right')
plt.savefig('./AQI 污染值数城市对比.png')
print('AQI 污染值数城市对比已生成')
# plt.show()
plt.close()
```

生成图片如下：

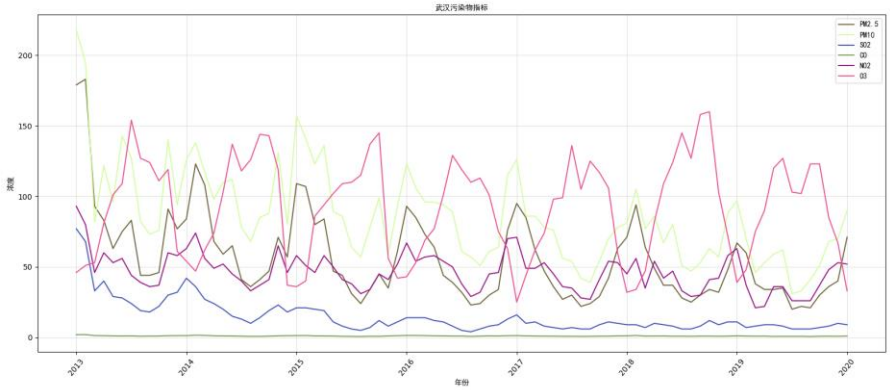


分析：由图中发现，在六个中国主要城市之中，各个城市 AQI 大于 50 的时间几乎占据七年中 90%以上，在 100 以下的良好天气也几乎只占 40% 。但是在 2017 年开始，AQI 在 125 以上天数明显减少，当时人们对于环境问题开始逐渐关注起来，除了哈尔滨在 2019 年有异常反弹外，其余五个城市基本有较好改善。

（3）绘制武汉污染物指标

```
color_number = 0
name_lable = ['PM2.5', 'PM10', 'SO2', 'CO', 'NO2', 'O3']
plt.figure(figsize=(20,8),dpi=100)
for j in range(2, 8):
    for i in lisx:
        lisy.append(dic1["武汉"][i][j])
    plt.plot(lisx, lisy, color=color_series[color_number],
label=name_lable[j-2])
    lisy = []  # 清空为下次作图
    color_number = color_number + 1  # 变换随机颜色
plt.xticks(list(range(0, 85, 12)), lisdate, rotation=50)
plt.title("武汉污染物指标", fontproperties='simHei')
plt.xlabel("年份", fontproperties='simHei')
plt.ylabel("浓度", fontproperties='simHei')
plt.grid(alpha=0.4)
plt.legend(prop="simHei", loc='upper right')
plt.savefig('./武汉污染物指标.png')
```

```
print('武汉污染物指标已生成')
# plt.show()
plt.close()
```



分析：由图中发现，在七年之间，CO 一直维持较小的值没有太多波动，SO2 的情况明显改善，由 2013 的 75 到 2020 趋于平稳的 10 左右，其余指标在每年之中波动，除 PM10 在波动中逐年递减，其余没有较大变化。

3. 全球温度分析

（1）数据处理

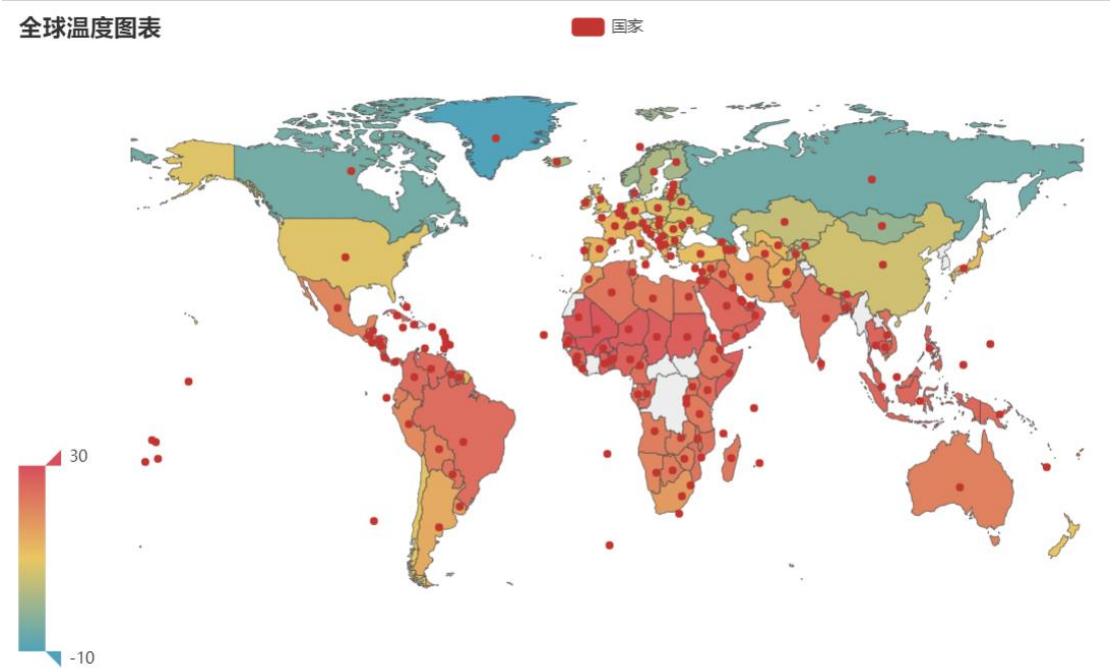
```
#处理数据
cou_tem = all_country.groupby("Country")['AverageTemperature'].mean()
# 国家和温度平均值打包
cou_tem = cou_tem.dropna()  # 删除缺省
country = list(cou_tem.index)  # 将国家转化列表
temperature = list(cou_tem.values)  # 将温度转化列表
zip_country_temperature1 = zip(country, temperature)  # 打包成元组好排序
zip_country_temperature2 = zip(country, temperature)
top_ten_country = sorted(zip_country_temperature1, key=lambda tup:
tup[1], reverse=True)[0:10]  # 打包成元组好排序,取最高温前十个
low_ten_country = sorted(zip_country_temperature2, key=lambda tup:
tup[1], reverse=False)[0:5]  # 打包成元组好排序,取最高温前十个
all_country['dt_year'] = pd.to_datetime(all_country['dt']).dt.year  #
统一时间格式分组年份,世界
all_country['dt_month'] = pd.to_datetime(all_country['dt']).dt.month
# 统一时间格式分组日期
China_date['dt_year'] = pd.to_datetime(China_date['dt']).dt.year  # 统
一时间格式分组年份，中国
year_tem =
all_country.groupby(['dt_year'])['AverageTemperature'].mean()  # 世界
年份和温度平均值打包
year_tem = year_tem.dropna()  # 删除缺省
```

```
year_tem_China =
China_date.groupby(['dt_year'])['AverageTemperature'].mean()  # 中国年份和温度平均值打包
year_tem_China = year_tem_China.dropna()  # 删除缺省
year = list(year_tem.index)  # 将年份转化列表
```

（2）绘制温度地图

```
# 绘制温度地图
map = Map()
map.add("国家", [list(i) for i in zip(country, temperature)], 'world')
map.set_series_opts(label_opts=opts.LabelOpts(is_show=False))
map.set_global_opts(
    title_opts=opts.TitleOpts(title="全球温度图表"),
    visualmap_opts=opts.VisualMapOpts(max_=30, min_=-10)  # 最高温不超过30，设置为30
)
map.render('全球温度地图.html')
print("全球温度地图已生成！！")
```

绘制出来为 html 格式无法上传图片，请查看附件中‘全球温度地图.html’，下图为网页截图



分析：在本图片中，我们可以发现，除了缺失数据国家以外，各个国家的温度的确较为符合其经纬度的特征温度。

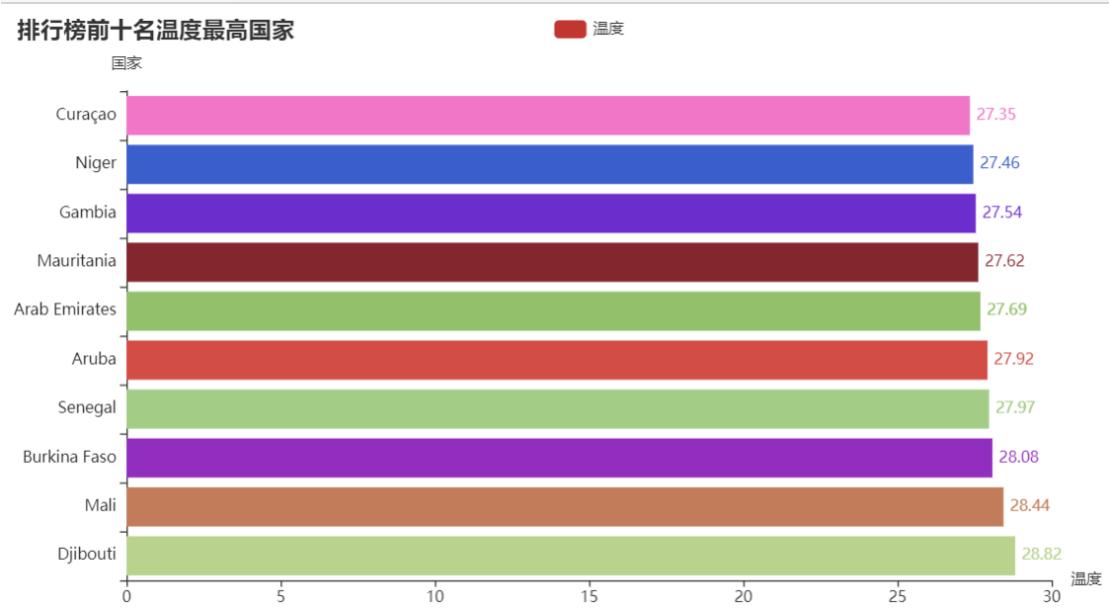
（3）绘制最高温度前十位国家的平均温度情况（条形图）

```
country = [x[0] for x in top_ten_country]  # 前十个国家
temperature = [x[1] for x in top_ten_country]  # 前十个国家对应温度

x = []  # 空列表存温度
color_series = randomcolor(len(country))
for i in range(10):
    x.append(
        opts.BarItem(
            name=country[i],
            value=round(temperature[i], 2),
            itemstyle_opts=opts.ItemStyleOpts(color=color_series[i])  #
设置每根柱子的颜色
        )
    )
    # 绘制柱形图
bar = Bar()
bar.add_xaxis(country)
bar.add_yaxis(
    series_name='温度',
    y_axis=x,
    is_selected=True,
    label_opts=opts.LabelOpts(is_show=True))
bar.set_series_opts(label_opts=opts.LabelOpts(is_show=True,
position='right'))
bar.set_global_opts(
    title_opts=opts.TitleOpts(title="排行榜前十名温度最高国家"),
    tooltip_opts=opts.TooltipOpts(
        is_show=True,
        trigger="axis",
        axis_pointer_type="shadow"),

    xaxis_opts=opts.AxisOpts(name='温度'),
    yaxis_opts=opts.AxisOpts(name='国家'),
)
bar.reversal_axis()  # 转换为条形图，不然国家无法完全显示
bar.render("平均温度最高的十个国家.html")
print("平均温度最高的十个国家条形图已生成！！")
```

绘制出来为 html 格式无法上传图片，请查看附件中‘平均温度最高的十个国家.html’，下图为网页截图



分析：图中取出温度最高的十个国家，在分析全球变暖问题时候，温度较高的城市需要重点关注其变化，也需要着重从这几个城市之中找到并分析引起变暖的成因，不能只归结到他们地理位置的温度本身就较高这一原因

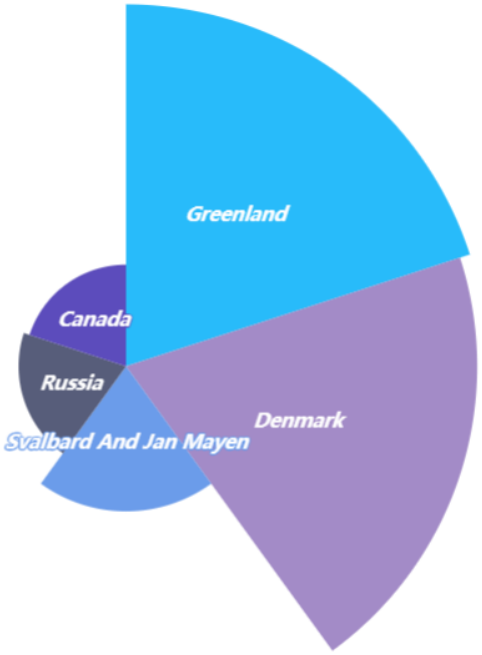
（4）绘制温度最低五个国家（玫瑰图）

```
country = [x[0] for x in low_ten_country]  # 后五个国家
temperature = [x[1] for x in low_ten_country]  # 后五个国家对应温度
# print(temperature)
color_series = randomcolor(len(country))
pie = Pie()
pie.add("", [list(i) for i in zip(country, temperature)], radius=['0%',
'25%'], center=['50%', '50%']
      , rosetype='area')
pie.set_global_opts(title_opts=opts.TitleOpts(title='温度最低五个国家'),
                    legend_opts=opts.LegendOpts(is_show=False)

                    )
pie.set_series_opts(label_opts=opts.LabelOpts(is_show=True,
position='inside', font_size=12
                                              , font_style='italic',
font_weight='bold',
                                              font_family='Microsoft YaHei'))
pie.set_colors(color_series)
pie.render('平均温度最低的五个国家.html')
print('平均温度最高的五个国家玫瑰图已生成！！')
```

绘制出来为 html 格式无法上传图片，请查看附件中‘平均温度最低的五个国家.html’，下图为网页截图

温度最低五个国家



分析：同理，在分析全球变暖问题时候，最低温度的几个城市也需要着重关注，因为全球变暖将导致这些冰川城市的冰川融化，分析全球变暖问题时候考虑这些国家的温度变化也显得十分重要

（5）绘制中国与全球年温度折线图

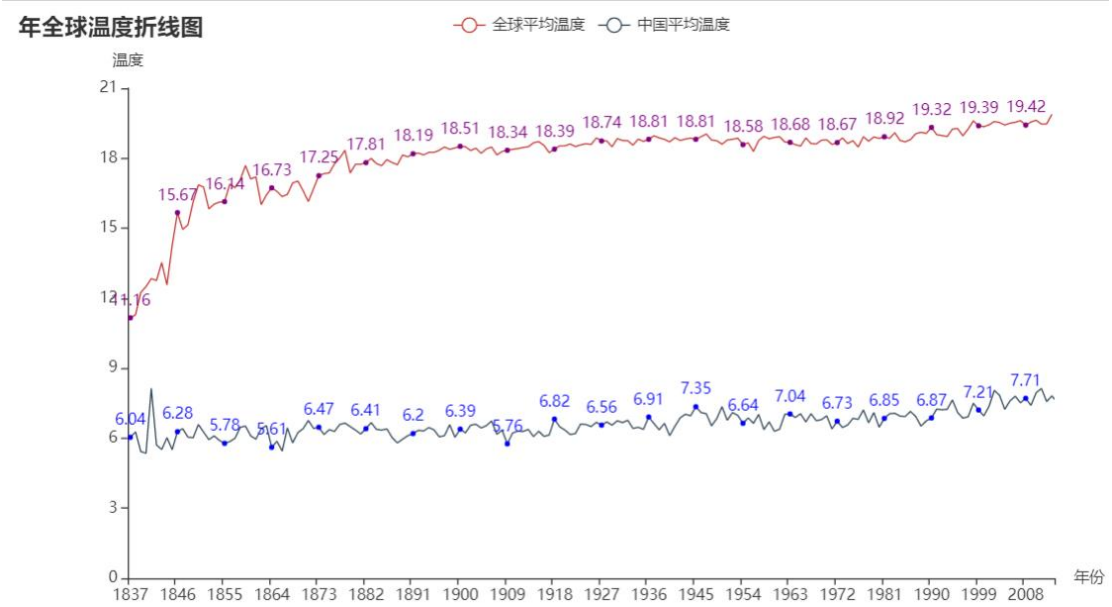
```
x, y = [], []  # 空列表存温度
for i in range(90, 267):  # 从1837到2013年
    x.append(
        opts.LineItem(
            name=year[i],
            value=round(temperature[i], 2),
            itemstyle_opts=opts.ItemStyleOpts(color='purple')
        )
    )
for j in range(0, 181):  # 从1837到2013年
    y.append(
        opts.LineItem(
            name=year_China[j],
            value=round(temperature_China[j], 2),
            itemstyle_opts=opts.ItemStyleOpts(color='blue')
        )
    )
line = Line()
line.add_xaxis(year[90:])
line.add_yaxis(
    series_name='全球平均温度',
```

```
    y_axis=x,
    is_selected=True,
    label_opts=opts.LabelOpts(is_show=True))
line.add_yaxis(
    series_name='中国平均温度',
    y_axis=y,
    is_selected=True,
    label_opts=opts.LabelOpts(is_show=True))
line.set_series_opts(label_opts=opts.LabelOpts(is_show=True))
line.set_global_opts(
    title_opts=opts.TitleOpts(title="年全球温度折线图"),
    tooltip_opts=opts.TooltipOpts(
        is_show=True,
        trigger="axis",
        axis_pointer_type="shadow"),

    xaxis_opts=opts.AxisOpts(name='年份'),
    yaxis_opts=opts.AxisOpts(name='温度'),
)

line.render("中国与全球年温度折线图.html")
print("全球温度折线图已生成！！")
# print(year)
```

绘制出来为 html 格式无法上传图片，请查看附件中‘中国与全球年温度折线图.html’，下图为网页截图



分析：图中显示了中国和全球温度变化的 1837 到 2013 年数据，可以明显观察到全球变暖的大趋势，虽然中国在全球变暖问题上趋势变换较为缓慢，但并不是松懈的理由，更要分析为何波动较大，以及趋向于 8° 如何缓解的问题思考

（6）绘制中国与全球预测温度图

```
模型预测分析线性和多项式分析
start = year.index(1900)  # 从 1900 年开始回归方程
start_China = year_China.index(1900)
year_start = year[start:]
year_start_China = year_China[start_China:]
year_start = [int(i) for i in year_start]  # 线性回归方程均为整数
year_start_China = [int(i) for i in year_start_China]
temp_start = temperature[start:]
temp_start_China = temperature_China[start_China:]
clf = LinearRegression()  # 线性回归分析
clf1 = LinearRegression()  # 后缀为 1 为中国
x = np.array(year_start).reshape(114, 1)
y = np.array(temp_start).reshape(114, 1)
x1 = np.array(year_start_China).reshape(114, 1)
y1 = np.array(temp_start_China).reshape(114, 1)
clf.fit(x, y)
clf1.fit(x1, y1)  # 一次分析

year_set_China = year_set = list(range(2014, 2036))  # 将预测的年份
predict_temp = clf.predict([[i] for i in year_set])  # 按线性回归方程估计计算未来温度
predict_temp_China = clf1.predict([[i] for i in year_set_China])  # 按线性回归方程估计计算中国未来温度
predict_temp_list = [i[0] for i in predict_temp]  # 化为一维列表
predict_temp_China_list = [i[0] for i in predict_temp_China]  # 化为一维列表

ploy = PolynomialFeatures(degree=2)  # 设置为 2 次项，多项式预测
x_ploy = ploy.fit_transform(x)
clf_ = LinearRegression()  # 设置二次线性实例
clf_.fit(x_ploy, y)
x_ployed = ploy.transform(x)
y_predict = clf_.predict(x_ployed)
predict_temp_two_list = [i[0] for i in y_predict]  # 化为一维列表

x, x1, x_dimensional = [], [], []  # 空列表存温度
for i in range(0, 22):
    x.append(
        opts.LineItem(
            name=year_set[i],
```

```
            value=round(predict_temp_list[i], 2),
            itemstyle_opts=opts.ItemStyleOpts(color='purple')
        )
    )
for i in range(0, 22):
    x1.append(
        opts.LineItem(
            name=year_set_China[i],
            value=round(predict_temp_China_list[i], 2),
            itemstyle_opts=opts.ItemStyleOpts(color='blue')
        )
    )
for i in range(0, 22):
    x_dimensional.append(
        opts.LineItem(
            name=year_set[i],
            value=round(predict_temp_two_list[i], 2),
            itemstyle_opts=opts.ItemStyleOpts(color='green')
        )
    )
line1 = Line()
line1.add_xaxis(year_set)
line1.add_yaxis(
    series_name='全球平均温度预测(一次)',
    y_axis=x,
    is_selected=True,
    label_opts=opts.LabelOpts(is_show=True))
line1.add_yaxis(
    series_name='中国平均温度预测(一次)',
    y_axis=x1,
    is_selected=True,
    label_opts=opts.LabelOpts(is_show=True))
line1.add_yaxis(
    series_name='全球平均温度预测(二次)',
    y_axis=x_dimensional,
    is_selected=True,
    label_opts=opts.LabelOpts(is_show=True))
line1.set_series_opts(label_opts=opts.LabelOpts(is_show=True))
line1.set_global_opts(
    title_opts=opts.TitleOpts(title="全球平均温度预测"),
    tooltip_opts=opts.TooltipOpts(
        is_show=True,
        trigger="axis",
        axis_pointer_type="shadow"),
```

```
    xaxis_opts=opts.AxisOpts(name='年份'),
    yaxis_opts=opts.AxisOpts(name='温度'),
)
line1.render("中国与全球预测温度图.html")
print("中国与全球预测温度图已生成！！")
```

绘制出来为 html 格式无法上传图片，请查看附件中‘中国与全球预测温度图.html’，下图为网页截图

分析：本图根据数据只有到 2013 为止的较为便捷利用的数据为回归样本，以线性回归方程以及多项式回归分别对 2014 到 2035 年的全球温度以及中国温度进行分析，并将预测样本中 2014 年到 2020 年与现在可获取数据进行对比可以发现，在预测模拟中，线性回归方程的全球温度预测模拟较为优秀，拟合程度高。也为人们敲响了警钟，如果按这个趋势的全球升温，不久人们将面临地球母亲的报复了，呼吁人们关注环境问题，关注全球变暖的问题。文明出行，为自己生活的家园做出自己的贡献。

（7）将上述表格统一到一个网页中

```
page = Page()
page.add(map)
page.add(pie)
page.add(bar)
page.add(line)
page.add(line1)

page.render('全球温度可视化汇总.html')
print("全球温度可视化汇总已生成！！")
```

绘制出来为 html 格式无法上传图片，请查看附件中‘全球温度可视化汇总.html’

四．结语

地球是我们赖以生存的地方，关注城市污染问题，关注全球变暖问题，就是在关注我们自身，关注我们以及我们后代的前程和未来！！！