民国南京城市历史地名交互可视化实例

1. 实验环境

1.1 安装 python3 及相关库

a) 下载地址：https://www.python.org/downloads/release/python-373/

b) 安装 python-3.8.0-amd64.exe，点击“Next”，直到结束。

c) 在安装过程中注意点击“Add Python 3.7 to PATH”。

d) 安装完成在命令行输入：python --version，得到如下输出：Python 3.8.0，Python3.8.0 安装成功

e) 本次实验采用了 python 库 geopandas、plotly，可以采用 pip 进行安装，命令如下 pip install geopandas plotly。

2. 数据集分析

2.1 数据下载

a) 首先打开数据集的网站：http://www.csdata.org/p/127/2/#dataset-profile。

b) 点击访问数据集。

点击后页面跳转下载数据，得到地名数据后，请用 7z 等解压工具解压，否则会出现乱码。文件夹地名数据下有 1909 地名数据，1927 地名数据，1937 地名数据三个文件夹，其中各文件夹下的.shp 是我们需要的目标数据集。

3. 可视化实例代码分析

3.1 导入依赖

```
import geopandas as gpd
import plotly.graph_objects as go
import plotly.io as pio
import plotly.offline as pyo
import plotly.express as px
```

3.2 读取文件

```
# 读取南京的 Shapefile 文件
city_shapefile_path = '..\\地名数据\\1909 年地名数据\\1909 年地名数据.shp'
city_shapefile_path1 = '..\\地名数据\\1927 年地名数据\\1927 年地名数据.shp'
city_shapefile_path2 = '..\\地名数据\\1937 年地名数据\\1937 年地名数据.shp'
# 替换为南京 Shapefile 文件的实际路径
nanjing_data = gpd.read_file(city_shapefile_path)
nanjing_data1 = gpd.read_file(city_shapefile_path1)
nanjing_data2 = gpd.read_file(city_shapefile_path2)
```

3.3 数据集的清洗及合并

首先三个数据集读取后发现列名不一致，列数也不一致，统一列名合并是非常重要的，数据质量很好没有空值和异常值，在这里不再做其他数据清洗工作。生成一列“时间”列作为标记，合并数据集后做区分。

```
nanjing_data=nanjing_data[['地名','一级分类分','精度坐标','纬度坐标','geometry']]
nanjing_data1.columns=['OBJECTID', '原图层编码', '地名', '一级分类分', '二级分类', '地址', '精度坐标', '纬度坐标', '图层',
```

```
        '数据来源', '类型', 'NEAR_FID', 'NEAR_DIST', 'geometry']
    nanjing_data1=nanjing_data1[['地名','一级分类分','精度坐标','纬度坐标
','geometry']]
    nanjing_data2.columns=['OBJECTID', '地名', '一级分类分', '二级分类', '
精度坐标', '纬度坐标', '分类', 'NEAR_FID',
        'NEAR_DIST', 'geometry']
    nanjing_data2=nanjing_data2[['地名','一级分类分','精度坐标','纬度坐标
','geometry']]

    nanjing_data['时间']='1909 年地名数据'
    nanjing_data1['时间']='1927 年地名数据'
    nanjing_data2['时间']='1937 年地名数据'
    nanjing_data3_1=nanjing_data.append(nanjing_data1)
    nanjing_data3=nanjing_data3_1.append(nanjing_data2)
    nanjing_data3=nanjing_data3.reset_index()
    nanjing_data3=nanjing_data3[[ '时间','地名', '一级分类分', '精度坐标', '
纬度坐标', 'geometry']]
```

3.4 可视化代码

可视化代码，主要分为读取数据，构建散点图，构建按钮，生成图例，添加交互细节，以及输出文档几部分组成。

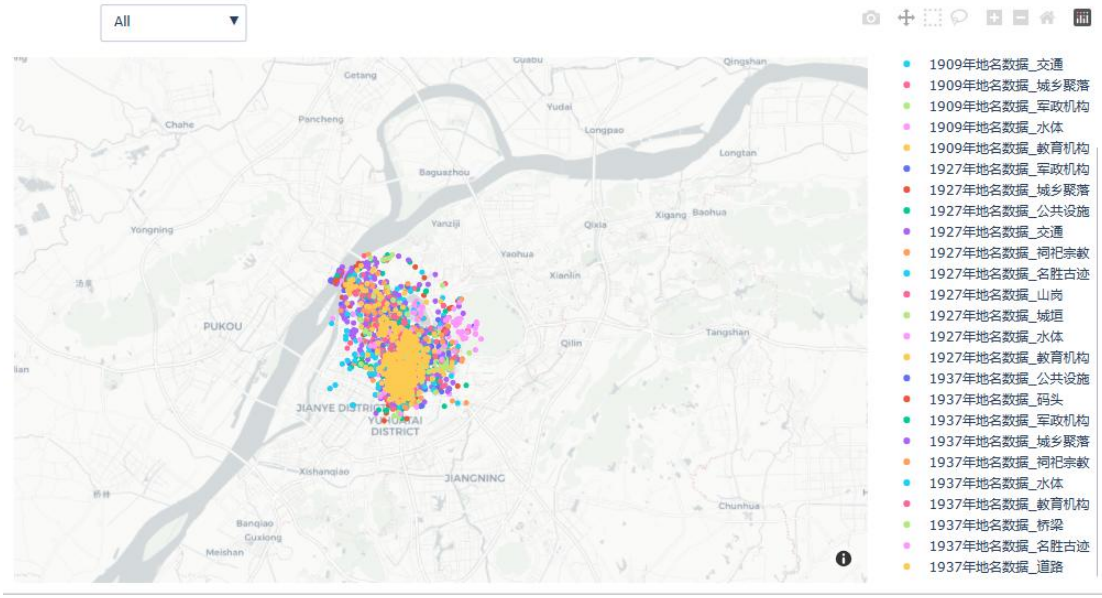
```
    # 读取南京的 Shapefile 文件
    nanjing_data = nanjing_data3.copy()
    nanjing_data['一级分类分'] = nanjing_data['时间'] + "_" +
nanjing_data['一级分类分']

    # 获取一级分类分的数量，确定颜色列表长度
    num_categories = len(nanjing_data['一级分类分'].unique())
    color_list = px.colors.qualitative.Plotly[:num_categories]
```

```
    # 创建散点图
    fig = go.Figure()

    # 添加散点图数据
    for time, color in zip(nanjing_data['时间'].unique(), color_list):
        data = nanjing_data[nanjing_data['时间'] == time]
        for category, category_color in zip(data['一级分类分'].unique(),
color_list):
            category_data = data[data['一级分类分'] == category]
            fig.add_trace(go.Scattermapbox(
                lat=category_data['纬度坐标'],
                lon=category_data['精度坐标'],
                mode='markers',
                marker=dict(color=category_color),
                text=category_data['地名'],
                hovertext= category_data['地名'],

                name=category
            ))

    # 设置地图样式和布局
    fig.update_layout(
        mapbox=dict(
            style='carto-positron',
            zoom=10,
            center=dict(lat=32.06, lon=118.80)
        ),
        margin=dict(r=0, t=30, l=0, b=0),
        showlegend=True
    )
```

```
# 添加时间下拉框按钮
buttons = []
time_list = nanjing_data['时间'].unique()

# 添加 "All" 按钮
visible_all = [True] * len(fig.data)
button_all = dict(
    label='All',
    method='update',
    args=[{'visible': visible_all}],
)
buttons.append(button_all)

for time in time_list:
    print(time)
    visible = [False] * num_categories
    for index in indexes:
        visible[index] = True

    button = dict(
        label=time,
        method='update',
        args=[{'visible': visible}],
    )
    buttons.append(button)

fig.update_layout(
    updatemenus=[
        dict(
            type='dropdown',
```

```
                direction='down',
                buttons=buttons,
                showactive=True,
                active=0,
                x=0.1,
                y=1.1,
                xanchor='left',
                yanchor='top',
            )
        ]
    )

    # 显示图表
    pyo.init_notebook_mode(connected=True)
    pyo.iplot(fig, filename='nanjing_scattermapbox.html')

    html = fig.to_html()
    with open("nanjing_map_with_points.html", "w", encoding="utf-8") as
f:
        f.write(html)
```

3.5 可视化成果



通过对民国南京城市历史地名数据集的可视化实现，最后成果如下：

- 实现了精确标识历史地名的功能，让地名在地图上清晰可见。
- 当鼠标滑动到地名上时，可以显示出地点名称、经纬坐标和地名分类等详细信息。
- 地图具备了鼠标滚轮滑动进行缩放和放大的功能，同时还可以自由移动到任意区域。
- 在左上角我们添加了一个下拉框选项，通过点选下拉框中的名称，可以跳转至 1909 年、1927 年、1937 年对应的时间轴地图图像。
- 根据地名的分类信息，我们对地图上的散点进行了不同的着色。您可以点选特定的分类，以便在地图上隐藏或显示该类别的地名。

这样的改进使得可视化成果更加直观、易于理解，并增加了交互性，让用户可以根据自己的需求浏览和探索相关信息。