问题3

思路：使用聚类（这里做不动了，直接调库（虽然前面也是调库doge））

In [1]:
```
import os
import pathlib
import numpy as np
import pandas as pd
from time import time
from sklearn.cluster import *
from numba import jit, njit, prange
import plotly.express as px
import plotly.graph_objs as go
from plotly.offline import init_notebook_mode, iplot
init_notebook_mode(connected=True)
```

In [2]:
```
ROOTDIR = pathlib.Path(os.path.abspath('.'))
IMG_HTML = ROOTDIR / 'img-html'
IMG_PNG = ROOTDIR / 'img-png'
IMG_SVG = ROOTDIR / 'img-svg'
DATA_RAW = ROOTDIR / 'data-raw'
DATA_COOKED = ROOTDIR / 'data-processed'
```

In [3]:
```
weak_grid_data = pd.read_csv(DATA_RAW / "附件1 弱覆盖栅格数据(筛选).csv")
data = weak_grid_data.sort_values(by=["x", "y"], ascending=[True, True])
Len = len(data)
data
```

Out[3]:

	x	y	traffic
139901	0	762	0.048741
140032	0	763	0.007663
140883	0	764	1.211044
141082	0	765	5.218416
141836	0	766	4.127705
...	...	...	...
64305	2499	2219	6.210206
65345	2499	2220	0.227273
168459	2499	2382	1.005365
169357	2499	2383	5.518662
13247	2499	2462	0.017207

182807 rows × 3 columns

In [4]:
```
dbscan = DBSCAN(eps=10)
kmeans = KMeans(n_clusters=1504)
```

聚类

Kmeans（未使用，时间较长，可不运行）

In [5]:
```
begin = time()
y_predict = kmeans.fit_predict(weak_grid_data.iloc[:, :2])
print("cost time:", time() - begin)  # 640s
```

```
cost time: 382.899968624115
```

DBSCAN（使用）

In [6]:
```
begin = time()
y_pred = dbscan.fit_predict(weak_grid_data.iloc[:, :2])
print("cost time:", time() - begin)  # 2.5s
```

```
cost time: 2.246206045150757
```

其他聚类（未使用，占用内存大，运行时间长）

In [9]:
```
# # Birch
# from sklearn.cluster import Birch
# begin = time()
# Birch().fit_predict(weak_grid_data.iloc[:, :2])
# print("cost time:", time() - begin)  # long long s
```

In [11]:
```
# # FeatureAgglomeration
# from sklearn.cluster import FeatureAgglomeration
# begin = time()
# FeatureAgglomeration(n_clusters=1504).fit_predict(weak_grid_data.iloc[:, :2])
# print("cost time:", time() - begin)  # s
```

可视化聚类结果

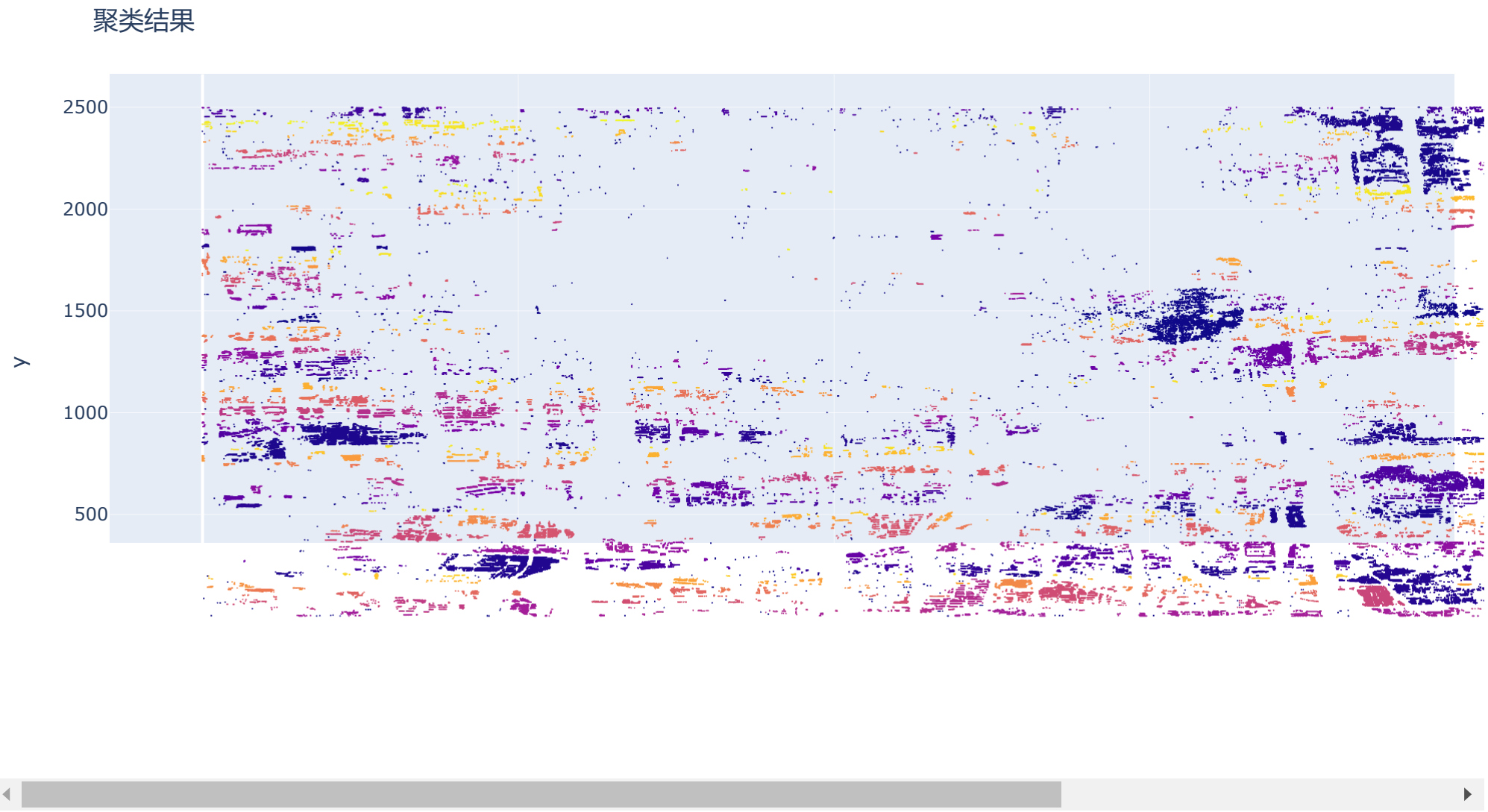
In [12]:
```
data = pd.concat([weak_grid_data, pd.DataFrame(y_pred)], axis=1)
data
```

Out[12]:

	x	y	traffic	0
0	66	1486	140.581390	-1
1	67	1486	140.518829	-1
2	177	1486	48.919178	0
3	187	1486	4.322495	0
4	284	1486	71.528404	1
...	...	...	...	...
182802	2350	2123	0.178571	36
182803	2353	2123	5.159708	36
182804	2354	2123	5.134017	36
182805	2355	2123	2.599999	36
182806	2372	2123	57.814999	1464

182807 rows × 4 columns

In [13]:
```
# todo plot scatter fig (done)
fig = px.scatter(data_frame=data, x='x', y='y', color=0)
fig.update_traces(marker={"size": 1})
fig.update_layout(title='聚类结果')
fig.write_html(IMG_HTML / "question3-DBSCAN.html")
fig.write_image(IMG_PNG / "question3-DBSCAN.png")
fig.write_image(IMG_SVG / "question3-DBSCAN.svg")
fig.show()
del fig
```



In [ ]: