Notebook 8: Correlation Graph and Parallel Coordinates Chart

Finding the relationship between migration, transit, and key metrics is the fundamental research question in our project. Therefore, we were trying to find a (statistical) way to examine the relationship. Then we decide to explore correlation analysis in Python and visualize all data via a parallel coordinates chart, which what this notebook focuses on.

In [1]:
```
from string import ascii_letters
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
```

In [2]:
```
metrics = pd.read_csv('Data/CombinedMetrics2.csv',dtype={'id':str})
```

In [3]:
```
metrics.head()
```

Out[3]:

	id	Geographic Area Name	Outflow_2018	Inflow_2018	TransitDensity	Total Population	Commuter	HomeWorker	Median Age	Bachelor Degree	...	GDP	Jobs	Income Leve
0	0500000US34003	Bergen County, New Jersey	20899	21226	0.1284	0.0104	0.3594	0.0523	0.0097	0.0612	...	0.1274	0.0427	0.366
1	0500000US36027	Dutchess County, New York	5118	9964	0.0121	-0.0117	0.1873	0.0829	0.0294	0.0689	...	0.1319	0.0402	0.127
2	0500000US34013	Essex County, New Jersey	20566	19458	0.3058	0.0050	0.3556	0.3090	0.0218	0.0977	...	0.1395	0.0571	0.452
3	0500000US09001	Fairfield County, Connecticut	15281	19508	0.0432	0.0108	0.2074	0.1525	0.0177	0.0611	...	0.0966	0.0263	0.165
4	0500000US34017	Hudson County, New Jersey	25448	20962	0.6851	0.0210	0.1679	0.1184	0.0174	0.1514	...	0.2237	0.1309	0.833

5 rows × 21 columns

Correlation Graph

In [4]:
```
metrics1 = metrics.drop(['id','Geographic Area Name'],axis=1)
```

In [5]:
```
metrics1.head()
```

Out[5]:

	Outflow_2018	Inflow_2018	TransitDensity	Total Population	Commuter	HomeWorker	Median Age	Bachelor Degree	DemographicsSum	GDP	Jobs	Income Level	GDPvsIncom
0	20899	21226	0.1284	0.0104	0.3594	0.0523	0.0097	0.0612	0.4929	0.1274	0.0427	0.3661	0.871
1	5118	9964	0.0121	-0.0117	0.1873	0.0829	0.0294	0.0689	0.3568	0.1319	0.0402	0.1270	1.024
2	20566	19458	0.3058	0.0050	0.3556	0.3090	0.0218	0.0977	0.7891	0.1395	0.0571	0.4526	1.461
3	15281	19508	0.0432	0.0108	0.2074	0.1525	0.0177	0.0611	0.4496	0.0966	0.0263	0.1655	0.690
4	25448	20962	0.6851	0.0210	0.1679	0.1184	0.0174	0.1514	0.4760	0.2237	0.1309	0.8330	1.247

correlation between outflow and others

In [6]:
```
outflow = metrics1.drop(['Inflow_2018'],axis=1)
```

In [7]:
```
outflow
```

Out[7]:

	Outflow_2018	TransitDensity	Total Population	Commuter	HomeWorker	Median Age	Bachelor Degree	DemographicsSum	GDP	Jobs	Income Level	GDPvsIncome	OwnedU
0	20899	0.1284	0.0104	0.3594	0.0523	0.0097	0.0612	0.4929	0.1274	0.0427	0.3661	0.8718	0.
1	5118	0.0121	-0.0117	0.1873	0.0829	0.0294	0.0689	0.3568	0.1319	0.0402	0.1270	1.0248	-0.
2	20566	0.3058	0.0050	0.3556	0.3090	0.0218	0.0977	0.7891	0.1395	0.0571	0.4526	1.4619	-0.
3	15281	0.0432	0.0108	0.2074	0.1525	0.0177	0.0611	0.4496	0.0966	0.0263	0.1655	0.6904	0.
4	25448	0.6851	0.0210	0.1679	0.1184	0.0174	0.1514	0.4760	0.2237	0.1309	0.8330	1.2476	0.
5	2507	0.0091	-0.0134	0.4665	0.2142	0.0382	0.0754	0.7809	0.0525	0.0167	0.0162	0.4286	-0.
6	2752	0.0000	-0.0241	-0.0727	0.0545	0.0396	0.0346	0.0321	0.0522	-0.0055	-0.0095	0.7605	-0.
7	7508	0.0349	-0.0021	0.1318	-0.0175	0.0104	0.0605	0.1832	0.1514	0.0642	0.2821	1.4234	-0.
8	24549	0.0318	0.0032	0.0933	0.2483	0.0213	0.0692	0.4353	0.1333	0.1175	1.1139	1.0249	-0.
9	12023	0.0294	-0.0100	0.0893	0.1997	0.0286	0.0720	0.3796	0.1238	0.0684	0.3806	0.5920	-0.
10	1787	0.0000	-0.0045	-1.0000	0.1263	0.0363	0.0771	-0.7648	0.1796	0.0628	0.1266	1.7621	-0.
11	11107	0.0395	-0.0055	0.2506	0.1290	0.0191	0.0621	0.4554	0.0926	0.0608	0.2787	0.6016	-0.
12	23745	0.1953	0.0044	0.1696	0.1746	0.0073	0.0849	0.4407	0.1450	0.0584	0.6739	0.7138	0.
13	8883	0.0161	-0.0044	0.0422	0.3057	0.0126	0.0564	0.4125	0.1342	0.0290	0.2862	1.7877	-0.
14	7510	0.0031	0.0181	0.0877	0.1436	0.0023	0.1239	0.3756	0.0975	0.1076	0.5825	0.6352	0.
15	6970	0.0083	0.0099	-0.0478	0.1152	0.0027	0.0600	0.1400	0.2122	0.0645	0.2652	1.8490	-0.
16	11069	0.0455	-0.0027	0.0267	0.2743	0.0165	0.0823	0.3970	0.0364	0.0306	0.1594	0.5875	-0.
17	1427	0.0000	-0.0243	-0.1026	0.1447	0.0530	0.2338	0.3046	0.1600	0.0513	0.0236	0.9603	0.
18	2649	0.0244	-0.0063	-0.0194	0.1263	0.0352	0.0557	0.1916	0.1434	0.0316	0.0224	0.9921	0.
19	6263	0.0282	0.0173	0.2091	0.1515	-0.0082	0.0136	0.3832	-0.0051	0.0757	0.2323	-0.0472	0.
20	11268	0.0361	0.0045	0.1852	0.1508	0.0221	0.0598	0.4224	0.2105	0.0717	0.2101	0.9755	-0.
21	14416	0.0439	-0.0083	0.0330	0.0602	0.0248	0.0735	0.1832	0.1475	0.0441	0.6609	0.9711	-0.
22	3014	0.0000	-0.0312	0.0476	-0.0750	0.0397	0.0449	0.0259	0.0701	0.0072	0.0084	0.6607	-0.
23	3083	0.0000	-0.0126	-0.0491	0.0507	0.0186	0.0839	0.0915	0.1504	0.0385	0.0795	1.1267	-0.
24	12883	0.1541	0.0144	0.3378	0.1854	0.0132	0.1133	0.6640	0.0835	0.0500	0.2633	0.7832	0.
25	19246	0.0954	0.0068	0.1165	0.1739	0.0124	0.0641	0.3737	0.1722	0.0460	0.3081	0.8426	0.

In [8]:
```
# Compute the correlation matrix
corr = outflow.corr()

# Generate a mask for the upper triangle
mask = np.triu(np.ones_like(corr, dtype=bool))

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(20, 20))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(150, 0, as_cmap=True)

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=1, vmin=-1, center=0,
            square=True, linewidths=1, cbar_kws={"shrink": .8}, annot = True, fmt='.2g')

ax.set_title('Correlation Between Outflow and Other Factors', size='20')
```

Out[8]: `Text(0.5, 1.0, 'Correlation Between Outflow and Other Factors')`

Calculate inflow

In [9]:
```
inflow = metrics1.drop(['Outflow_2018'],axis=1)
```

In [10]:
```
# Compute the correlation matrix
corr = inflow.corr()

# Generate a mask for the upper triangle
mask = np.triu(np.ones_like(corr, dtype=bool))

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(20, 20))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(150, 0, as_cmap=True)

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=1, vmin=-1, center=0,
            square=True, linewidths=1, cbar_kws={"shrink": .8}, annot = True, fmt='.2g')

ax.set_title('Correlation Between Inflow and Other Factors', size='20')
```

Out[10]: `Text(0.5, 1.0, 'Correlation Between Inflow and Other Factors')`

Parallel Coordinates Graphs

In [11]:
```
metrics.head()
```

Out[11]:

	id	Geographic Area Name	Outflow_2018	Inflow_2018	TransitDensity	Total Population	Commuter	HomeWorker	Median Age	Bachelor Degree	...	GDP	Jobs	Income Leve
0	0500000US34003	Bergen County, New Jersey	20899	21226	0.1284	0.0104	0.3594	0.0523	0.0097	0.0612	...	0.1274	0.0427	0.366
1	0500000US36027	Dutchess County, New York	5118	9964	0.0121	-0.0117	0.1873	0.0829	0.0294	0.0689	...	0.1319	0.0402	0.127
2	0500000US34013	Essex County, New Jersey	20566	19458	0.3058	0.0050	0.3556	0.3090	0.0218	0.0977	...	0.1395	0.0571	0.452
3	0500000US09001	Fairfield County, Connecticut	15281	19508	0.0432	0.0108	0.2074	0.1525	0.0177	0.0611	...	0.0966	0.0263	0.165
4	0500000US34017	Hudson County, New Jersey	25448	20962	0.6851	0.0210	0.1679	0.1184	0.0174	0.1514	...	0.2237	0.1309	0.833

5 rows × 21 columns

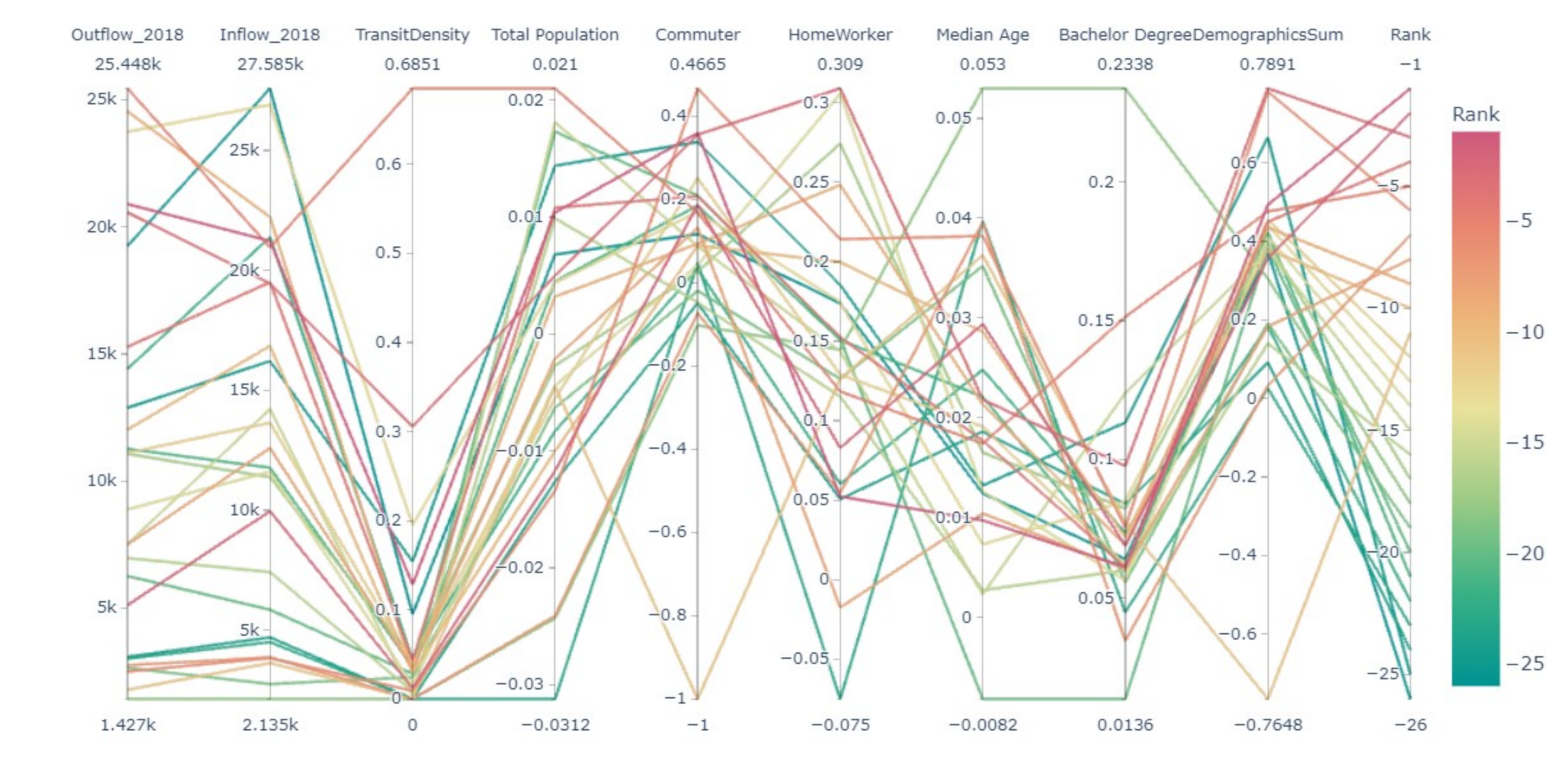
In [12]:
```
list(metrics)
```

Out[12]:
```
['id',
 'Geographic Area Name',
 'Outflow_2018',
 'Inflow_2018',
 'TransitDensity',
 'Total Population',
 'Commuter',
 'HomeWorker',
 'Median Age',
 'Bachelor Degree',
 'DemographicsSum',
 'GDP',
 'Jobs',
 'Income Level',
 'GDPvsIncome',
 'OwnedUnits',
 'RentalUnits',
 'Values',
 'Rents',
 'OwnedAffordability',
 'RentalAffordability']
```

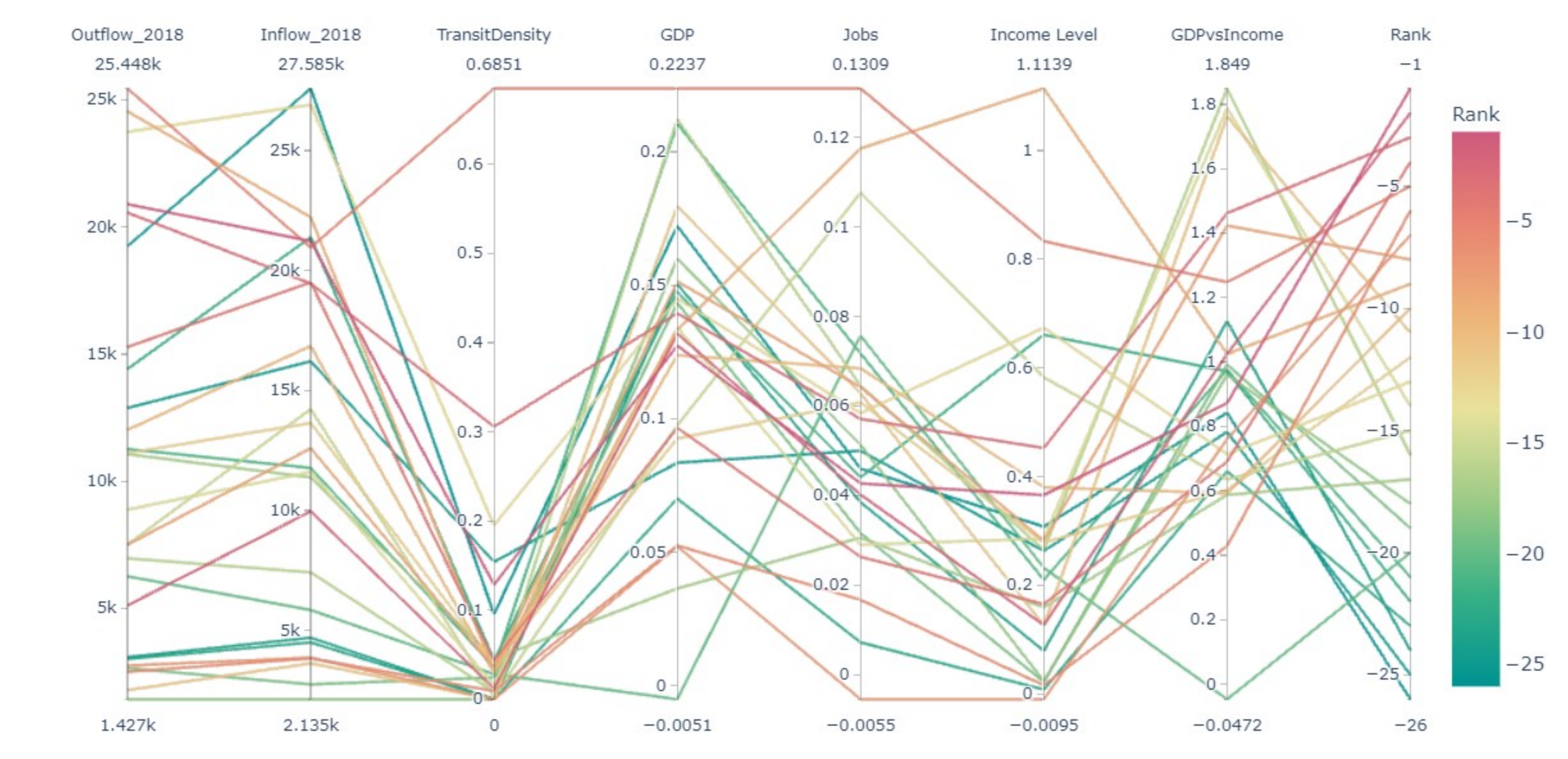
In [13]:
```
demographics = [
 'Geographic Area Name',
 'Outflow_2018',
 'Inflow_2018',
 'TransitDensity',
 'Total Population',
 'Commuter',
 'HomeWorker',
 'Median Age',
 'Bachelor Degree',
 'DemographicsSum',
]

economics = [
 'Geographic Area Name',
 'Outflow_2018',
 'Inflow_2018',
 'TransitDensity',
 'GDP',
 'Jobs',
 'Income Level',
 'GDPvsIncome',
]

housing = [
 'Geographic Area Name',
 'Outflow_2018',
 'Inflow_2018',
 'TransitDensity',
 'OwnedUnits',
 'RentalUnits',
 'Values',
 'Rents',
 'OwnedAffordability',
 'RentalAffordability',
]
```

In [14]:
```
demographics = metrics[demographics]
economics = metrics[economics]
housing = metrics[housing]
```

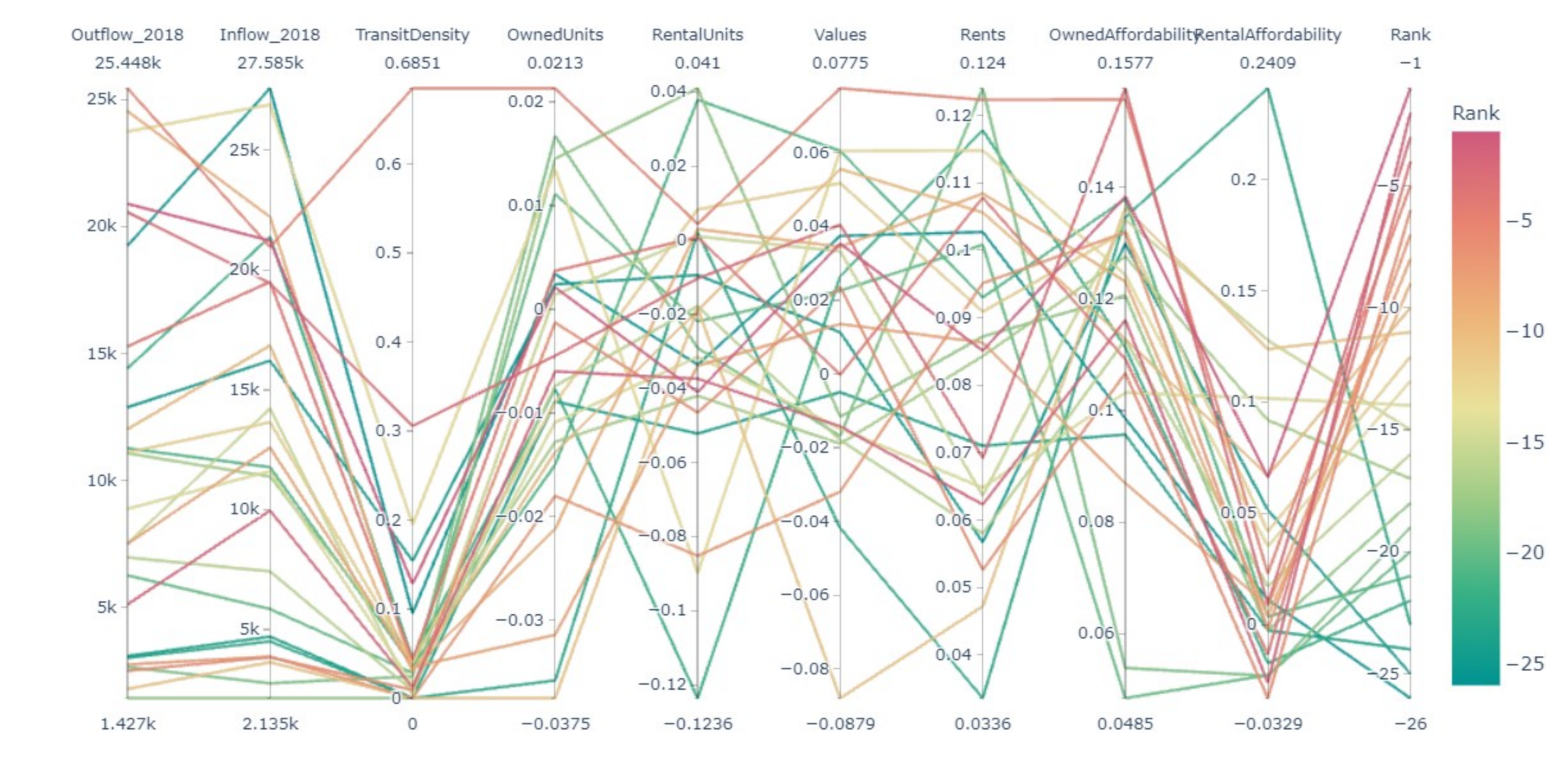
In [15]:
```
demographics = demographics.sort_values(by='Outflow_2018')
demographics['Rank'] = -np.arange(len(demographics))-1

fig = px.parallel_coordinates(demographics,
                              color="Rank",
                              color_continuous_scale = px.colors.diverging.Temps,
                              )
fig.show()
```



In [16]:
```
economics = economics.sort_values(by='Outflow_2018')
economics['Rank'] = -np.arange(len(economics)) - 1

fig2 = px.parallel_coordinates(economics,
                               color="Rank",
                               color_continuous_scale = px.colors.diverging.Temps,
                               )
fig2.show()
```



In [17]:
```
housing = housing.sort_values(by='Outflow_2018')
housing['Rank'] = -np.arange(len(housing)) - 1

fig3 = px.parallel_coordinates(housing,
                               color="Rank",
                               color_continuous_scale = px.colors.diverging.Temps,
                               )


fig3.show()
```



In [18]:
```
#fig.write_html("ExportData/vs_demographics_Interactive.html")
#fig2.write_html("ExportData/vs_economics_Interactive.html")
#fig3.write_html("ExportData/vs_housing_Interactive.html")
```

Concluding Note:

Those are our concluding graphs – we really like the way how they look. The correlation coefficient numbers and the associated heatmap for those numbers provide a straightforward way to present the correlations. As for the parallel coordinates, they present the dynamics of each county in terms of their metrics. The graph-making process itself is not hard. Thank to our previous steps, we have trimmed and ready-to-use data frame for those graphs.

In [ ]: