Creating a feature matrix from a networkx graph

In this notebook we will look at a few ways to quickly create a feature matrix from a networkx graph.

In [1]:
```
import networkx as nx
import pandas as pd

G = nx.read_gpickle('major_us_cities')
```

Node based features

In [2]:
```
G.nodes(data=True)
```

Out[2]:
```
[('El Paso, TX', {'location': (-106, 31), 'population': 674433}),
 ('Long Beach, CA', {'location': (-118, 33), 'population': 469428}),
 ('Dallas, TX', {'location': (-96, 32), 'population': 1257676}),
 ('Oakland, CA', {'location': (-122, 37), 'population': 406253}),
 ('Albuquerque, NM', {'location': (-106, 35), 'population': 556495}),
 ('Baltimore, MD', {'location': (-76, 39), 'population': 622104}),
 ('Raleigh, NC', {'location': (-78, 35), 'population': 431746}),
 ('Mesa, AZ', {'location': (-111, 33), 'population': 457587}),
 ('Arlington, TX', {'location': (-97, 32), 'population': 379577}),
 ('Sacramento, CA', {'location': (-121, 38), 'population': 479686}),
 ('Wichita, KS', {'location': (-97, 37), 'population': 386552}),
 ('Tucson, AZ', {'location': (-110, 32), 'population': 526116}),
 ('Cleveland, OH', {'location': (-81, 41), 'population': 390113}),
 ('Louisville/Jefferson County, KY',
  {'location': (-85, 38), 'population': 609893}),
 ('San Jose, CA', {'location': (-121, 37), 'population': 998537}),
 ('Oklahoma City, OK', {'location': (-97, 35), 'population': 610613}),
 ('Atlanta, GA', {'location': (-84, 33), 'population': 447841}),
 ('New Orleans, LA', {'location': (-90, 29), 'population': 378715}),
 ('Miami, FL', {'location': (-80, 25), 'population': 417650}),
```

In [3]:
```
# Initialize the dataframe, using the nodes as the index
df = pd.DataFrame(index=G.nodes())
```

Extracting attributes

Using `nx.get_node_attributes` it's easy to extract the node attributes in the graph into DataFrame columns.

In [4]:
```
df['location'] = pd.Series(nx.get_node_attributes(G, 'location'))
df['population'] = pd.Series(nx.get_node_attributes(G, 'population'))

df.head()
```

Out[4]:

	location	population
El Paso, TX	(-106, 31)	674433
Long Beach, CA	(-118, 33)	469428
Dallas, TX	(-96, 32)	1257676
Oakland, CA	(-122, 37)	406253
Albuquerque, NM	(-106, 35)	556495

Creating node based features

Most of the networkx functions related to nodes return a dictionary, which can also easily be added to our dataframe.

In [5]:
```
df['clustering'] = pd.Series(nx.clustering(G))
df['degree'] = pd.Series(G.degree())

df
```

Out[5]:

	location	population	clustering	degree
El Paso, TX	(-106, 31)	674433	0.700000	5
Long Beach, CA	(-118, 33)	469428	0.745455	11
Dallas, TX	(-96, 32)	1257676	0.763636	11
Oakland, CA	(-122, 37)	406253	1.000000	8
Albuquerque, NM	(-106, 35)	556495	0.523810	7
Baltimore, MD	(-76, 39)	622104	0.800000	10
Raleigh, NC	(-78, 35)	431746	0.615385	13
Mesa, AZ	(-111, 33)	457587	0.750000	8
Arlington, TX	(-97, 32)	379577	0.763636	11
Sacramento, CA	(-121, 38)	479686	0.777778	9
Wichita, KS	(-97, 37)	386552	0.622222	10

Edge based features

In [6]:
```
G.edges(data=True)
```

Out[6]:
```
[('El Paso, TX', 'Albuquerque, NM', {'weight': 367.88584356108345}),
 ('El Paso, TX', 'Mesa, AZ', {'weight': 536.256659972679}),
 ('El Paso, TX', 'Tucson, AZ', {'weight': 425.41386739988224}),
 ('El Paso, TX', 'Phoenix, AZ', {'weight': 558.7835703774161}),
 ('El Paso, TX', 'Colorado Springs, CO', {'weight': 797.7517116740046}),
 ('Long Beach, CA', 'Oakland, CA', {'weight': 579.5829987228403}),
 ('Long Beach, CA', 'Mesa, AZ', {'weight': 590.156204210031}),
 ('Long Beach, CA', 'Sacramento, CA', {'weight': 611.0649790490104}),
 ('Long Beach, CA', 'Tucson, AZ', {'weight': 698.6566667728368}),
 ('Long Beach, CA', 'San Jose, CA', {'weight': 518.2330606219175}),
```

```
('Long Beach, CA', 'Fresno, CA', {'weight': 360.4704577972272}),
('Long Beach, CA', 'San Diego, CA', {'weight': 151.45008247402757}),
('Long Beach, CA', 'Phoenix, AZ', {'weight': 567.4125390872786}),
('Long Beach, CA', 'San Francisco, CA', {'weight': 585.6985397766858}),
('Long Beach, CA', 'Los Angeles, CA', {'weight': 31.69419563651866}),
('Long Beach, CA', 'Las Vegas, NV', {'weight': 385.2597725411484}),
('Dallas, TX', 'Arlington, TX', {'weight': 29.425931317908415}),
('Dallas, TX', 'Wichita, KS', {'weight': 548.0572491959326}),
('Dallas, TX', 'Oklahoma City, OK', {'weight': 306.2597807397289}),
('Dallas, TX', 'New Orleans, LA', {'weight': 711.0141469371868}),
```

In [7]:
```
# Initialize the dataframe, using the edges as the index
df = pd.DataFrame(index=G.edges())
```

Extracting attributes

Using `nx.get_edge_attributes`, it's easy to extract the edge attributes in the graph into DataFrame columns.

In [8]:
```
df['weight'] = pd.Series(nx.get_edge_attributes(G, 'weight'))

df
```

Out[8]:

	weight
(El Paso, TX, Albuquerque, NM)	367.885844
(El Paso, TX, Mesa, AZ)	536.256660
(El Paso, TX, Tucson, AZ)	425.413867
(El Paso, TX, Phoenix, AZ)	558.783570
(El Paso, TX, Colorado Springs, CO)	797.751712
(Long Beach, CA, Oakland, CA)	579.582999
(Long Beach, CA, Mesa, AZ)	590.156204
(Long Beach, CA, Sacramento, CA)	611.064979
(Long Beach, CA, Tucson, AZ)	698.656667
(Long Beach, CA, San Jose, CA)	518.233061
(Long Beach, CA, Fresno, CA)	360.470458

Creating edge based features

Many of the networkx functions related to edges return a nested data structures. We can extract the relevant data using list comprehension.

In [9]:
```
df['preferential attachment'] = [i[2] for i in nx.preferential_attachment(G, df.index)]

df
```

Out[9]:

	weight	preferential attachment
(El Paso, TX, Albuquerque, NM)	367.885844	35
(El Paso, TX, Mesa, AZ)	536.256660	40
(El Paso, TX, Tucson, AZ)	425.413867	40
(El Paso, TX, Phoenix, AZ)	558.783570	45
(El Paso, TX, Colorado Springs, CO)	797.751712	30
(Long Beach, CA, Oakland, CA)	579.582999	88
(Long Beach, CA, Mesa, AZ)	590.156204	88
(Long Beach, CA, Sacramento, CA)	611.064979	99
(Long Beach, CA, Tucson, AZ)	698.656667	88
(Long Beach, CA, San Jose, CA)	518.233061	88
(Long Beach, CA, Fresno, CA)	360.470458	99
(Long Beach, CA, San Diego, CA)	151.450082	121
(Long Beach, CA, Phoenix, AZ)	567.412539	99
(Long Beach, CA, San Francisco, CA)	585.698540	88
(Long Beach, CA, Los Angeles, CA)	31.694196	121
(Long Beach, CA, Las Vegas, NV)	385.259773	132
(Dallas, TX, Arlington, TX)	29.425931	121
(Dallas, TX, Wichita, KS)	548.057249	110
(Dallas, TX, Oklahoma City, OK)	306.259781	132
(Dallas, TX, New Orleans, LA)	711.014147	88
(Dallas, TX, Houston, TX)	361.541859	99
(Dallas, TX, Kansas City, MO)	730.377588	154
(Dallas, TX, San Antonio, TX)	406.006566	77
(Dallas, TX, Memphis, TN)	675.331624	154
(Dallas, TX, Tulsa, OK)	382.464108	121
(Dallas, TX, Austin, TX)	292.911466	88
(Dallas, TX, Fort Worth, TX)	49.933599	121
(Oakland, CA, Sacramento, CA)	109.821598	72
(Oakland, CA, San Jose, CA)	61.902923	64
(Oakland, CA, Fresno, CA)	250.230973	72
...	...	...
(Memphis, TN, Tulsa, OK)	548.096113	154
(Memphis, TN, Fort Worth, TX)	721.784550	154
(Memphis, TN, Indianapolis, IN)	617.295220	182
(Memphis, TN, Nashville-Davidson, TN)	315.750110	182
(Los Angeles, CA, Las Vegas, NV)	367.372801	132
(New York, NY, Detroit, MI)	772.901141	99
(New York, NY, Columbus, OH)	765.963451	135
(New York, NY, Virginia Beach, VA)	461.656907	81
(Denver, CO, Omaha, NE)	777.886325	36
(Denver, CO, Colorado Springs, CO)	101.650759	24
(Omaha, NE, Tulsa, OK)	566.558436	99
(Omaha, NE, Minneapolis, MN)	469.537646	36

(Omaha, NE, Colorado Springs, CO)	796.839490	54
(Omaha, NE, Milwaukee, WI)	695.285566	90
(Seattle, WA, Portland, OR)	232.980035	2
(Tulsa, OK, Austin, TX)	674.071177	88
(Tulsa, OK, Fort Worth, TX)	397.046256	121
(Austin, TX, Fort Worth, TX)	279.260772	88
(Minneapolis, MN, Milwaukee, WI)	479.287760	40
(Indianapolis, IN, Detroit, MI)	386.140841	143
(Indianapolis, IN, Nashville-Davidson, TN)	404.391322	169
(Indianapolis, IN, Milwaukee, WI)	391.538845	130
(Indianapolis, IN, Columbus, OH)	270.306348	195
(Detroit, MI, Milwaukee, WI)	404.703490	110
(Detroit, MI, Nashville-Davidson, TN)	756.943865	143
(Detroit, MI, Columbus, OH)	263.423765	165
(Nashville-Davidson, TN, Milwaukee, WI)	770.146706	130
(Nashville-Davidson, TN, Columbus, OH)	536.274548	195
(Milwaukee, WI, Columbus, OH)	532.568423	150
(Columbus, OH, Virginia Beach, VA)	701.876666	135

235 rows × 2 columns

In the case where the function expects two nodes to be passed in, we can map the index to a lamda function.

In [10]:
```
df['Common Neighbors'] = df.index.map(lambda city: len(list(nx.common_neighbors(G, city[0], city[1]))))

df
```

Out[10]:

	weight	preferential attachment	Common Neighbors
(El Paso, TX, Albuquerque, NM)	367.885844	35	4
(El Paso, TX, Mesa, AZ)	536.256660	40	3
(El Paso, TX, Tucson, AZ)	425.413867	40	3
(El Paso, TX, Phoenix, AZ)	558.783570	45	3
(El Paso, TX, Colorado Springs, CO)	797.751712	30	1
(Long Beach, CA, Oakland, CA)	579.582999	88	7
(Long Beach, CA, Mesa, AZ)	590.156204	88	5
(Long Beach, CA, Sacramento, CA)	611.064979	99	7
(Long Beach, CA, Tucson, AZ)	698.656667	88	5
(Long Beach, CA, San Jose, CA)	518.233061	88	7
(Long Beach, CA, Fresno, CA)	360.470458	99	8
(Long Beach, CA, San Diego, CA)	151.450082	121	10
(Long Beach, CA, Phoenix, AZ)	567.412539	99	6
(Long Beach, CA, San Francisco, CA)	585.698540	88	7
(Long Beach, CA, Los Angeles, CA)	31.694196	121	10
(Long Beach, CA, Las Vegas, NV)	385.259773	132	10
(Dallas, TX, Arlington, TX)	29.425931	121	10
(Dallas, TX, Wichita, KS)	548.057249	110	6
(Dallas, TX, Oklahoma City, OK)	306.259781	132	9
(Dallas, TX, New Orleans, LA)	711.014147	88	5
(Dallas, TX, Houston, TX)	361.541859	99	8
(Dallas, TX, Kansas City, MO)	730.377588	154	6
(Dallas, TX, San Antonio, TX)	406.006566	77	6
(Dallas, TX, Memphis, TN)	675.331624	154	8
(Dallas, TX, Tulsa, OK)	382.464108	121	9
(Dallas, TX, Austin, TX)	292.911466	88	7
(Dallas, TX, Fort Worth, TX)	49.933599	121	10
(Oakland, CA, Sacramento, CA)	109.821598	72	7
(Oakland, CA, San Jose, CA)	61.902923	64	7
(Oakland, CA, Fresno, CA)	250.230973	72	7
...	...	...	...
(Memphis, TN, Tulsa, OK)	548.096113	154	7
(Memphis, TN, Fort Worth, TX)	721.784550	154	8
(Memphis, TN, Indianapolis, IN)	617.295220	182	5
(Memphis, TN, Nashville-Davidson, TN)	315.750110	182	6
(Los Angeles, CA, Las Vegas, NV)	367.372801	132	10
(New York, NY, Detroit, MI)	772.901141	99	5
(New York, NY, Columbus, OH)	765.963451	135	7
(New York, NY, Virginia Beach, VA)	461.656907	81	7
(Denver, CO, Omaha, NE)	777.886325	36	2
(Denver, CO, Colorado Springs, CO)	101.650759	24	3
(Omaha, NE, Tulsa, OK)	566.558436	99	3
(Omaha, NE, Minneapolis, MN)	469.537646	36	3
(Omaha, NE, Colorado Springs, CO)	796.839490	54	3
(Omaha, NE, Milwaukee, WI)	695.285566	90	3
(Seattle, WA, Portland, OR)	232.980035	2	0
(Tulsa, OK, Austin, TX)	674.071177	88	6
(Tulsa, OK, Fort Worth, TX)	397.046256	121	9
(Austin, TX, Fort Worth, TX)	279.260772	88	7
(Minneapolis, MN, Milwaukee, WI)	479.287760	40	3
(Indianapolis, IN, Detroit, MI)	386.140841	143	7
(Indianapolis, IN, Nashville-Davidson, TN)	404.391322	169	11
(Indianapolis, IN, Milwaukee, WI)	391.538845	130	7
(Indianapolis, IN, Columbus, OH)	270.306348	195	10

(Detroit, MI, Milwaukee, WI)	404.703490	110	6
(Detroit, MI, Nashville-Davidson, TN)	756.943865	143	6
(Detroit, MI, Columbus, OH)	263.423765	165	10
(Nashville-Davidson, TN, Milwaukee, WI)	770.146706	130	7
(Nashville-Davidson, TN, Columbus, OH)	536.274548	195	9
(Milwaukee, WI, Columbus, OH)	532.568423	150	6
(Columbus, OH, Virginia Beach, VA)	701.876666	135	7

235 rows × 3 columns

In [ ]: