Homework 1

Deepak Akhare

September 13, 2022

Problem 1: Katz centrality

$$c_{Katz} = \beta(\mathbf{I} - \alpha\mathbf{A})^{-1}\vec{1} \tag{1}$$

By definition: $\alpha > 0$

For the convergence of Katz centrality, we need to seek for α such that $(I - \alpha\mathbf{A})^{-1}$ does not diverge, i.e., the inverse of $(I - \alpha\mathbf{A})$ should exist, which requires

$$det(\mathbf{A} - \alpha^{-1}\mathbf{I}) \neq 0$$

However for Eigen-value matrix Λ

$$det(\mathbf{A} - \Lambda\mathbf{I}) = 0$$

The first value of α that makes this determinant 0 is

$$\alpha = \frac{1}{\lambda_{max}}$$

Therefore,

$$0 < \alpha < \frac{1}{\lambda_{max}} \tag{2}$$

will be a sufficient condition for the convergence of Katz centrality

Problem 2:

Number of walk of size 2 from v_i to v_j that go through $v_k \in V$ is

$$N_{ij}^{(2)} = \sum_{k=1}^{n} A_{ik} A_{kj} = [A^2]_{ij} \tag{3}$$

A walk of size 2 from v_i to v_j that go through $v_k \in V$ clearly means that v_k is the neighbor of both v_i and v_j. Therefore the total number of common neighbors $|N(v_i) \cap N(v_j)|$ between nodes v_i and v_j is

$$|N(v_i) \cap N(v_j)| = N_{ij}^{(2)} = \sum_{k=1}^{n} A_{ik} A_{kj} = [A^2]_{ij} \tag{4}$$

Problem 3A:

program to compute Jaccard's similarity matrix S

```
A = nx.adjacency_matrix(G)
S = np.zeros_like(A.todense(), dtype=float)
for i in range(A.shape[0]):
    for j in range(A.shape[0]):
        S[i,j] = np.array((sum(A[:,i].multiply(A[:,j])) /
        len((A[:,i]+A[:,j]).nonzero()[0])).todense())[0][0]
```

program to calculate edge between "Ginori" family and other families in the Florentine Families graph

```

new_edges, metric = [], []
for v in Ginori_dict:
    u = 'Ginori'
    p = Ginori_dict[v]
    G.add_edge(u, v)
    print(f"({u}, {v}) -> {p:.8f}")
    new_edges.append((u, v))
    metric.append(p)
```

Updated program for plotting

```

# -- plot Florentine Families graph
nx.draw_networkx_nodes(G, nodelist=nodes, label=nodes, pos=layout, node_size=600)
nx.draw_networkx_edges(G, edgelist=new_edges, pos=layout, edge_color='gray', width=4)

ne = nx.draw_networkx_edges(G, edgelist=new_edges, pos=layout, edge_color=np.asarray(
                                          metric), width=4, alpha=0.7)
plt.colorbar(ne)
plt.axis('off')
plt.savefig('similarity.png')
```

Problem 3B:

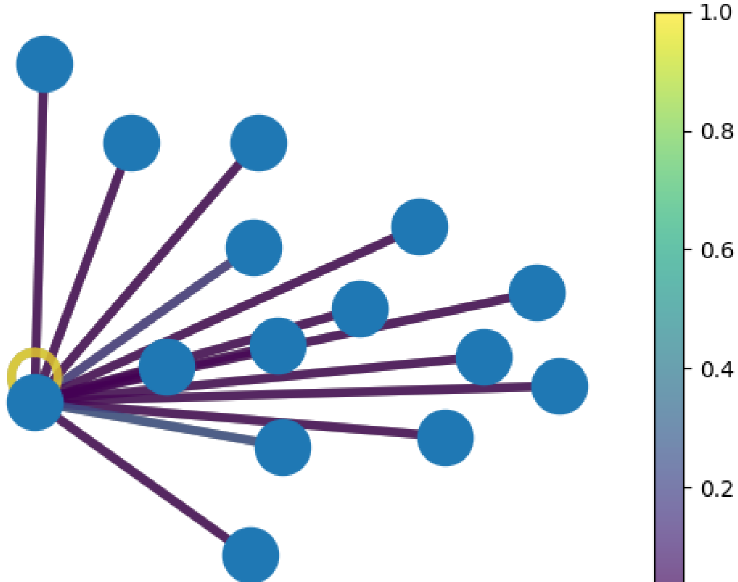

Figure 1: Similarity between "Ginori" family and other families in the Florentine Families graph