Graph Based Model: Community Detection & Link Prediction

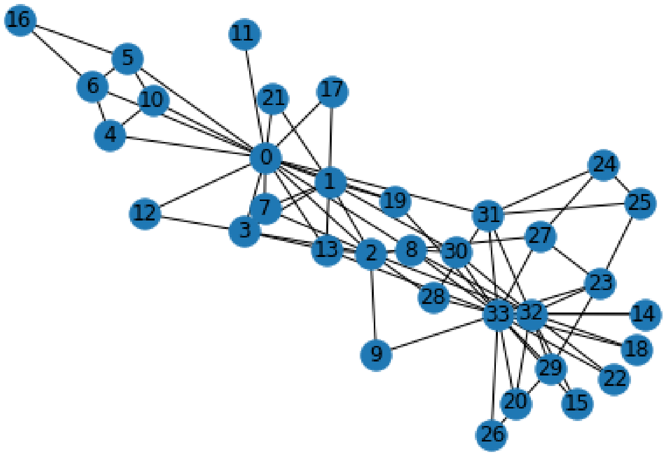
1. **Graph Used:** Karate Club Graph

The dataset contains social ties among the members of a university karate club collected by Wayne Zachary.

Nodes	Edges	Maximum Degree	Minimum degree	Average Degree
34	78	17	1	4

Google Colab Code Notebook Link:
https://colab.research.google.com/drive/1Mxr5ol5YUkIxZ_CuUOpva3n7E500gmeR?usp=sharing

Visualization of graph:



2. Community Detection using different Algorithms

2.1. Girvan Newman Method

The Girvan Newman Algorithm is the classic division clustering method which works on the centrality measure of **Edge betweenness** where betweenness is the number of the shortest paths between pairs of nodes.

Girvan-Newman will remove the edges with the largest **edge betweenness** in every iteration/level.

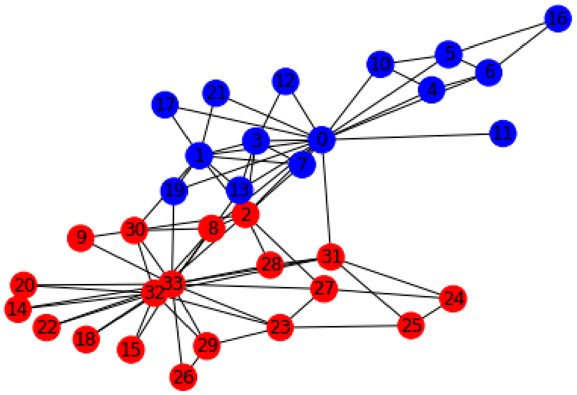
Algorithm:

1. For a given graph calculate the betweenness of all existing edges present in the graph.
2. Now remove all the edge(s) with the highest betweenness.
3. Now recalculate the betweenness of all the edges that got affected by the removal of edges.
4. Now repeat steps 2 and 3 until no edges remain.

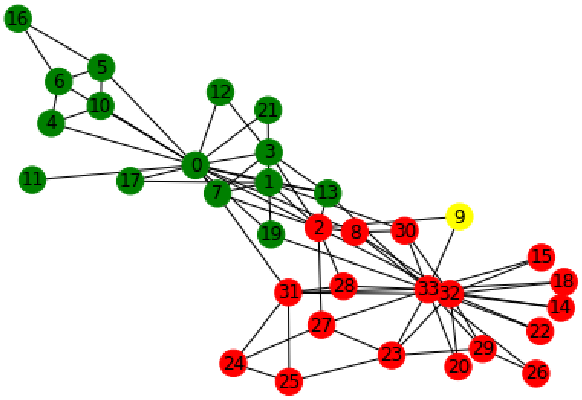
For example if level =1, there will be two communities detected.
For level =2, there will be three communities detected.
For level =3, there will be four communities detected and so on.

Observations:

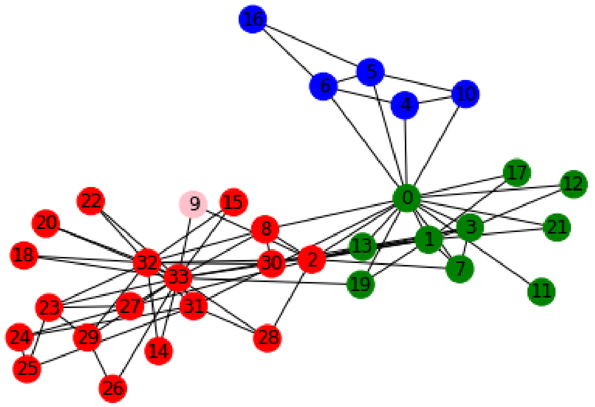
Level: 1

No. of Communities Detected	2
Nodes present in different communities	**Community 1:** [2, 8, 9, 14, 15, 18, 20, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33] **Community 2:** [0, 1, 3, 4, 5, 6, 7, 10, 11, 12, 13, 16, 17, 19, 21]
Visualization	 16 10 5 6 12 4 21 17 3 0 11 1 7 19 13 2 9 30 8 31 28 20 33 32 27 24 14 22 18 23 25 15 29 26

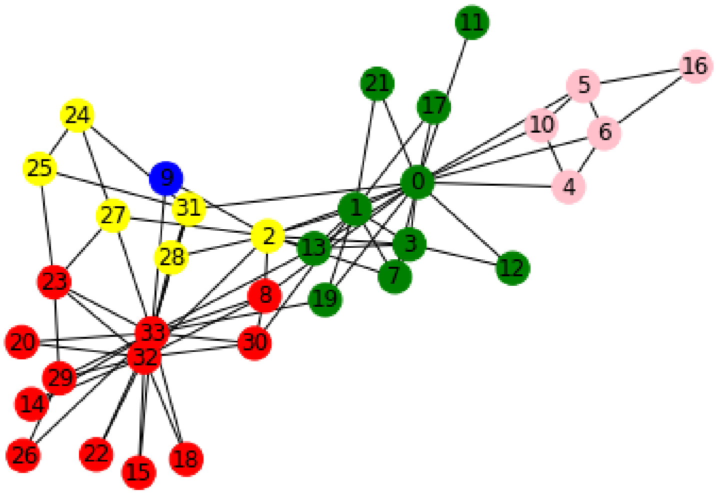
Level: 2

No. of Communities Detected	3
Nodes present in different communities	**Community 1:** [32, 33, 2, 8, 14, 15, 18, 20, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31] **Community 2:** [0, 1, 3, 4, 5, 6, 7, 10, 11, 12, 13, 16, 17, 19, 21] **Community 3:** [9]
Visualization	 16 5 6 12 10 21 4 3 0 11 17 1 7 13 9 2 8 30 19 15 33 32 18 31 28 14 27 22 29 24 23 20 26 25

Level: 3

No. of Communities Detected	4
Nodes present in different communities	**Community 1:** [32, 33, 2, 8, 14, 15, 18, 20, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31] **Community 2:** [0, 1, 3, 7, 11, 12, 13, 17, 19, 21] **Community 3:** [4, 5, 6, 10, 16] **Community 4:** [9]
Visualization	 16 5 6 4 10 17 12 0 22 9 15 20 8 3 21 18 13 1 32 30 2 33 7 19 23 27 31 11 24 29 28 14 25 26

Level: 4

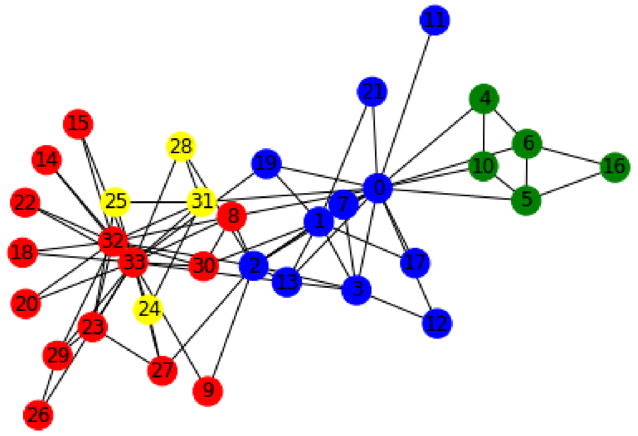
No. of Communities Detected	5
Nodes present in different communities	**Community 1:** [32, 33, 8, 14, 15, 18, 20, 22, 23, 26, 29, 30] **Community 2:** [0, 1, 3, 7, 11, 12, 13, 17, 19, 21] **Community 3:** [2, 24, 25, 27, 28, 31] **Community 4:** [4, 5, 6, 10, 16] **Community 5:** [9]
Visualization	 11 16 21 5 17 24 10 6 25 9 0 4 31 1 27 2 28 13 3 23 7 12 8 19 33 20 30 32 29 14 26 22 18 15

2.2. Louvain Method

This method is divided in 2 phases: **Modularity Optimization and Community Aggregation.**

This approach is based on modularity, which tries to maximize the difference between the actual number of edges in a community and the expected number of edges in the community.

Observation:

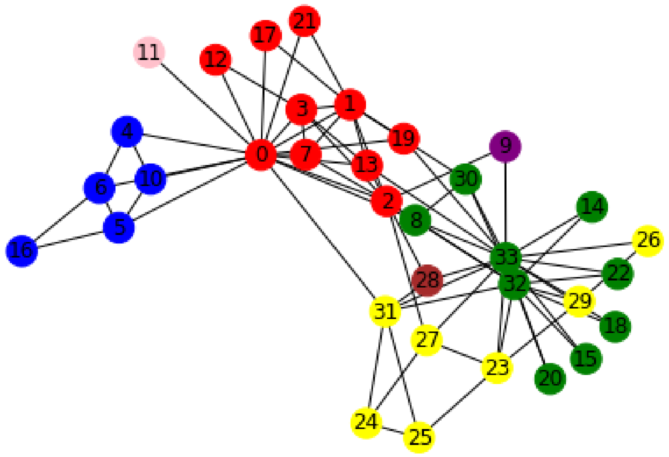
No. of Communities Detected	4
Nodes present in different communities	**Community 1:** [0, 1, 2, 3, 7, 11, 12, 13, 17, 19,21] **Community 2:** [4, 5, 6, 10, 16] **Community 3:** [24, 25, 28, 31] **Community 4:** [8, 9, 14, 15, 18, 20, 22, 23, 26, 27, 29, 30, 32, 33]
Visualization	

2.3. Surprise Communities

The algorithm is almost similar to the Louvain community detection algorithm except that it uses surprises, i.e. probability instead of modularity. Nodes are moved from one community to another such that surprises are greedily improved.

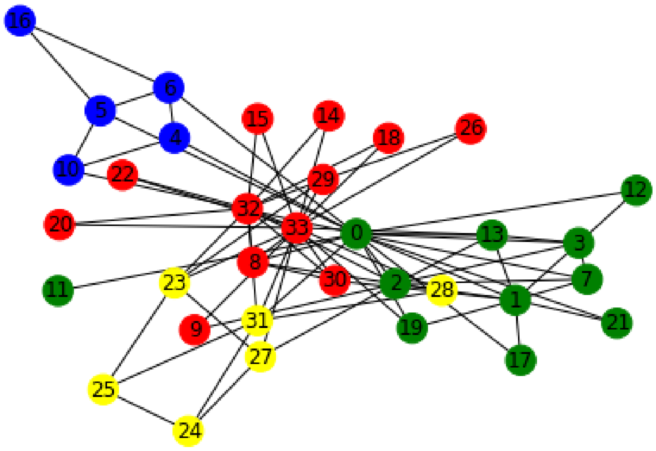
The use of surprises works well in the limit of many small communities and the use of modularity works well in the limit of a few large communities.

No. of Communities Detected	8
Nodes present in different communities	**Community 1:** [0, 1, 2, 3, 7, 12, 13, 17, 19, 21] **Community 2:** [8, 14, 15, 18, 20, 22, 30, 32, 33] **Community 3:** [23, 24, 25, 27, 31] **Community 4:** [4, 5, 6, 10, 16] **Community 5:** [26, 29] **Community 6:** [9] **Community 7:** [11] **Community 8:** [28]

No. of Communities Detected	8
Visualization	 11 12 17 21 3 1 19 9 4 0 7 13 30 6 10 2 14 8 5 16 26 33 28 22 32 29 31 18 27 15 23 20 24 25

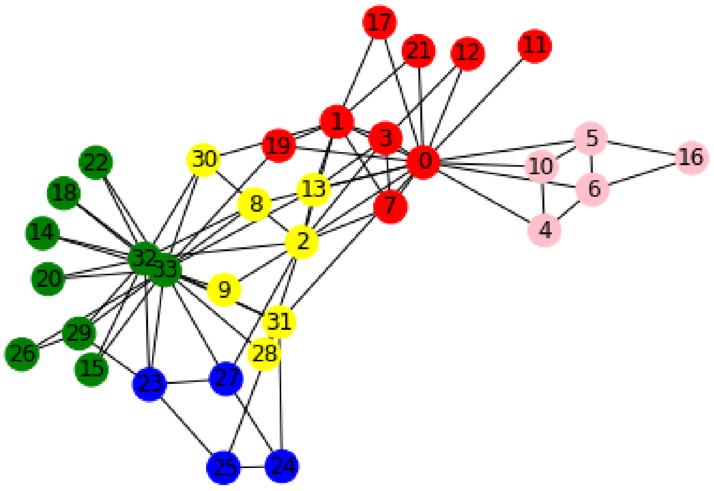
2.4. Leiden Algorithm:

Louvain has a tendency to discover very weakly connected communities. Therefore, they have proposed the much faster Leiden algorithm which guarantees that communities are well connected.

No. of Communities Detected	4
Nodes present in different communities	**Community 1:** [8, 9, 14, 15, 18, 20, 22, 26, 29, 30, 32, 33] **Community 2:** [0, 1, 2, 3, 7, 11, 12, 13, 17, 19, 21] **Community 3:** [23, 24, 25, 27, 28, 31] **Community 4:** [4, 5, 6, 10, 16]
Visualization	 16 6 5 15 14 4 18 26 10 22 29 12 32 20 33 0 13 3 8 7 11 23 30 2 28 1 21 9 31 19 27 17 25 24

2.5. Walktrap Algorithm:

In this algorithm communities in *large networks are identified via random walks* which are used to compute distances between nodes. Nodes are then assigned into groups with small intra- and larger inter-community distances via bottom-up hierarchical clustering.

No. of Communities Detected	5
Nodes present in different communities	**Community 1:** [0, 1, 3, 7, 11, 12, 17, 19, 21] **Community 2:** [14, 15, 18, 20, 22, 26, 29, 32, 33] **Community 3:** [2, 8, 9, 13, 28, 30, 31] **Community 4:** [4, 5, 6, 10, 16] **Community 5:** [23, 24, 25, 27]
Visualization	 17 21 12 11 1 3 0 5 16 10 6 4 7 19 30 22 18 13 8 14 2 32 33 20 9 31 29 28 26 15 23 27 25 24

3. Link Prediction

3.1. Using Preferential Attachment Method (PAC)

For each pair of non connected node pair (Ni,Nj), PAC score is calculated by

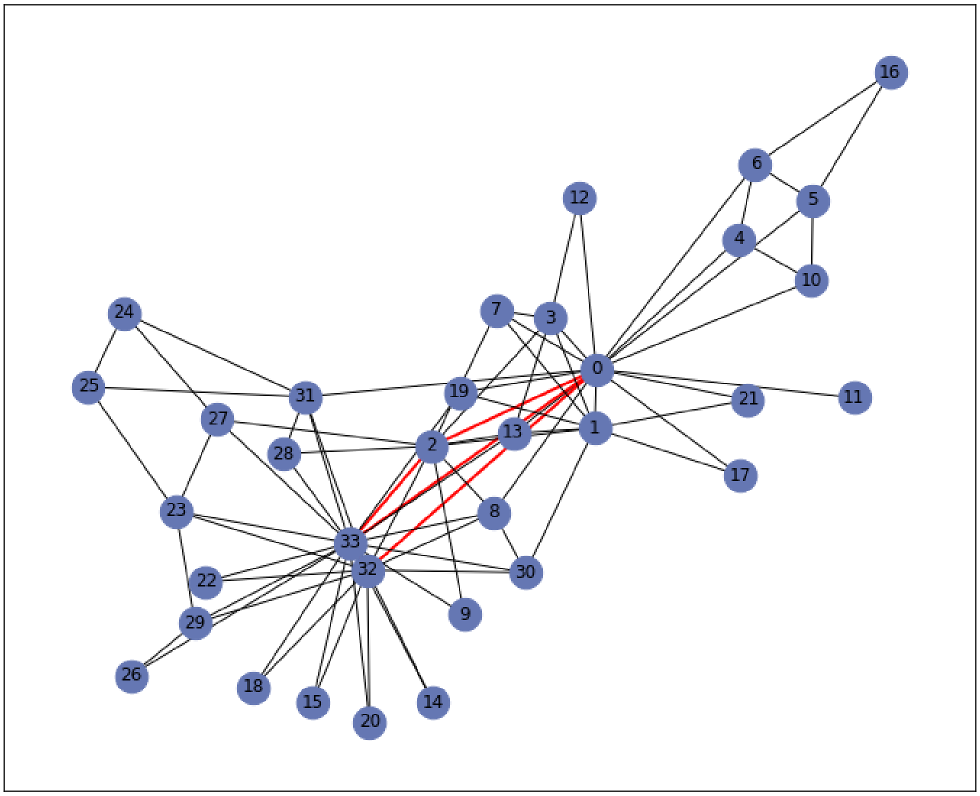
PAC Score of (Ni,Nj) = degree of (Ni)* degree of (Nj)

Observation:

Top 5 Link Predicted as per PAC score

Link Predicted	PAC Score
0:33	272
32:33	204
0:32	192
2:33	170
0:2	160

Visualization of Predicted Link:


16
6
12
5
4
10
24
7
3
0
25
31
19
21
11
27
13
1
2
28
17
23
8
33
32
30
22
29
9
26
18
15
20
14