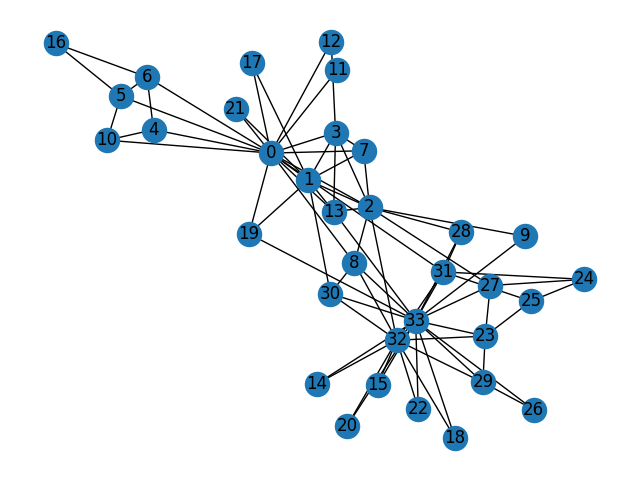
HW#1

2022711835 Junhee Kwon

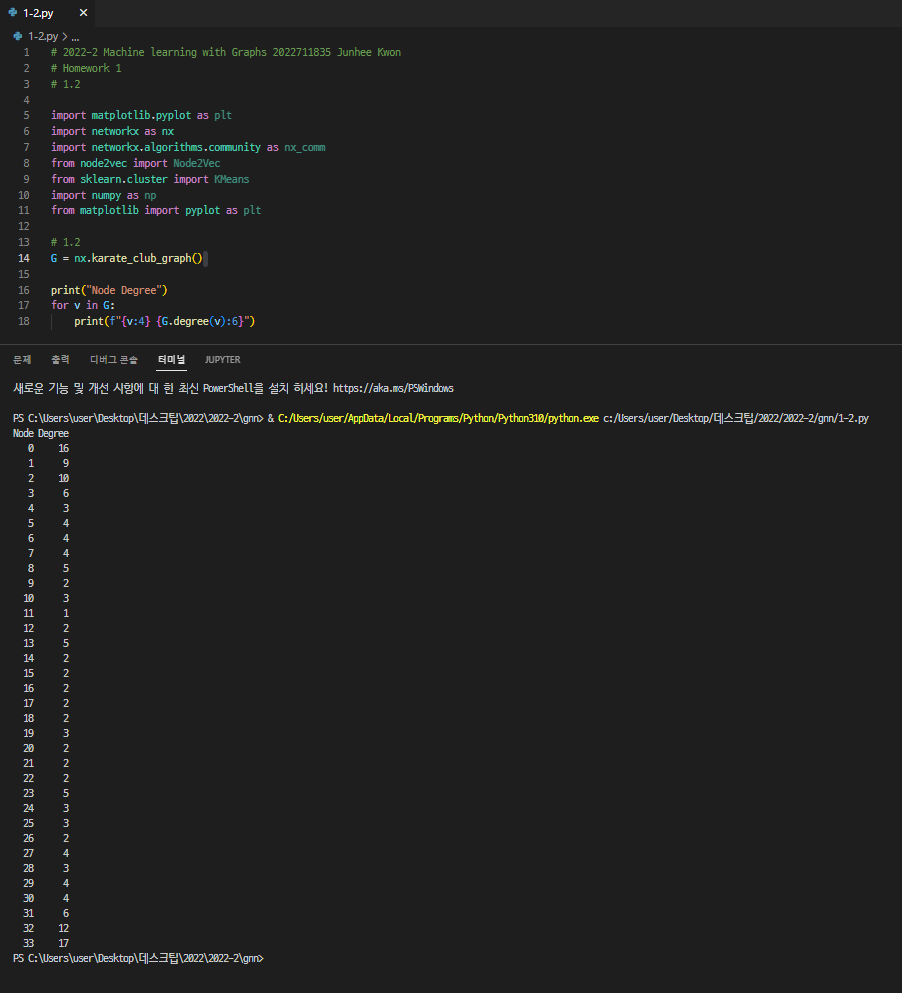
1. Data Preparation

Through matplotlib, I drew the graph of the karate club dataset.



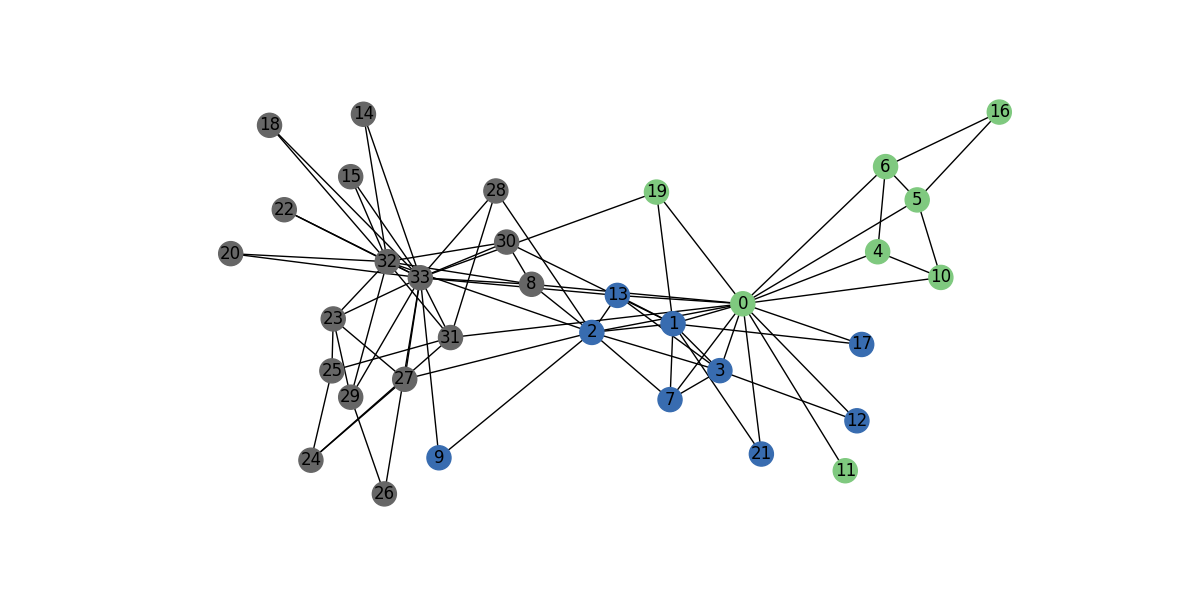
1. Degree Computation

By using the degree method of the network library, I could find the degree information of the nodes.



1. Community Detection

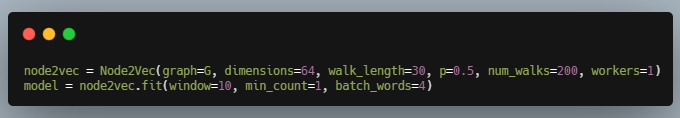
Through the *greedy\_modularity\_communities()* method of the networks library, I could cluster the nodes in 3 communities.

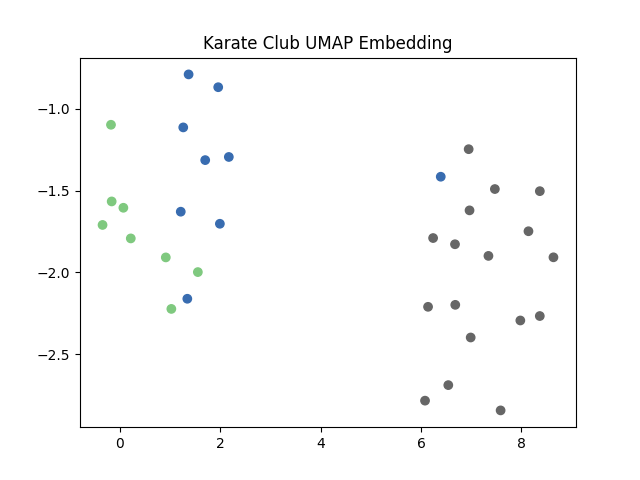


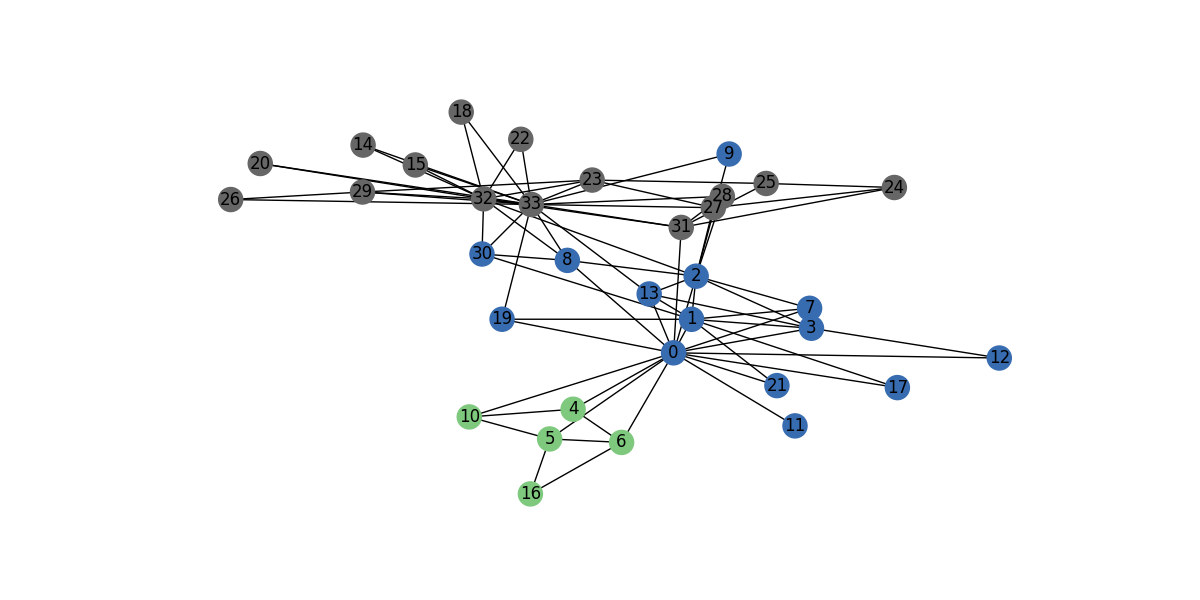
1. Learning node2vec with initial parameters

Based on the [GitHub - eliorc/node2vec: Implementation of the node2vec algorithm.](https://github.com/eliorc/node2vec) page, I set the initial values of the node2vec method.

You can check out the node2vec code with the initial values below.





1. Learning node2vec with your best parameters

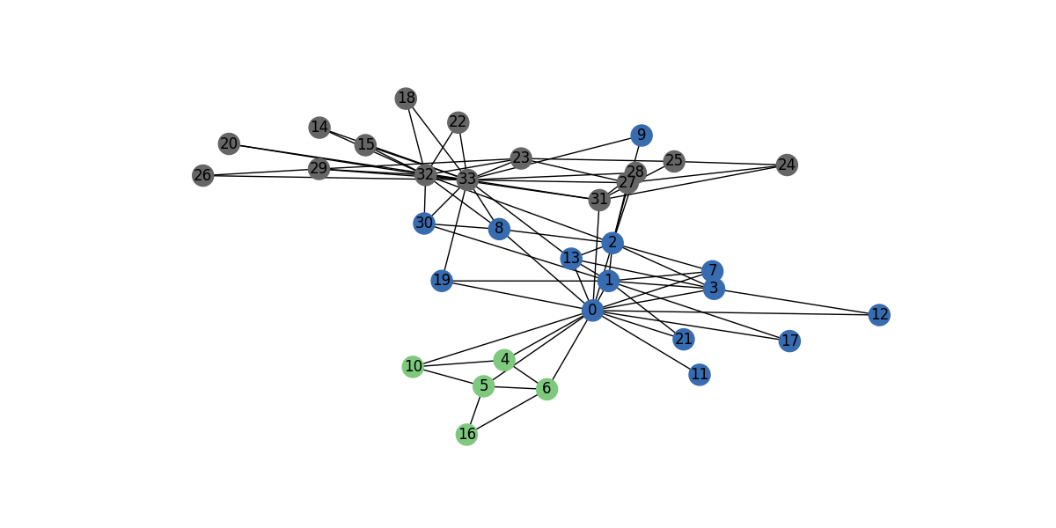
To get the better result than that of initial values, I tried to adjust the parameters.

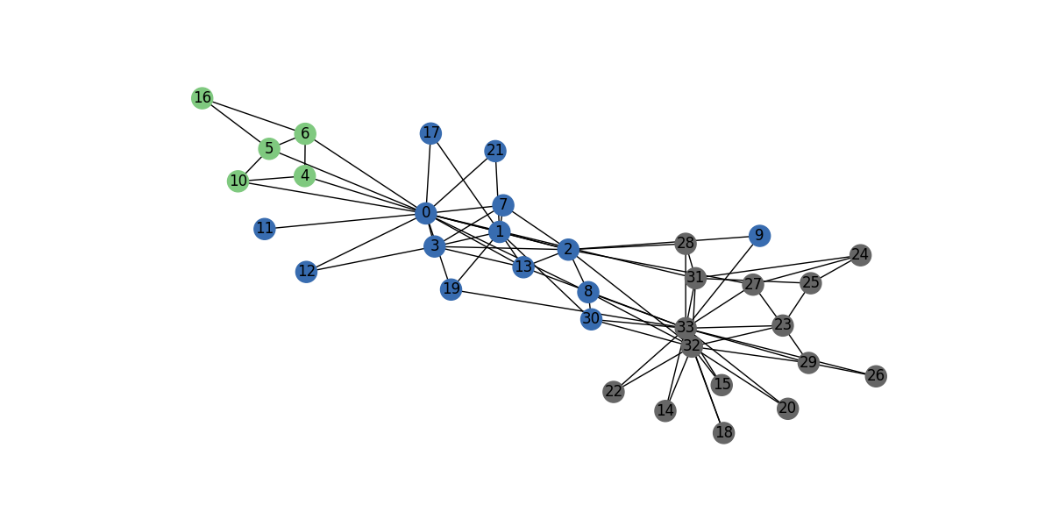
**Initial parameter**

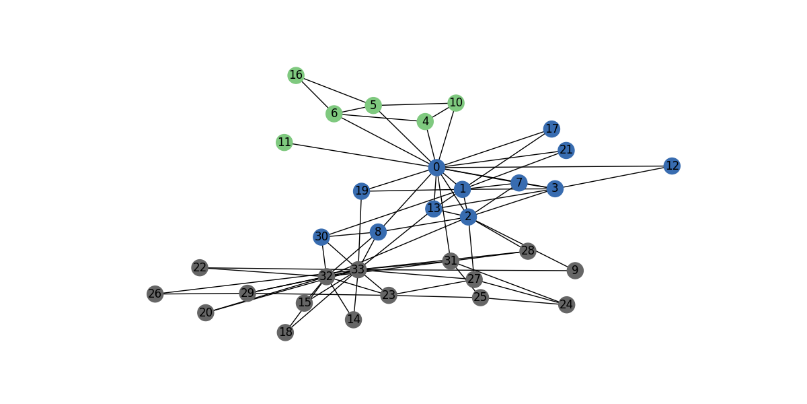
* p=1 (default)
* q=1 (default)
* dimensions=64
* walk-length=30
* num-walks=200
* workers=1 (for Windows)
* windows=10

**Adjusted parameter**

* p= 0.0001
* q= 0.99999
* dimensions=64
* walk-length=10
* num-walks=200
* workers=1 (for Windows)
* windows=10

1. Values of p and q
   1. Low p and high q case can explore the graph like BFS that considers the narrow region.
   2. High p and low q case can explore the graph like DFS that considers the broad region.
   3. Low p and high q case shows the lower error than other cases. To find the community, we can use the low p and high q.
   4. Low p & High q case’s graph
   5. High p & Low q case’s graph



1. For better result, adjusted the other parameters
   1. Change window value from 10 to 15: worse result
   2. num\_walks, reducing window, walk-length, dimension do not have effect to the result.