Data Science

Network Data Analysis

노기섭 교수

(kafa46@hongik.ac.kr)

Lecture Goals

- 노드(Node), 엣지(Edge), 가중치(Weight) 등 그래프 데이터의 핵심 개념
- 중심성 지표(차수, 매개, 고유벡터, 페이지랭크)의 의미와 해석
- 네트워크 데이터를 전처리하여 구조적 특징을 탐색 및 시각화
- 파이썬의 networkx 라이브러리를 이용한 분석 파이프라인

네트워크 데이터란?

네트워크 구성 요소

노드(Node)

- 분석 대상의 개별 객체. 사람, 도시, 제품 등 상황에 따라 다양한 의미를 가짐

엣지(Edge)

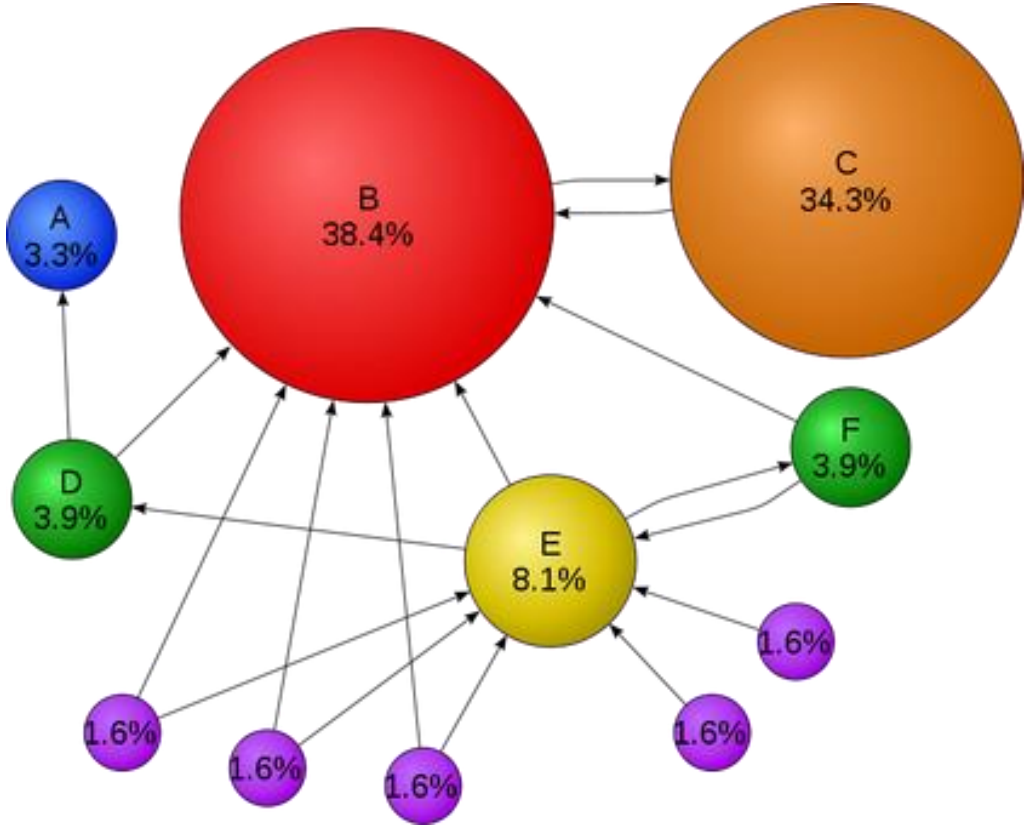
- 노드 간 관계를 나타내는 연결선.
- 방향성 여부(Directed/Undirected)에 따라 의미가 크게 달라짐
- 가중치가 포함되면 관계의 강도나 비용을 표현할 수 있음.

속성(Attribute)

- 노드(node), 엣지(edge)가 가진 부가 정보
- 예)
  - 사람 노드의 나이
  - 도시 간 이동 거리 등

네트워크 구성 요소

Concept of Google Search Engine



네트워크 데이터 구성

데이터 형태	설명	예시
엣지 리스트 (Edge List)	각 행이 두 노드의 연결을 의미	source, target, weight
인접 행렬 (Adjacency Matrix)	행/열이 노드를 나타내고 요소가 연결 여부를 표시	소규모 고정 네트워크
이벤트 로그 (Interaction Log)	시간 순서 이벤트를 관계로 해석	메시지 전송 기록, API 호출 로그

NetworkX 패키지

NetworkX 소개

- 데이터를 분석하고 시각화하기 위한 파이썬 대표 라이브러리
- 소셜 네트워크, 물류 네트워크, 통신망, 웹 링크 구조 등 객체 간의 관계 데이터를 표현하고 분석할 때 유용

NetworkX 주요 특징

- 데이터 구조: Graph, DiGraph, MultiGraph 등 다양한 형태 지원
- 분석 기능: 중심성, 커뮤니티 탐지, 경로 탐색, 연결 성분 분석 등
- 시각화 기능: matplotlib 또는 pyvis 등과 연동 가능
- 유연성: pandas, numpy, scipy 등과 자연스럽게 호환

NetworkX 패키지 - 주요 클래스

클래스	설명	예시
Graph()	무방향 그래프 (기본형)	친구 관계, 상호 협력
DiGraph()	방향 그래프	이메일 발송, 링크 구조
MultiGraph()	중복 엣지를 허용하는 무방향 그래프	두 사람 간 여러 거래
MultiDiGraph()	중복 엣지 + 방향 포함	반복 메시지, 교차 경로

NetworkX 패키지 - 주요 함수 및 파라미터

함수	주요 파라미터	설명
add_node(node, **attr)	node: 노드 이름 attr: 속성(딕셔너리 형태)	노드 추가 시 속성 지정 가능
add_edge(u, v, **attr)	u, v: 연결할 노드 attr: 엣지 속성 (예: weight)	엣지 추가 및 가중치 설정
from_pandas_edgelist(df, source, target, edge_attr=None, create_using=nx.Graph())	df: 데이터프레임 source/target: 컬럼 이름 edge_attr: 가중치 컬럼	pandas 엣지리스트로부터 그래프 생성
from_numpy_matrix(matrix)	matrix: 인접 행렬(numpy 배열)	인접 행렬로부터 그래프 구성
from_scipy_sparse_array(matrix, create_using=nx.Graph())	matrix: 희소 행렬 create_using: 그래프 타입	대용량 네트워크 처리용
degree(weight=None)	weight: 가중치 컬럼 지정	노드의 연결 수(차수) 계산
neighbors(node)	node: 기준 노드	인접 노드 리스트 반환
subgraph(nodes)	nodes: 선택 노드 리스트	부분 그래프 생성
to_pandas_edgelist(G)	G: 그래프	엣지 리스트를 pandas 데이터프레임으로 변환

NetworkX 패키지 - 간단 예제

```
# 의존성 패키지 설치
pip install networkx
```

```
import networkx as nx
import pandas as pd

# 엣지 리스트로 그래프 생성
df = pd.DataFrame({
    "source": ["A", "A", "B", "C"],
    "target": ["B", "C", "D", "D"],
    "weight": [3, 5, 2, 1]
})

G = nx.from_pandas_edgelist(
    df,
    source="source",
    target="target",
    edge_attr="weight",
    create_using=nx.DiGraph()
)

# 기본 정보 출력
print("노드 수:", G.number_of_nodes())
print("엣지 수:", G.number_of_edges())
print("노드 목록:", list(G.nodes()))
print("엣지 목록:", list(G.edges(data=True)))
```

```
# 실행 결과
노드 수: 4
엣지 수: 4
노드 목록: ['A', 'B', 'C', 'D']
엣지 목록: [('A', 'B', {'weight': 3}), ('A', 'C', {'weight': 5}),
            ('B', 'D', {'weight': 2}), ('C', 'D', {'weight': 1})]
```

네트워크 데이터 전처리

네트워크 데이터가 너무 큰 경우

- 메모리 & 계산 자원 한계
  - 일부 네트워크 알고리즘은 시간 복잡도가 매우 높음.
- 노이즈 제거 & 분석 품질 향상
  - 종종 스팸 노드, 의미 없는 연결, 오류 데이터가 섞여 있음
- 해석 가능한 구조로 단순화
  - 현실의 네트워크는 너무 복잡해서 그대로 보면 무질서한 거대한 거미줄처럼 보임.
- 알고리즘 적용을 가능하게 조정
- 시각화 가능한 수준으로 줄이기 위해
  - 노드가 많으면 그냥 구름처럼 보임 ㅠ

네트워크 데이터 전처리 - 실습

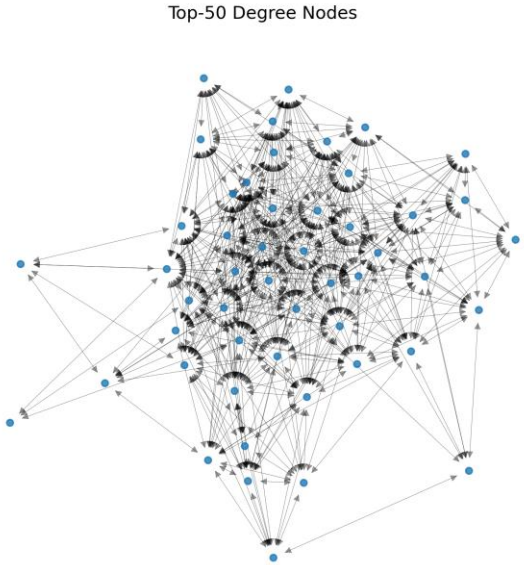
데이터셋

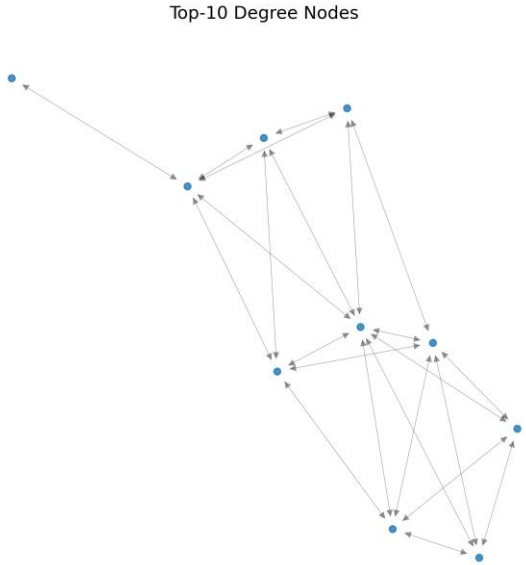
- 실습 데이터 추천: Email Networks (Kaggle)

- 직접 다운로드: email-Enron.mtx (Matrix Market 형식, 희소행렬 데이터 저장 형식)

실습 코드

https://www.deepshark.org/courses/data_science/w/10_network_data_analysis#mtx_preprocessing



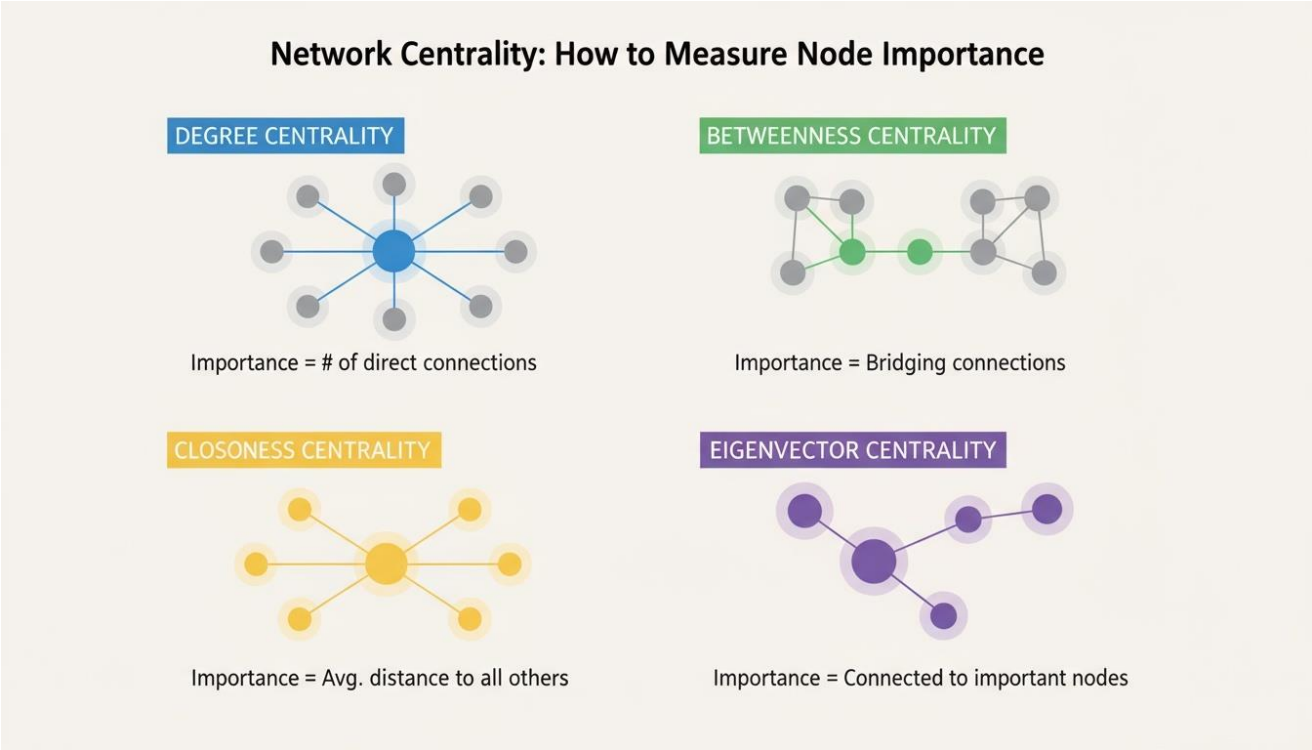


Network 중심성 분석

중심성(Centrality)

네트워크에서 중심성(Centrality)

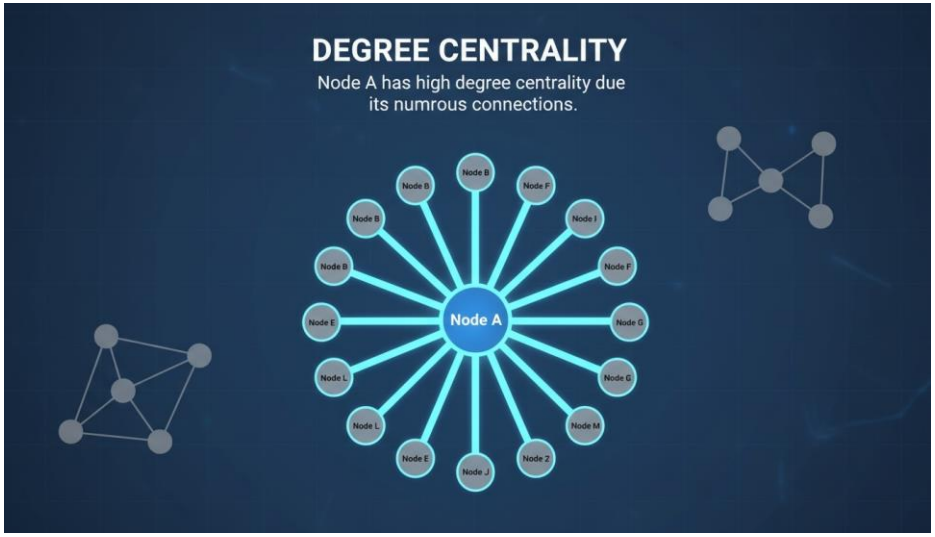
- "누가 중요하고 영향력이 있는가?"를 수치화하는 방법
- 초점을 두는 대상(연결 수, 경로, 확산 등)이 다름.



Degree 중심성 (Degree Centrality)

Degree 중심성 (Degree Centrality)

- 한 노드가 얼마나 많은 엣지와 연결되어 있는지를 나타내는 가장 단순한 형태의 중심성
- 연결 수가 많을수록 중심성 ↑
- 방향 그래프는 in-degree(들어오는 엣지 수), out-degree(나가는 엣지 수)로 구분하여 해석
- 예시: SNS에서 친구가 많은 사용자, 도로망에서 많은 도로와 연결된 교차로.



$$C_D(v) = \frac{\deg(v)}{N-1}$$

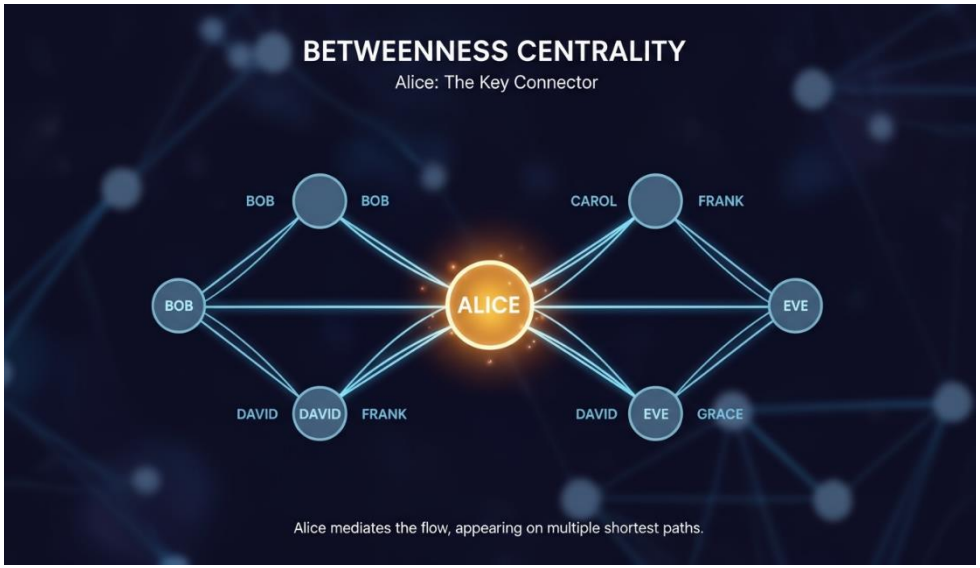
$\deg(v)$: 노드 v의 연결수

N: 전체 노드 수

Betweenness 중심성 (Betweenness Centrality)

Betweenness 중심성 (Betweenness Centrality)

- 특정 노드가 다른 노드 쌍 간의 최단 경로 위에 얼마나 자주 등장하는가를 측정
- 네트워크 내에서 정보나 자원의 흐름을 중개하는 역할을 하는 노드를 식별할 때 유용
- 예시: 물류 네트워크의 주요 허브, 조직 내 중개 관리자 등



$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

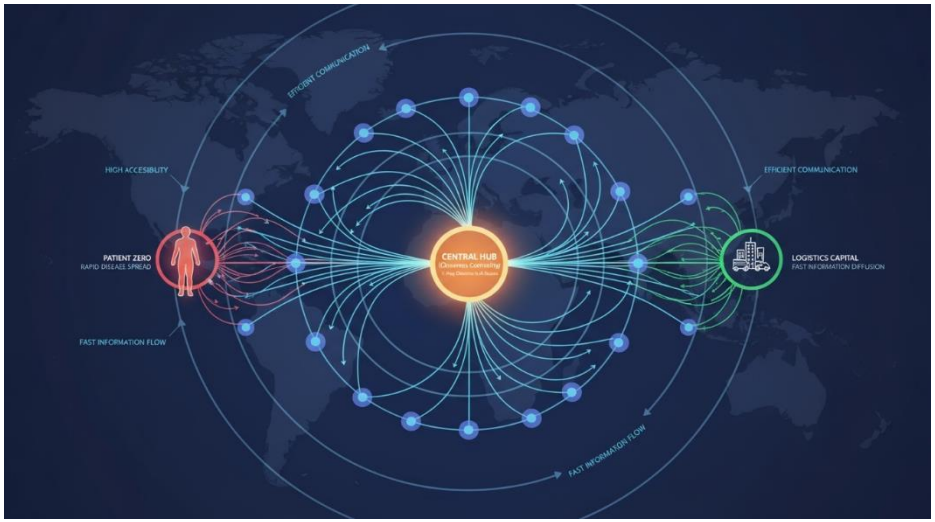
σ_{st}: 노드 s에서 t로 가는 최단경로 개수

$\sigma_{st}(v)$: 그 경로 중 노드 v가 포함된 경로의 개수

Closeness 중심성 (Closeness Centrality)

Closeness 중심성 (Closeness Centrality)

- 한 노드에서 다른 모든 노드까지의 평균 거리의 역수로 정의된다.
- 네트워크 전체와의 접근성(Accessibility) 혹은 정보 확산 속도를 나타낸다.
- 값이 높을수록 다른 노드로 빠르게 접근할 수 있다.
- 예시: 감염병 확산 네트워크에서 전염이 빠른 사람, 전국 물류망의 중심 도시.



$$C_C(v) = \frac{N-1}{\sum_{t \neq v} d(v,t)}$$

$d(v,t)$: 노드 v의에서t까지 최단 거리

N: 전체 노드 수

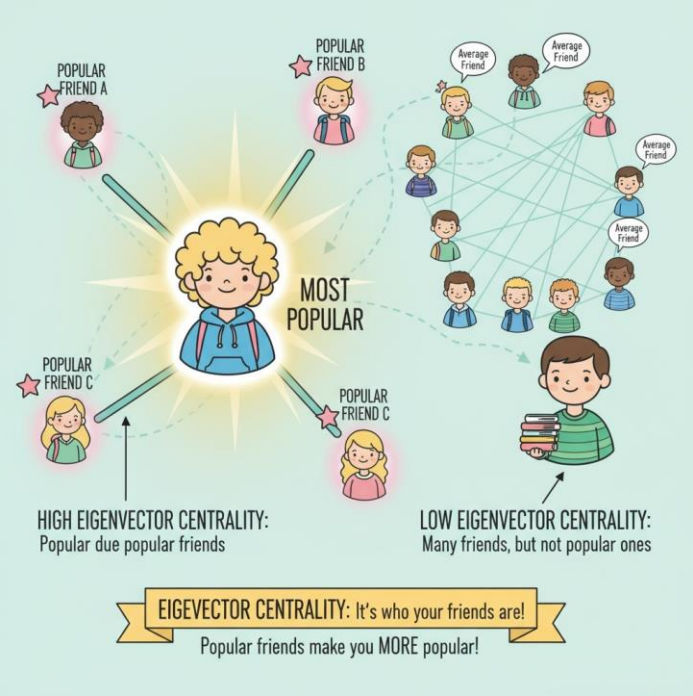
Eigenvector 중심성 (Eigenvector Centrality)

- 학술 인용 네트워크 : 영향력 있는 연구자 탐색
- SNS 분석 : 인플루언서 추천
- 범죄/외교 네트워크 : 핵심 인물 탐색

Eigenvector 중심성

- 노드의 중요도가 연결된 이웃 노드의 중요도에 의해 결정된다는 아이디어를 기반

- 단순히 많이 연결된 노드가 중요한 것이 아니라, 영향력 있는 이웃과 연결된 노드가 더 중요



$$x_v = \sum_{t \in N(v)} A_{vt} x_t$$

x_v: 노드 v의 중심성 값

A_{vt}: 인접행렬의 원소
(연결되면 1, 아니면 0)

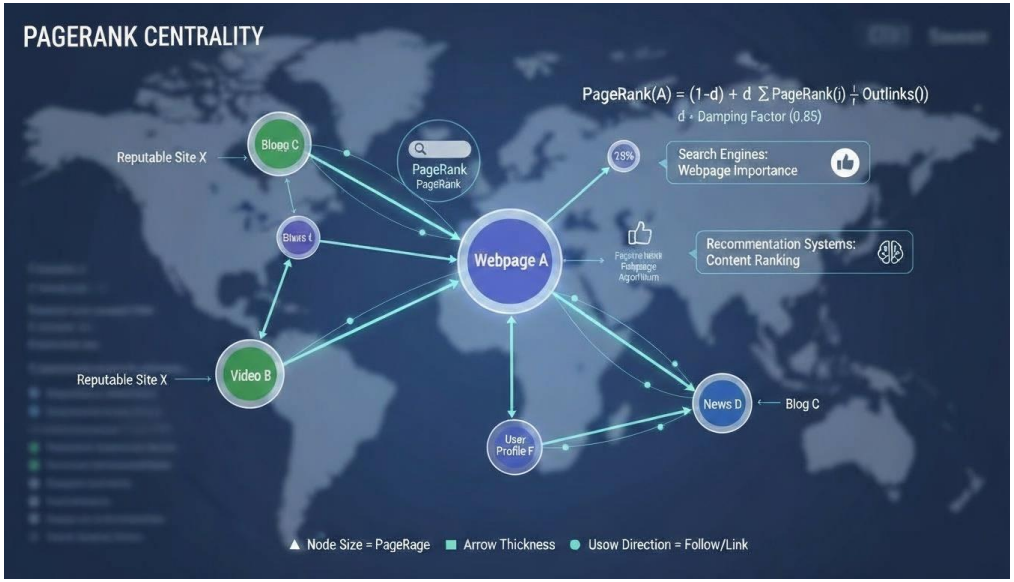
$N(v)$: 노드 v에 연결된
이웃 노드 집합

Eigenvector & Eigenvalue 관련 동영상
https://youtu.be/PFDu9oVAE-g?si=oCUZjVxbyMAgdH2I

PageRank 중심성 (PageRank Centrality)

PageRank 중심성 (PageRank Centrality)

- 사용자가 링크를 따라 무작위로 이동한다고 가정하여 각 노드의 중요도를 확률적으로 계산
- 단순 연결 수뿐 아니라 링크의 품질과 방향성을 함께 고려
- 예시: 검색 엔진의 웹페이지 중요도 계산, 추천 시스템의 콘텐츠 순위 평가 등



$$PR(v) = \frac{1-\alpha}{N} + \alpha \sum_{t \in M(v)} \frac{PR(t)}{L(t)}$$

α: damping factor (감쇄 계수), 일반적으로 0.85

$M(v)$: 노드 v로 연결된 이웃 노드 집합

$L(t)$: 노드 t의 출발 엣지 수 (out-degree)

N: 전체 노드 수

PageRank 관련 동영상: https://youtu.be/meonLcN7LD4?si=J389VUW_v8xAZK2T

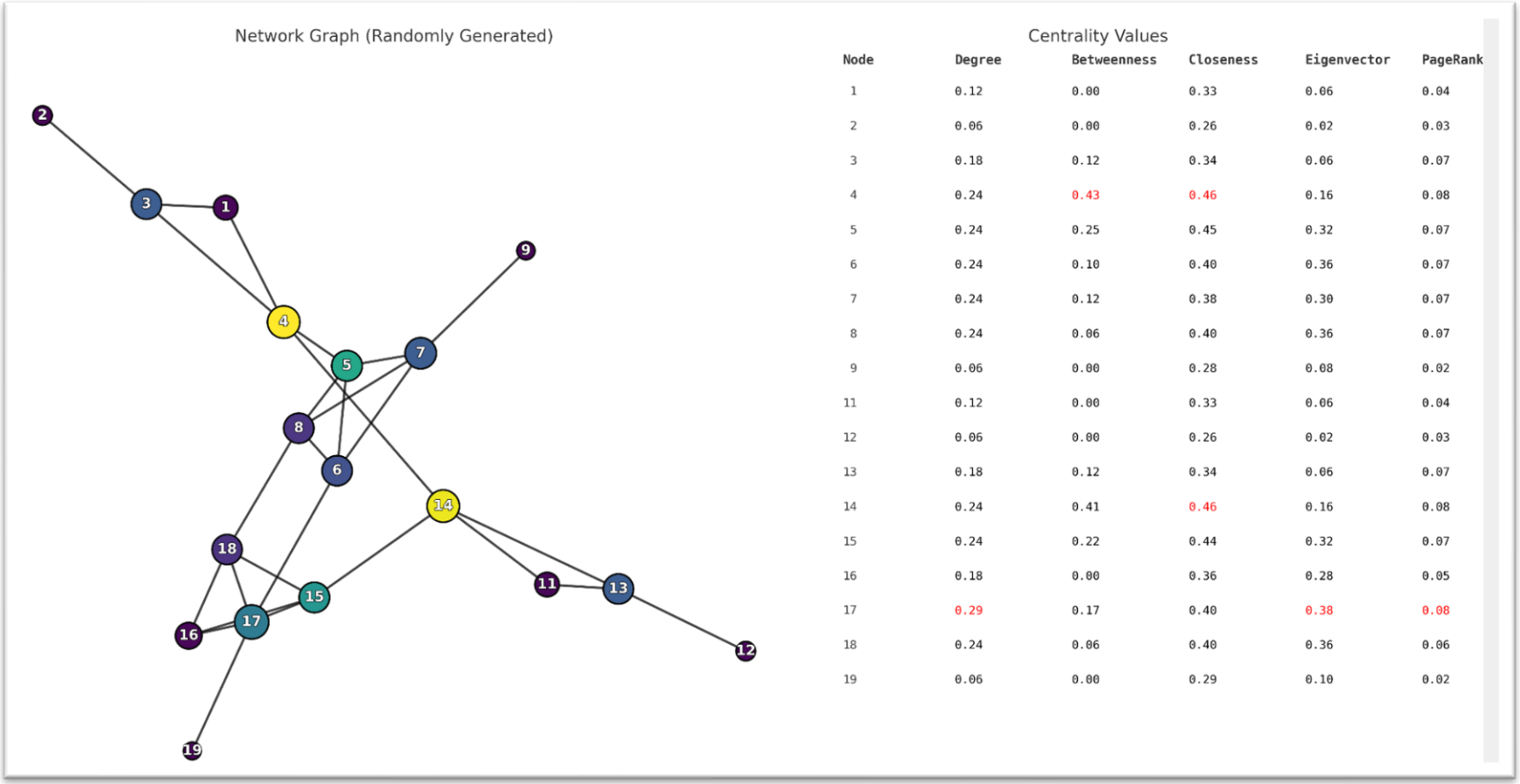
Centrality 요약 정리

중심성	계산 방식	해석 포인트	활용 사례
Degree (차수)	특정 노드에 연결된 엣지 수	단순하지만 직관적인 활동성 지표	SNS 친구 수, 도로 교차점 연결
Betweenness (매개)	최단경로 상에 등장하는 비율	정보 흐름을 통제하는 노드 탐색	물류 허브, 중개 플랫폼
Closeness (근접)	다른 노드까지의 평균 거리 역수	네트워크 전체와 얼마나 가까운지	전염 확산 속도, 서비스 커버리지
Eigenvector (고유벡터)	영향력 있는 이웃과 연결 정도	"영향력 있는 이웃" 효과 반영	인플루언서 탐색, 학술 네트워크
PageRank	확률적 무작위 이동 기반 점수	웹 페이지 랭킹, 추천 시스템	검색 엔진, 콘텐츠 추천

중심성 계산 실습

실습 코드

https://www.deepshark.org/courses/data_science/w/10_network_data_analysis#centrality_comparison



Centrality Values

Node	Degree	Betweenness	Closeness	Eigenvector	PageRank
1	0.12	0.00	0.33	0.06	0.04
2	0.06	0.00	0.26	0.02	0.03
3	0.18	0.12	0.34	0.06	0.07
4	0.24	0.43	0.46	0.16	0.08
5	0.24	0.25	0.45	0.32	0.07
6	0.24	0.10	0.40	0.36	0.07
7	0.24	0.12	0.38	0.30	0.07
8	0.24	0.06	0.40	0.36	0.07
9	0.06	0.00	0.28	0.08	0.02
11	0.12	0.00	0.33	0.06	0.04
12	0.06	0.00	0.26	0.02	0.03
13	0.18	0.12	0.34	0.06	0.07
14	0.24	0.41	0.46	0.16	0.08
15	0.24	0.22	0.44	0.32	0.07
16	0.18	0.00	0.36	0.28	0.05
17	0.29	0.17	0.40	0.38	0.08
18	0.24	0.06	0.40	0.36	0.06
19	0.06	0.00	0.29	0.10	0.02

중심성 시각화

중심성 시각화 (Centrality Visualization)

- 노드 크기를 중심성 값에 비례하도록 조정
- 색상으로 중심성 등급(상/중/하)을 구분
- 그래프 레이아웃을 선택(Force-directed, Circular, Geographical 등)해 정보 전달력을 높임

네트워크 시각화 전략

시각화 유형	장점	단점	주요 도구
정적 시각화	보고서나 논문에 삽입하기 용이 기본 구조를 빠르게 전달 가능	복잡한 네트워크 세부 관계 파악이 어려움	matplotlib, seaborn, geopandas, ggplot
인터랙티브 시각화	노드 상세 정보 조회 가능 확대/축소 등 탐색적 분석 가능	배포 환경에 따라 성능 및 호환성 고려 필요	pyvis, plotly, bokeh, kepler.gl, Gephi

pyvis 이용한 시각화

Pyvis 패키지

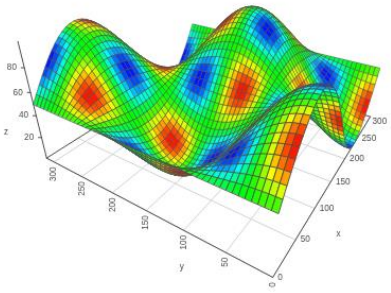
- 즉시 인터랙티브: NetworkX 그래프를 곧바로 vis.js 기반 HTML로 변환, 드래그·확대·툴팁 기본 제공
- 옵션 제어 용이: JSON으로 물리 엔진/레이아웃/색상/라벨을 직관적으로 커스터마이즈
- 오프라인 OK: cdn_resources="in_line" 설정으로 방화벽 환경에서도 단일 HTML만 배포하면 끝
- 임베딩 친화적:
  - Flask, Django, Streamlit, Jupyter 어디든 iframe/HTML로 삽입 가능
  - 데모 공유 간편
- 시각화 팁: 노드 크기=PageRank, 색상=Betweenness처럼 핵심 지표와 스타일을 매핑해 인사이트 전달

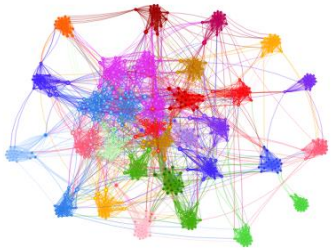
편리성

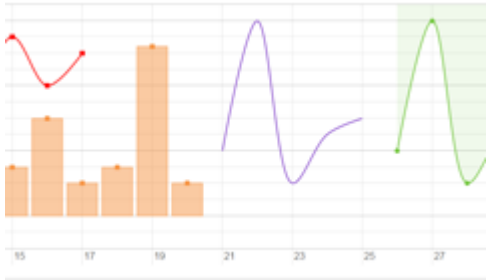
- 내부적으로 vis.js를 포함한 HTML을 생성
- 별도의 JS/CSS 파일을 연결할 필요 없이 net.generate_html() 혹은 net.show()로 바로 동작하는 단일 HTML

참고: 인터렉티브 javascript 모듈: vis.js

- **공식 정보:**
  - 공식 웹사이트 링크 (https://visjs.org)
- **주요 특징:**
  - 고성능 렌더링, 물리 시뮬레이션, 풍부한 인터랙션
  - 커스터마이징 용이, 다중 클러스터 렌더링, 연동 가능한 통계/필터
- **PyVis: Python에서 Vis.js 사용:**
  - NetworkX 친화성
  - 설정 난이도 ↓
  - 오프라인 배포
  - 웹 임베딩 용이







pyvis 이용한 인터랙티브 시각화 실습

실습 코드

https://www.deepshark.org/courses/data_science/w/10_network_data_analysis#interactive_visualization

인터렉티브 시각화 결과

https://www.deepshark.org/courses/data_science/media/weeks/files/chap10/interactive_network.html

커뮤니티 탐지와 서브네트워크

커뮤니티 탐지

네트워크 분석에서 자주 묻는 핵심 질문

- "이 네트워크는 몇 개의 집단으로 나뉘는가?"

커뮤니티 탐지

- 네트워크를 집단 단위로 분리하여각 그룹의 구조적 특징과 역할을 이해하는 데 활용
- 대표 알고리즘
  - Louvain (별도 설치)
  - NetworkX 제공 모듈: networkx.community → Girvan–Newman, Label Propagation 등 제공 (https://networkx.org/documentation/stable/reference/algorithms/community.html)

다양한 실습용 그래프 데이터셋

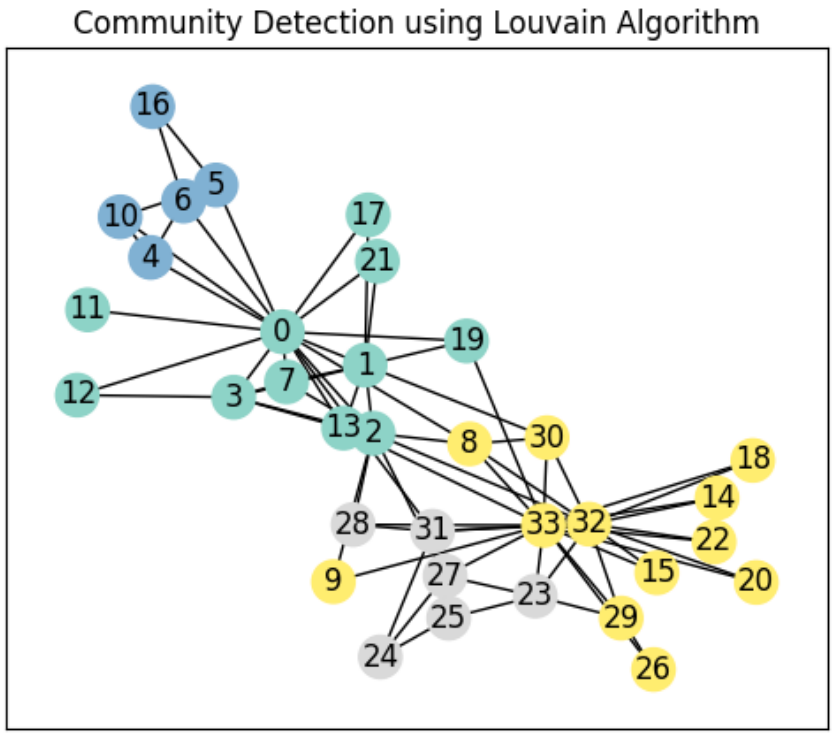
- 스탠포드 SNAP (Large Network Dataset Collection)
- (분석 및 활용)
  - 커뮤니티로 분류한 뒤, 각 그룹의 평균 중심성을 비교하면그룹 간 영향력 차이 파악 등

커뮤니티 탐지 실습

실습 코드

https://www.deepshark.org/courses/data_science/w/10_network_data_analysis#subnetwork

```
# Louvain 알고리즘을 이용하려는 경우 -> 별도 패키지 설치
pip install python-louvain
```




```
# 실행 결과
Community 0: [0, 1, 2, 3, 7, 11, 12, 13, 17, 19, 21]
Community 1: [24, 25, 28, 31]
Community 2: [4, 5, 6, 10, 16]
Community 3: [8, 9, 14, 15, 18, 20, 22, 23, 26, 27, 29, 30, 32, 33]
```

실습해볼 만한 주제 (Topics)

실습 1: 소셜 네트워크 분석

- 데이터: 동아리 활동 로그 또는 SNS 친구 목록(익명화).
- 목표:
  - 엣지 리스트를 그래프로 변환하고 기본 통계(노드 수, 엣지 수, 평균 차수)를 계산한다.
  - 중심성 지표 3가지 이상을 계산하여 상위 10개 노드를 도출한다.
  - 커뮤니티 탐지 결과를 시각화하고, 그룹별 특징을 정리한다.

실습 2: 교통 네트워크 분석

- 데이터: 도시별 버스/지하철 노선, 물류 창고 간 이동 경로 등.
- 목표:
  - 이동 경로 데이터를 방향 그래프로 구성하고, 경로 가중치를 설정한다.
  - 혼잡도(매개 중심성)와 접근성(근접 중심성)이 높은 노드를 비교 분석한다.
  - 인터랙티브 시각화로 경로 탐색 시나리오를 설명한다.

Homeworks Briefing (with SNAP Dataset)

Homeworks

https://www.deepshark.org/courses/data_science/w/10_network_data_analysis#homework_briefing

수고하셨습니다 ..^^..