

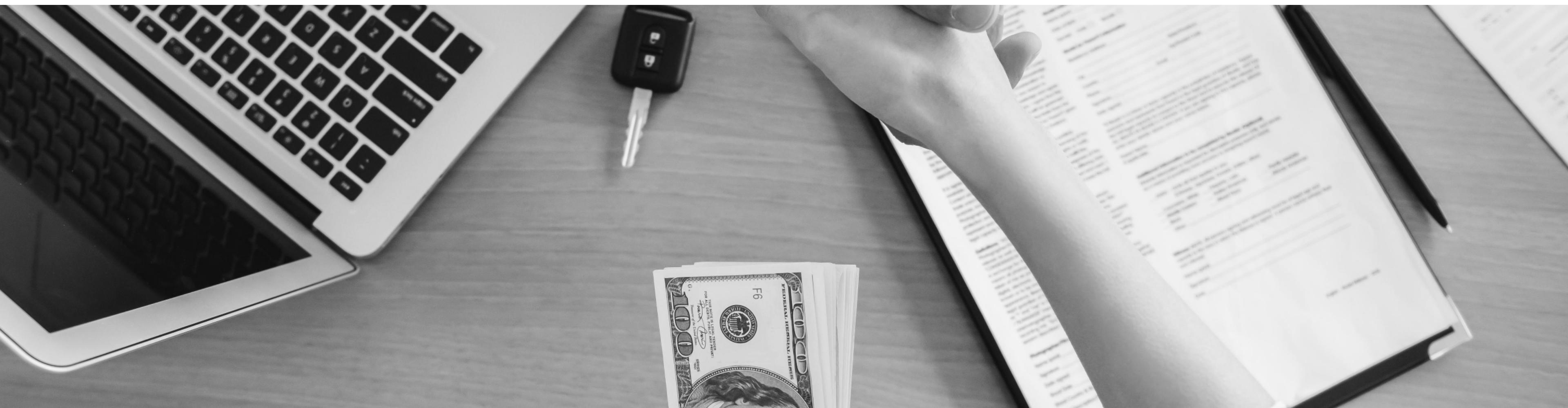
군집분석

Cluster Analysis

For NLP study of DNA



발표자 : 조성민



Introduce



- 2019.03
 - 수원대학교 19학번 데이터과학부 입학
- 2020.02 ~ 2021.09
 - 군 복무
- 2022.09 ~
 - 학부 연구생 (prof. 안홍렬)
 - 블로그 : <https://smcho1201.tistory.com/>
 - 깃허브 : <https://github.com/SeongminCC>

- ▶ 정인호, 김동현, 조성민, 안홍렬. (2022). Comparison of lesion segmentation deep learning models according to medical image types 한국정보과학회 학술발표논문집
- ▶ 조성민, 서형준. [2022] University of Suwon intramural competition for KLUE data. 대상 수상
- ▶ 조성민, 김시은, 김은수, 정인호, 차유진 [2022]University of Suwon competition for Foundation. 우수상 수상

Index



01. 텍스트 유사도



02. word2vec



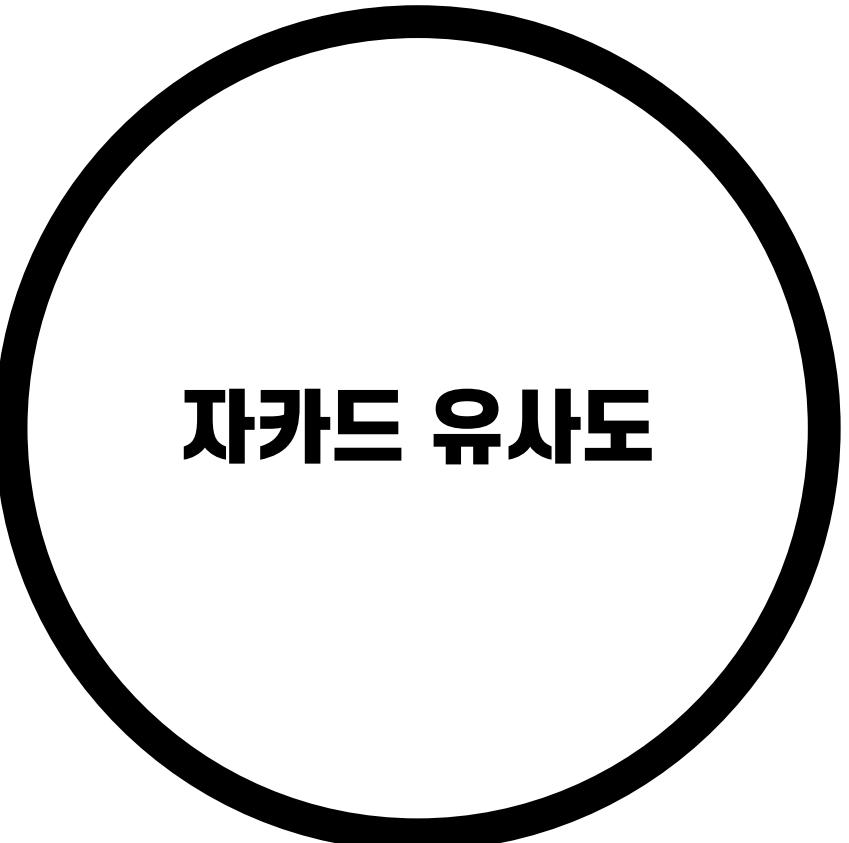
03. 계층적 군집화



04. 비계층적 군집화

01. 텍스트 유사도

텍스트가 얼마나 유사한지 표현하는 방식



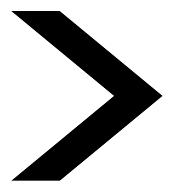
자카드 유사도



코사인 유사도

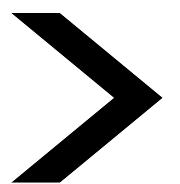
자카드 유사도

"Thick like a man of action and act like man of thought."



[['think', 'like', 'a', 'man', 'of', 'action', 'and', 'act',
'like', 'man', 'of', 'thought', '.']]

"Try no to become a man of success but rather try to become a man of value."



[['try', 'no', 'to', 'become', 'a', 'man', 'of', 'success',
'but', 'rather', 'try', 'to', 'become', 'a', 'man',
'of', 'value', '.']]

교집합

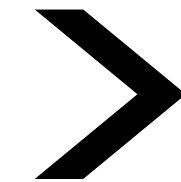


합집합

0.2222..

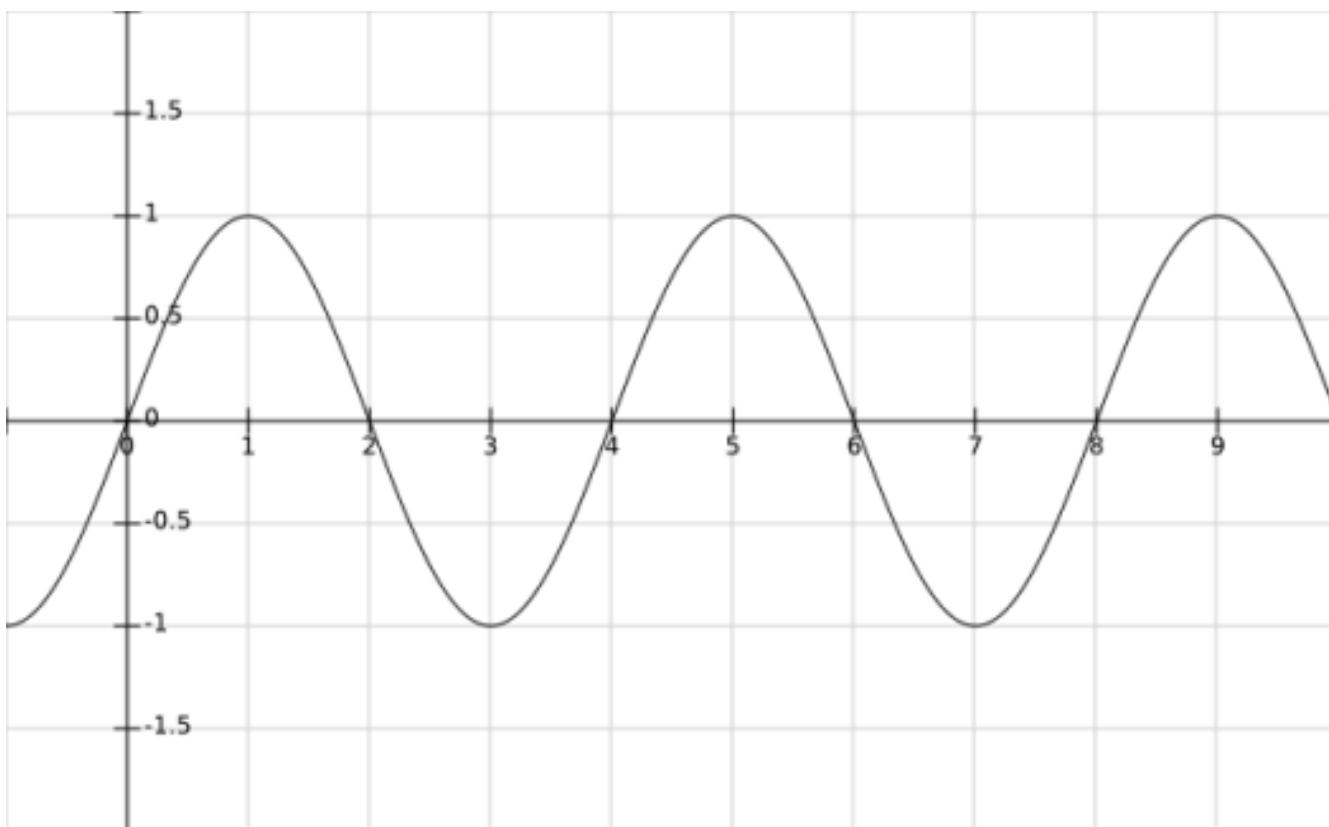
코사인 유사도

"Thick like a man of action and act like man of thought."



TF-IDF

([[0.28050841, 0.28050841, 0.28050841, 0., 0.,
0., 0., 0., 0.56101683, 0.42666776,
0., 0., 0.33134558, 0., 0.,
0.28050841, 0.28050841, 0., 0., 0.]])



Theta : 0 \rightarrow 유사도 : 1

Theta : pi \rightarrow 유사도 : -1

02. word2vec

워드 임베딩의 종류 중 하나, 단어를 벡터로 변환

why use?

- 벡터로 바꾸어야 유사도 계산이 가능

what?

- 얇은 신경망 (2개의 layer로 존재하는 신경망)
- distributed representation (분산표현)
- 2013년에 만들어진 임베딩 벡터 알고리즘

분산표현

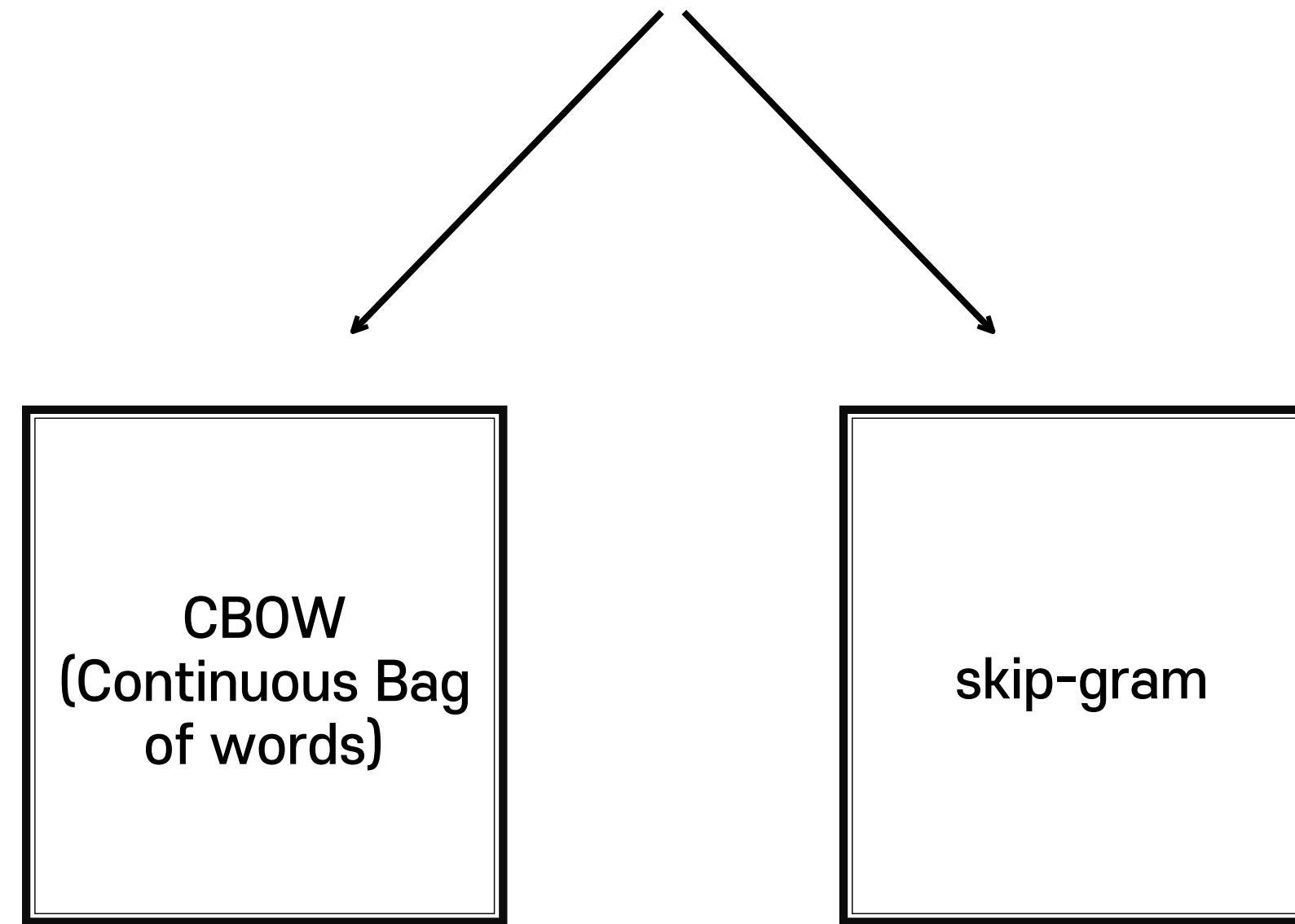
비슷한 위치에 등장하는 단어들은 비슷한 의미를 가짐

one-hot encoding > [0, 0, 1, 0, 0]

word2vec > [0.5, 0.3, 0.6, 0.2, 0.1]

word2vec의 원리

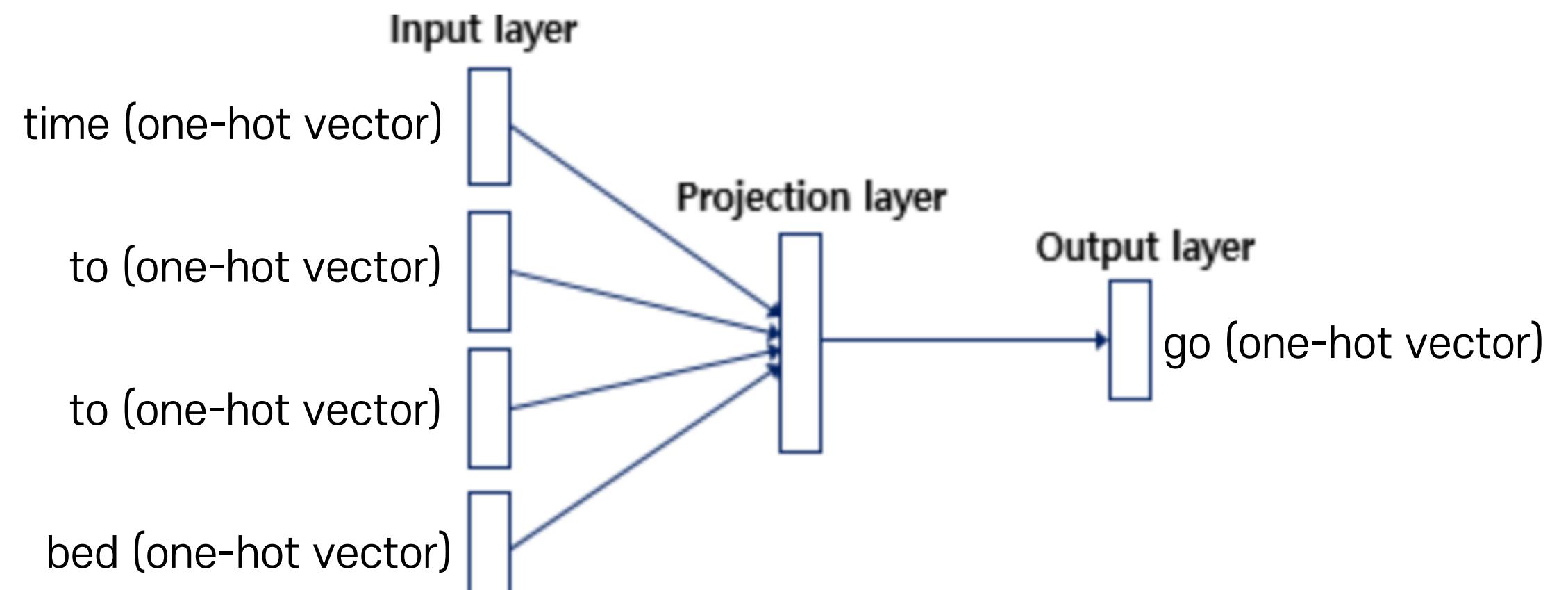
"Its time to go to bed" => {"Its", "time", "to", "go", "to", "bed"}



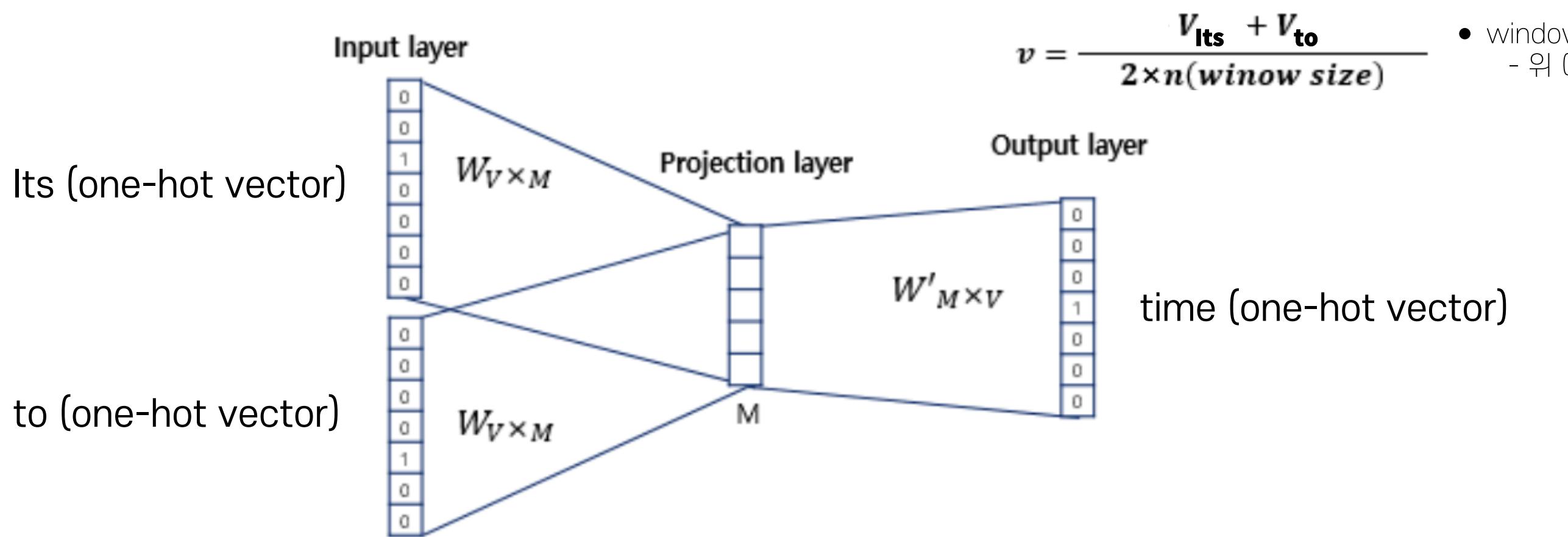
CBOW

주변 단어를 통해 중간 단어를 예측

Its time to go to bed



$$\begin{array}{c}
 \mathbf{x} \quad \times \quad \mathbf{W}_{V \times M} \quad = \quad \mathbf{v} \\
 \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 0.5 & 2.1 & 1.9 & 1.5 & 0.8 \\ 0.8 & 1.2 & 2.8 & 1.8 & 2.1 \\ 2.1 & 1.8 & 1.5 & 1.7 & 2.7 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} = \begin{bmatrix} 2.1 & 1.8 & 1.5 & 1.7 & 2.7 \end{bmatrix} \\
 \text{lookup table}
 \end{array}$$



$$v = \frac{V_{\text{its}} + V_{\text{to}}}{2 \times n(\text{window size})}$$

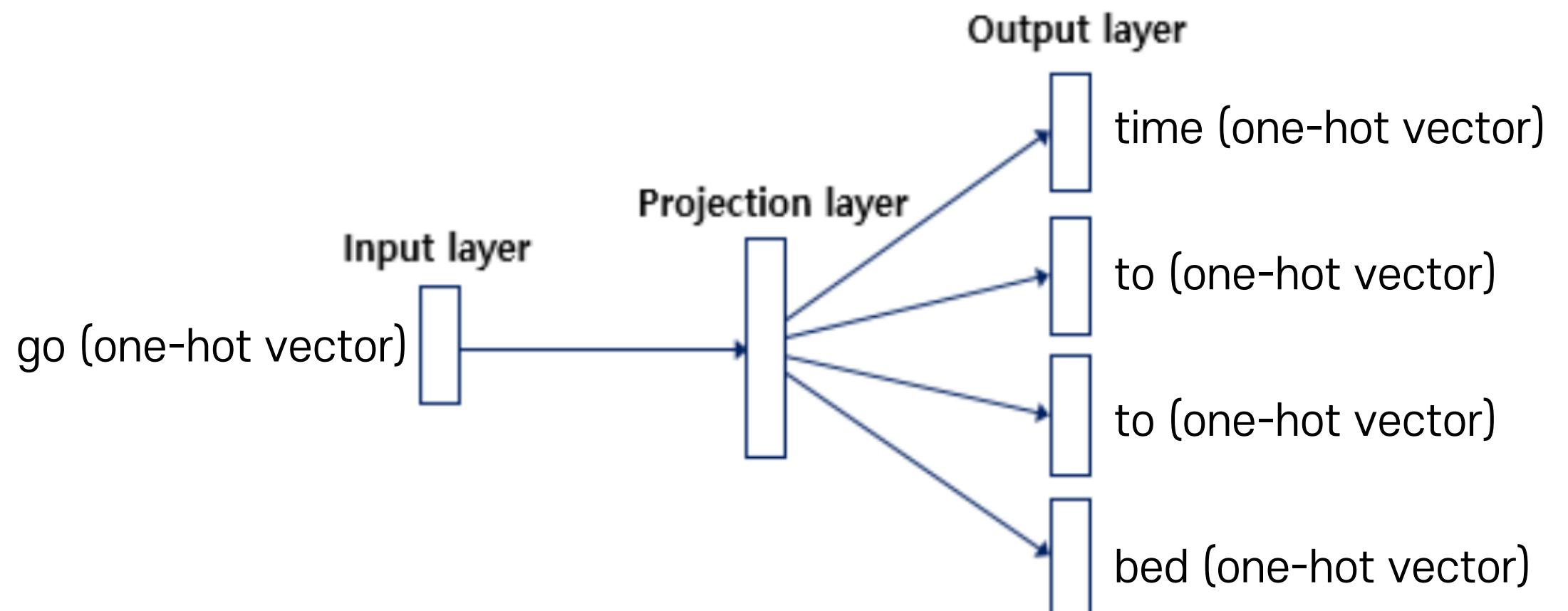
- window : 주변 단어의 개수
- 위 예시의 경우 window : 1

- projection layer 차원은 M 고정
 - 차원이 높아지면 계산량이 크게 증가
- 워드 임베딩 후의 차원 : M

Skip-gram

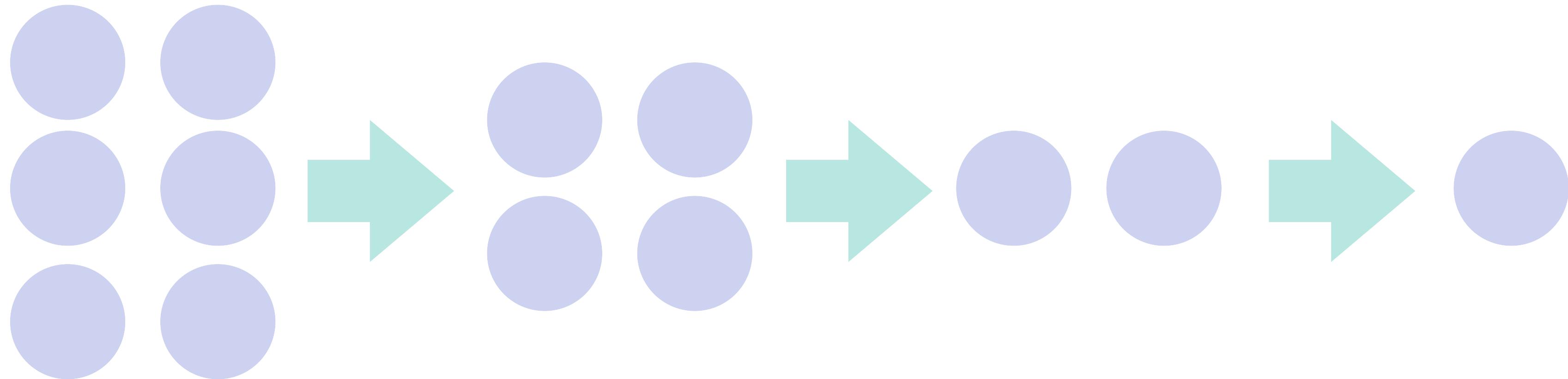
중심 단어에서 주변 단어를 예측

Its time to go to bed



03. 계층적 군집화

가까운 대상끼리 순차적으로 군집을 뿜어나아감



계층적 군집화를 하기 위해 군집간의 거리를 측정해야 한다.

- **비계층적 거리 측정법**

계층적, 비계층적 군집화에 모두 사용됨

단점 : 계층적 거리 측정법에 비해 계산량이 많음

- 중심거리
- 단일거리
- 완전거리
- 평균거리

- **계층적 거리 측정법**

계층적 군집화에서만 사용 가능

- 중앙값 거리
- 가중 거리
- 와드 거리

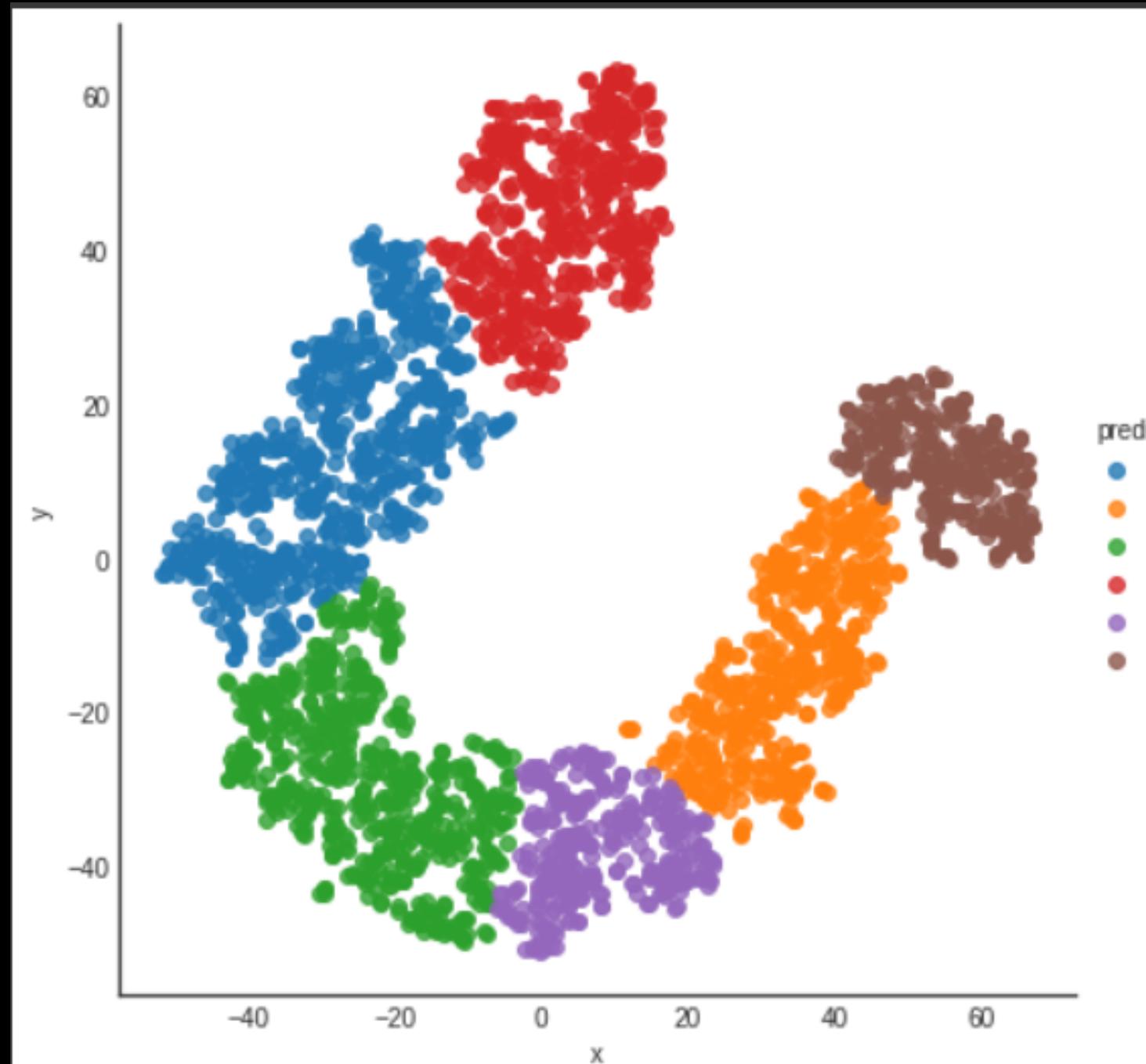
코드 예제

ward 거리를 사용한 계층적 군집화

```
from sklearn.cluster import AgglomerativeClustering  
  
ward = AgglomerativeClustering(n_clusters=6, linkage='ward')  
predict = ward.fit_predict(df)  
predict  
  
results = df  
results[ 'predict' ] = predict  
results[0:10]
```

	x	y	predict
어릴	23.151413	-17.665709	1
때	58.077744	11.722444	5
보	61.714249	17.495483	5
고	61.205811	16.189301	5
지금	60.398102	9.636668	5
다시	61.829369	16.032358	5
봐도	61.341206	8.923100	5
재밌	66.877228	4.089137	5
어요	65.897354	3.070244	5
ㅋㅋ	65.895126	0.682670	5

코드 예제



ward 거리기준으로 합체 군집화를
통해 최종적으로 6개의 군집을 이룸

04. 비계층적 군집화

랜덤하게 군집을 묶어나아감

- 나눌 cluster의 개수를 미리 지정함
- 계층적 군집화에 비해 계산 복잡도가 작음 -> 대량의 데이터에 유리

k-means : 대표적인 비계층적 군집화 알고리즘

K-means 클러스트링 알고리즘 장단점

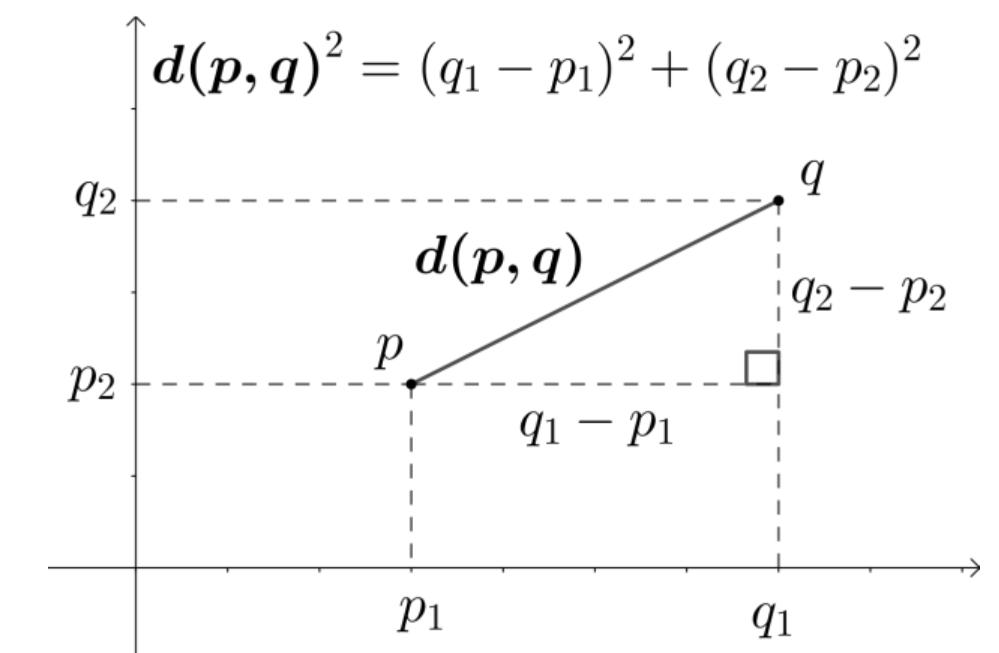
- **장점**

- 직관적이고 구현이 쉬움
- 대용량 데이터에 적용 가능
- 수렴성 보장

- **단점**

- 초기값에 민감함 (클러스트의 개수)
- 이상치에 영향을 받음 (Euclidean Distance를 기반으로 하기 때문)
- 그룹 내 분산 구조를 반영하기 힘듬 (Euclidean Distance를 기반으로 하기 때문)

- 유클리드 거리



코드 예제

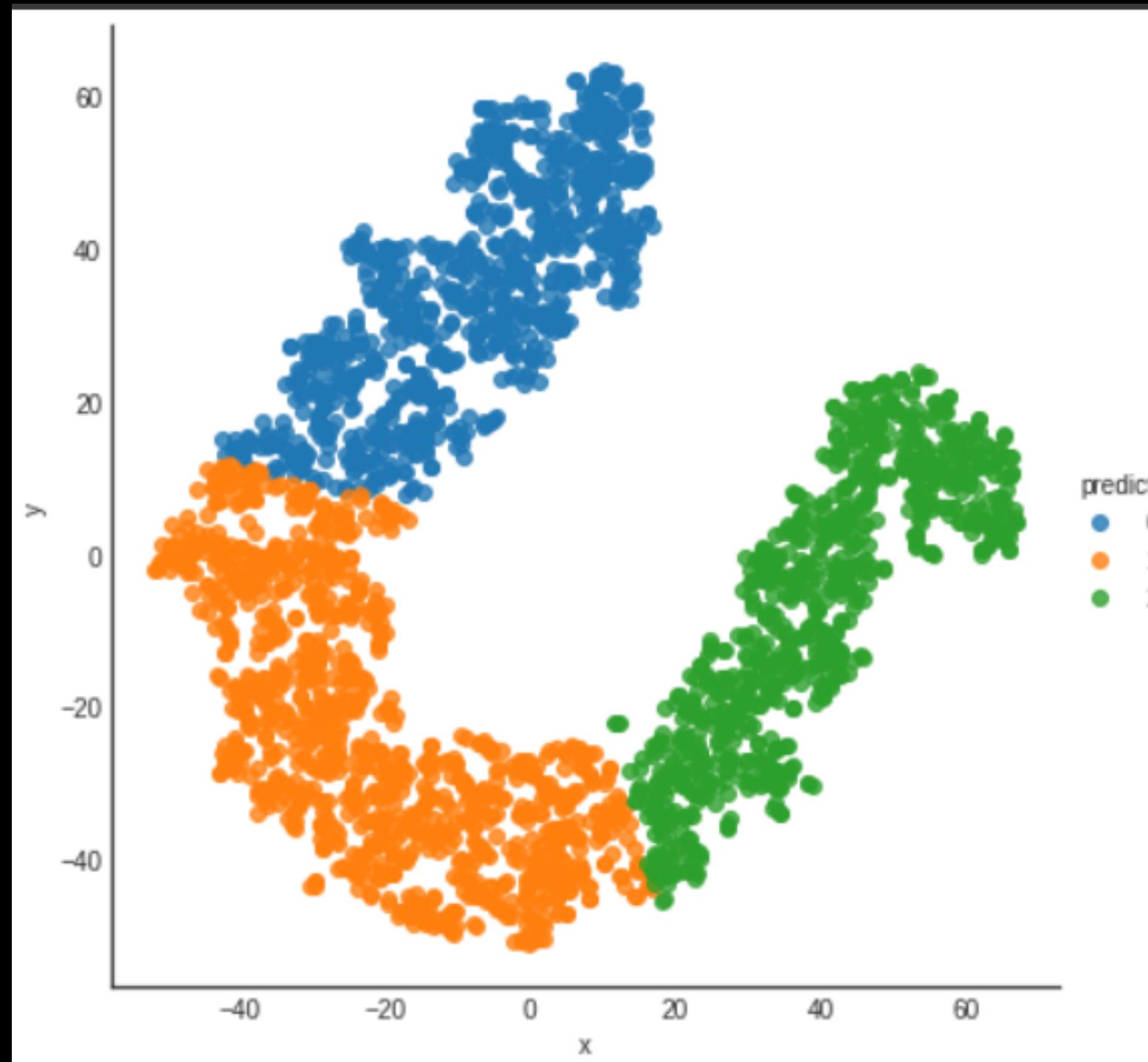
k-means 알고리즘을 사용한 비계층적 군집화

- 클러스터 개수 : 3

```
from sklearn.cluster import KMeans  
  
kmeans = KMeans(n_clusters=3)  
predict = kmeans.fit_predict(df)  
  
results = df  
results['predict'] = predict  
results[:10]
```

	x	y	predict
어릴	23.151413	-17.665709	2
때	58.077744	11.722444	2
보	61.714249	17.495483	2
고	61.205811	16.189301	2
지금	60.398102	9.636668	2
다시	61.829369	16.032358	2
봐도	61.341206	8.923100	2
재밌	66.877228	4.089137	2
어요	65.897354	3.070244	2
ㅋㅋ	65.895126	0.682670	2

코드 예제



총 군집이 처음에 설정하였던 클러스터 개수 3개만큼 형성된 모습

감사합니다
Thank you!

