LogoType™

*Today News_* K_mean기술

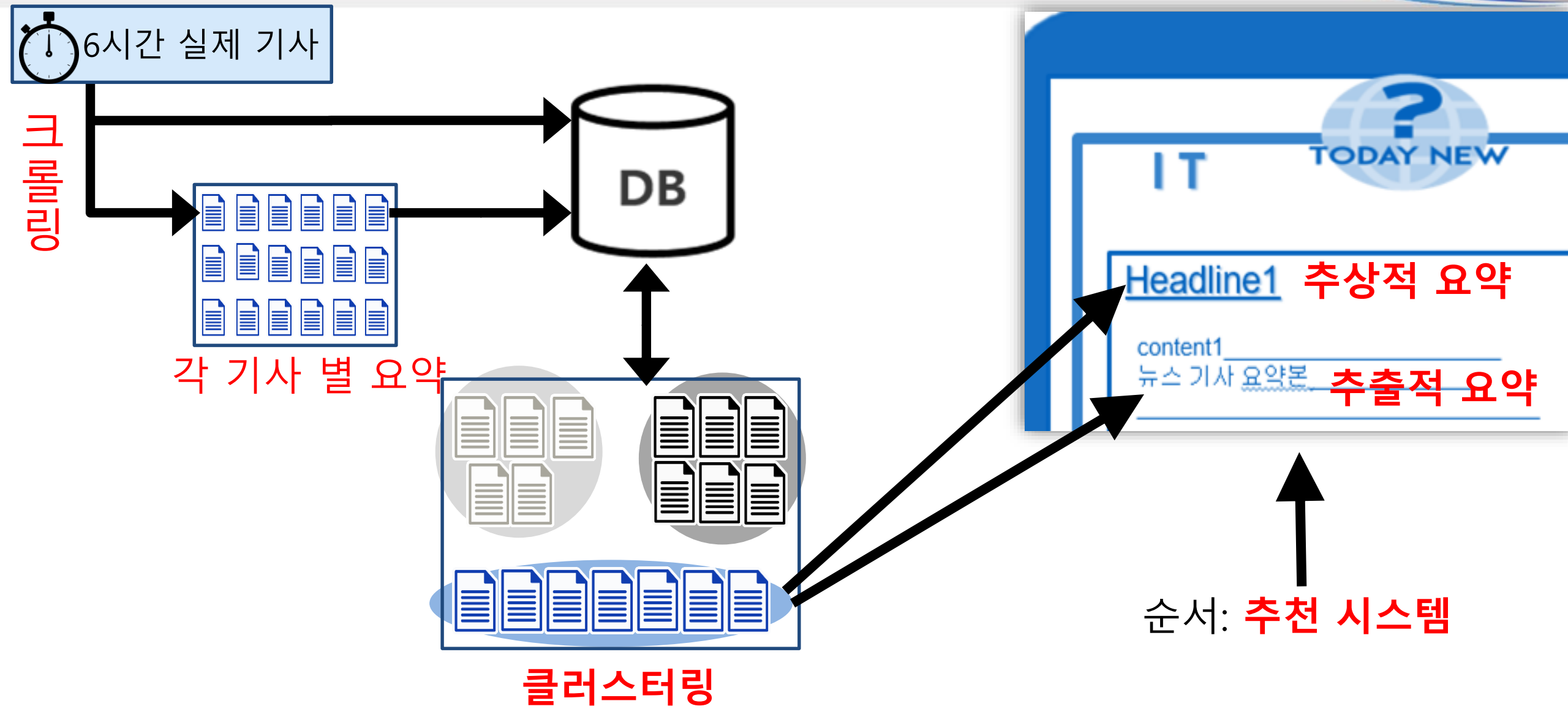
AI summary news application

MUD Team

201601364 박주영(T)　201500000 홍승환
201600599 김아연　201701138 김혜원
201700000 이산가비두사

목차

1. *Today News* 기술 적용 구조



2. K_Mean Clustering

How Using?

- 카테고리 별 뉴스기사 내용이 DATA SET
- K_Mean 클러스터링을 통해
- → 비슷한 뉴스 끼리 묶음
- 큰cluster일수록 사회에서 중요도가 높음

K_Mean Clustering

$$J = \sum_{i=1}^{N} \sum_{k=1}^{K} w^{(i,k)} d(x^{(i)}, \mu^{(k)})^2$$

N개의 데이터, 이를 K개 cluster로 분류 시,
J(cluster내 오차제곱합=관성값) 최소가 되게 w(I,k)와 u(k)를 결정
X(i)는 i번째 데이터, u(k)는 k번째 cluste의 중심
W(I,k)는 x(i)가 k번째 클러스터에 속하면 1, 아니면 0

What?

- Clustering (군집)

: 기계학습에서 비지도학습의 기법

데이터 셋에서 서로 유사한 관찰치들을 그룹으로 묶어 분류하여 몇 가지의 군집(cluster)를 찾아내는 것 비지도 학습

- Clustering종류
  - ***클러스터 중심(Centroid)***
    =평균기반 K_MEANS
  - 빈도수가 많은 중간점(medoid) 기반 K_medoids
  - 밀도 기반
  - 계층적 클러스터링

2. K_Mean Clustering

2.1 K_Mean 원리

< STEPS >

1. 데이터 준비
2. cluster(개수) 결정
3. 클러스터 중심(centroid) 정하기
   - randomly select centroid
   - manually assign centroid
   - kmean++
4. 근접 클러스터 에 데이터 할당
5. 갱신된 클러스터의 중심으로 centroid 옮기기
6. 4-5번 반복 → cluster갱신이 더 이상 일어나지 않을 때까지

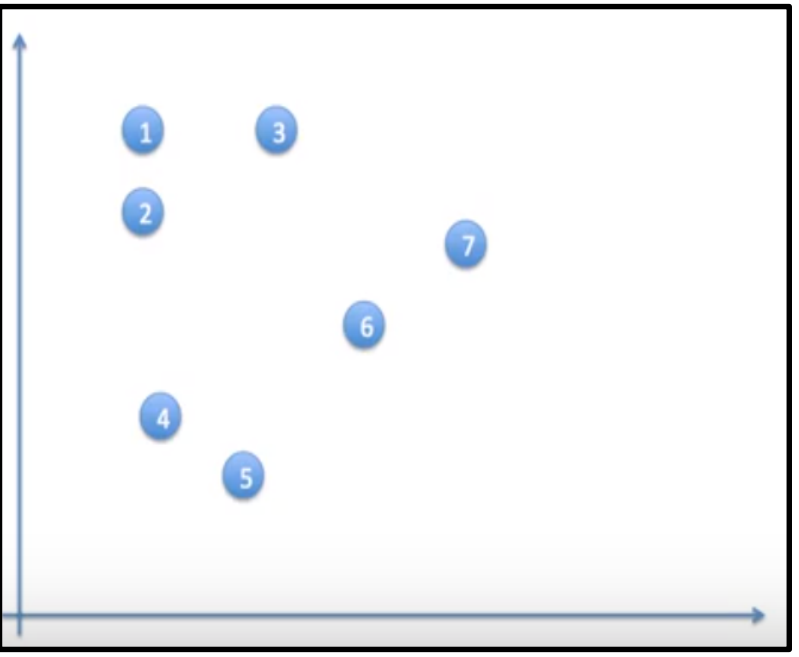
#. 유클리드 거리의 제곱 사용

특성이 m개인 두 데이터 x, y의 유클리드 거리의 제곱

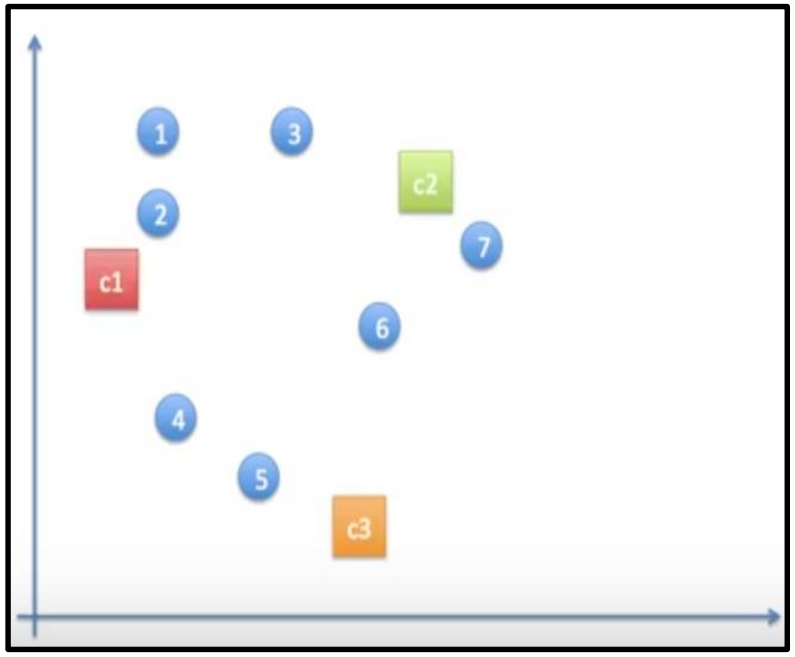
$$d(x,y)^2 = \sum_{j=1}^{m}(x_j - y_j)^2$$

2. K_Mean Clustering

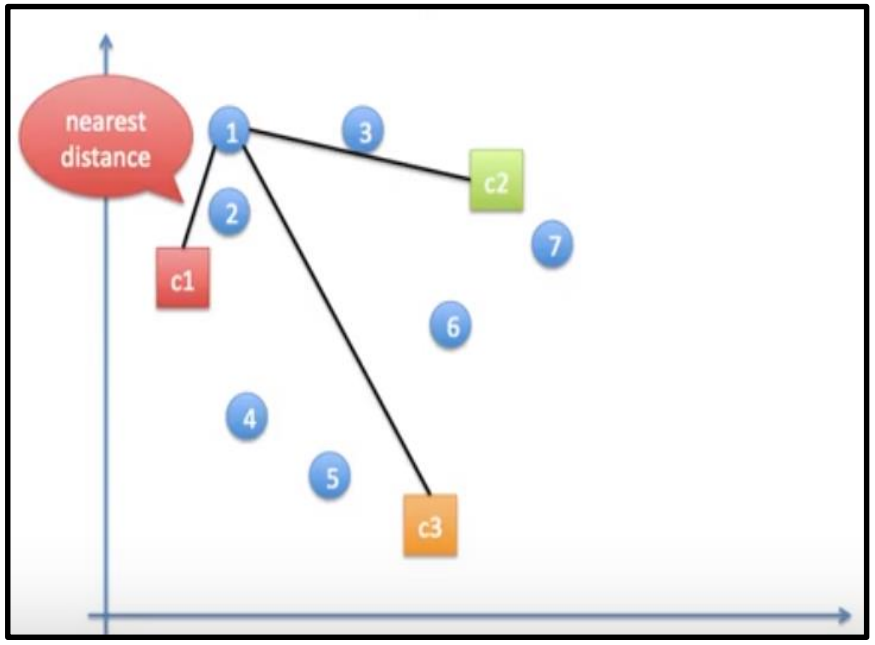
2.1 K_Mean 원리



1. data 분산 그래프



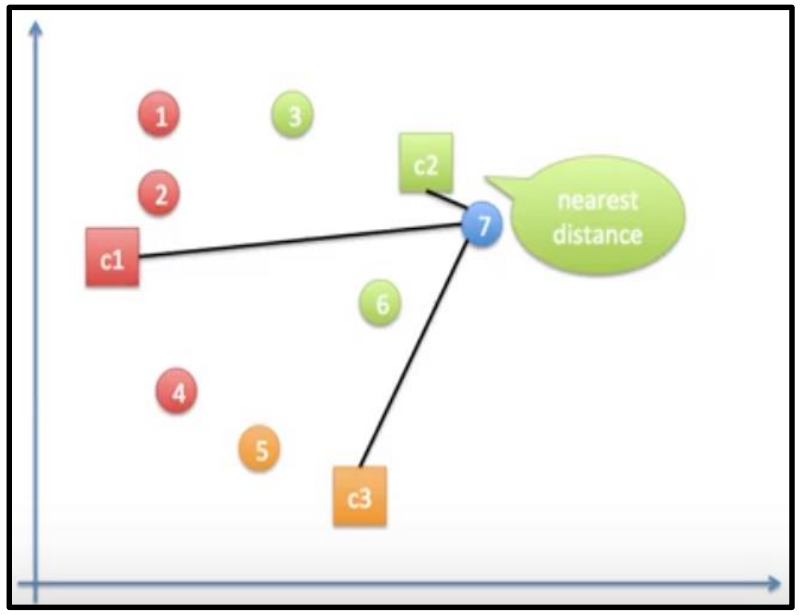
2. cluster 개수 지정
+ random centroid 지정



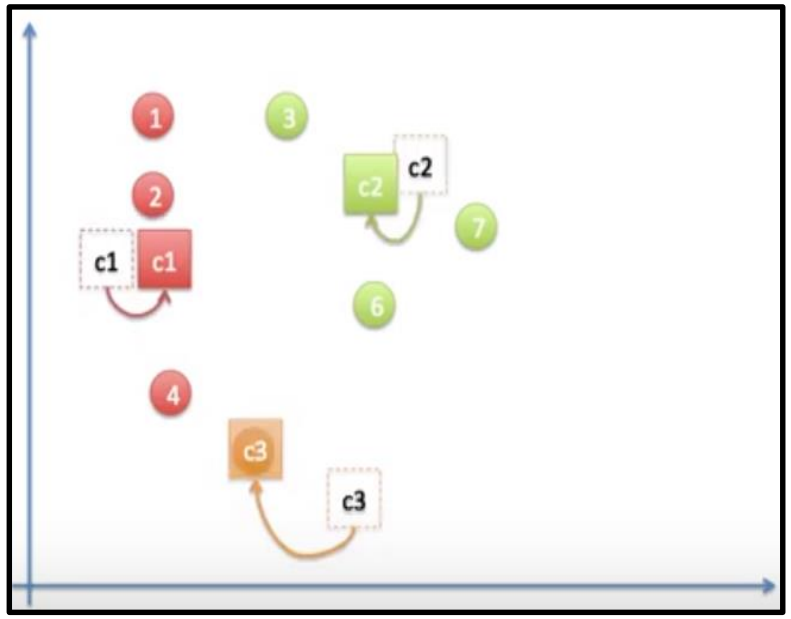
3. (각 data-각 centroid)의 유클리드형 distance측정

2. K_Mean Clustering

2.1 K_Mean 원리



4. 가장 가까운 거리의 centroid로 clustering됨 (색깔로 나타냄)

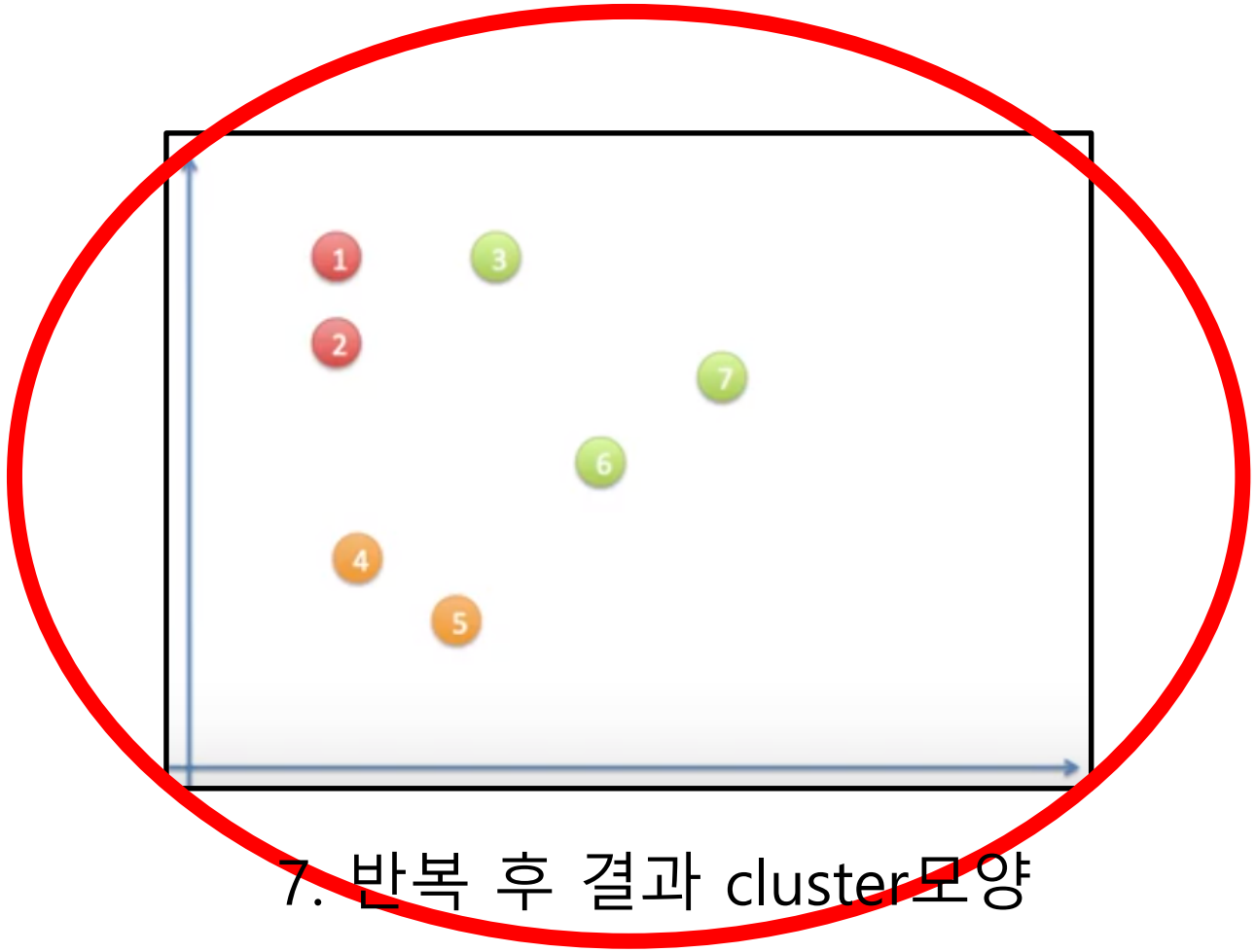


5. 첫 clustering이 끝난 후 각 cluster마다 data의 중심으로 centroid 이동

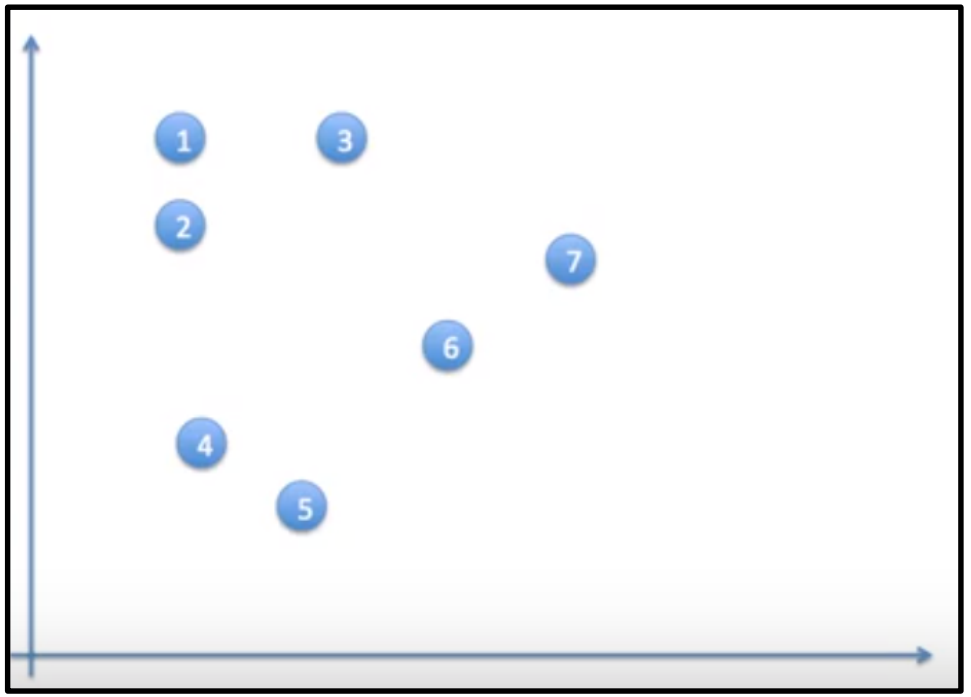
6. 새 centroid를 중심으로 반복(새로운 center이 안나올때까지)

2. K_Mean Clustering

2.1 K_Mean 원리



7. 반복 후 결과 cluster모양



1. 처음 DATA SET 분산 모양

중요한 이유 : 여러 cluster가 나올 때 cluster의 크기 순으로 Headline과 Contents를 추출할 토픽을 선별 할 것
모든 뉴스기사를 보여주는 것이 아니므로 최종 cluster의 중심과 가까운 data를 선별 해 보여줄 것

2. K_Mean Clustering

2.1 K_Mean 원리

STEP_ 3.클러스터링 중심(centroid) 정하기

- **randomly select centroid**
- **manually assign centroid** (수동으로 지정 :

  ex) 특정 국가와 가까운 사람들 군집화 시키기)
- **kmean++**

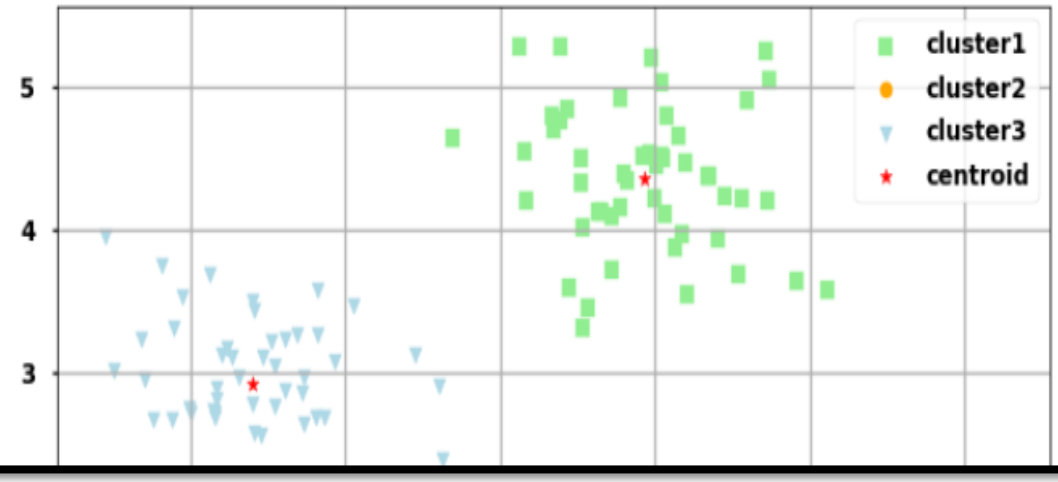
  : 첫 번째 data point 위치에 c1지정

  c2는 c1에서 가장 먼 것

  c3는 c1과 c2와 가장 먼 것

(cN은 N번째 centroid)

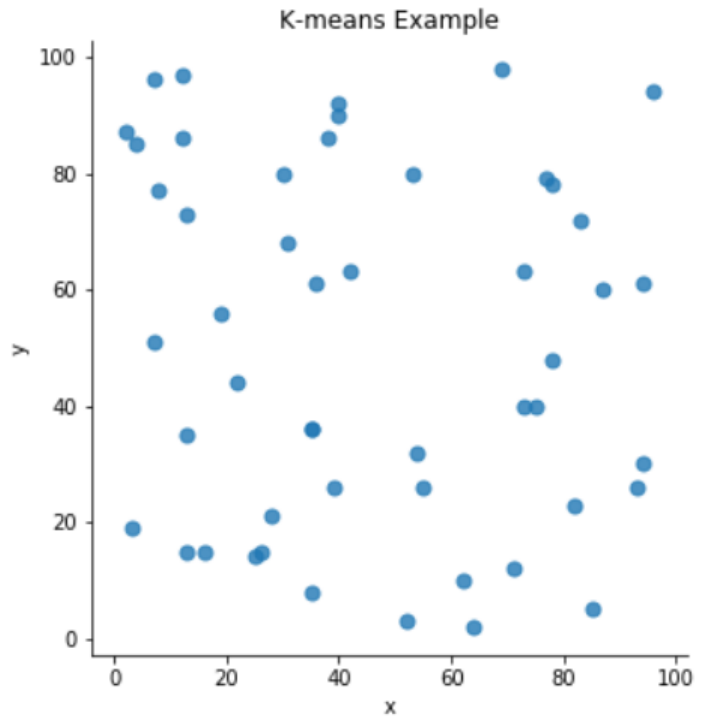
```
from sklearn.cluster import KMeans

init_centroid = 'random'
#init_centroid = 'k-means++'

km = KMeans(n_clusters=3, init=init_centroid, random_state=0)
y_km = km.fit_predict(X)

plt.scatter(X[y_km==0, 0], X[y_km==0, 1], c='lightgreen',marker='s',s=50,label ='cluster1')
plt.scatter(X[y_km==1, 0], X[y_km==1, 1], c='orange',marker='o',s=50,label ='cluster2')
plt.scatter(X[y_km==2, 0], X[y_km==2, 1], c='lightblue',marker='v',s=50,label ='cluster3')
plt.scatter(km.cluster_centers_[:,0], km.cluster_centers_[:,1], c='red',marker='*',s=50,label ='centroid')

plt.legend()
plt.grid(True)
plt.show()
```



2. K_Mean Clustering

2.2 K_Means module Simple Code

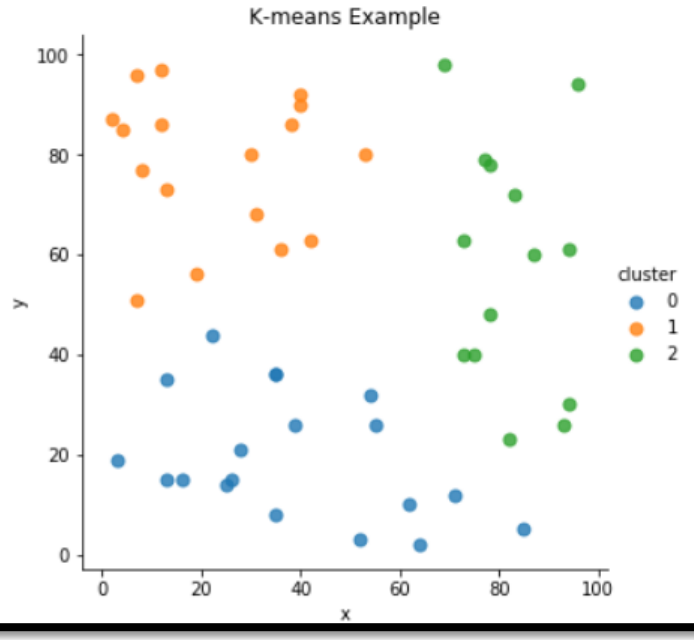
```
#산점도 그래프
sb.lmplot('x','y', data = df, fit_reg = False, scatter_kws = {"s":50})
#fit_reg = False는 회귀 옵션 제거, scatter_kws = {"s":50}는 산점도 그래프 점 크기 설정
plt.title('K-means Example')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```




```
#그 다음 생성도니 데이터프레임을 객체로 변환해주고 클러스트 갯수를 정해 분류한다.
#할때마다 바뀐다.
df_obj = df.values #numpy를 사용하기 위해 객체로 변환
kmeans = KMeans(n_clusters = 3).fit(df_obj) #3개의 클러스트 발생
kmeans.cluster_centers_
kmeans.labels_
```

```
array([1, 0, 2, 1, 1, 1, 2, 0, 1, 2, 1, 0, 2, 1, 0, 0, 1, 1, 0, 0, 1, 0,
       0, 2, 2, 0, 1, 1, 2, 0, 1, 0, 2, 0, 2, 0, 1, 0, 2, 0, 1, 2, 0, 2,
       0, 2, 0, 2, 1, 1])
```

```
#다음은 생성된 라벨을 새로운 컬럼값으로 저장하고 산점도 그래프를 그려준다.
df['cluster'] = kmeans.labels_

sb.lmplot('x','y', data = df, fit_reg = False, scatter_kws = {"s":50}, hue = "cluster")
plt.title('K-means Example')
plt.show()
```

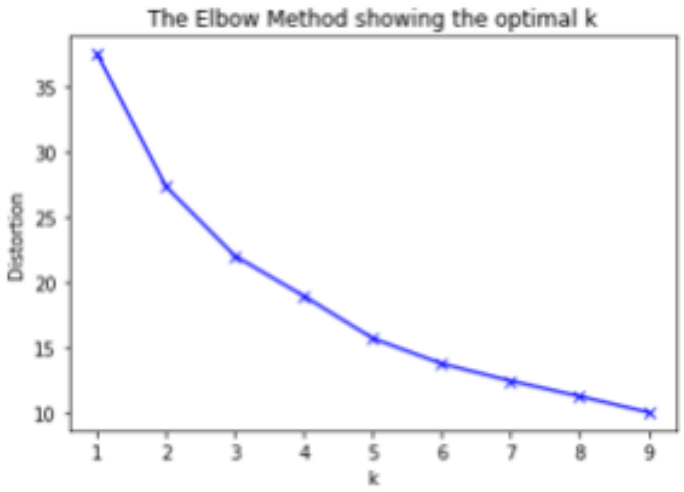


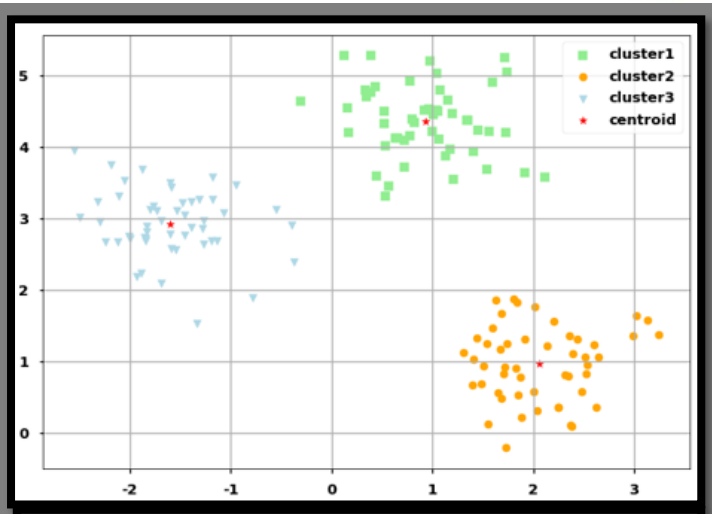
2. K_Mean Clustering

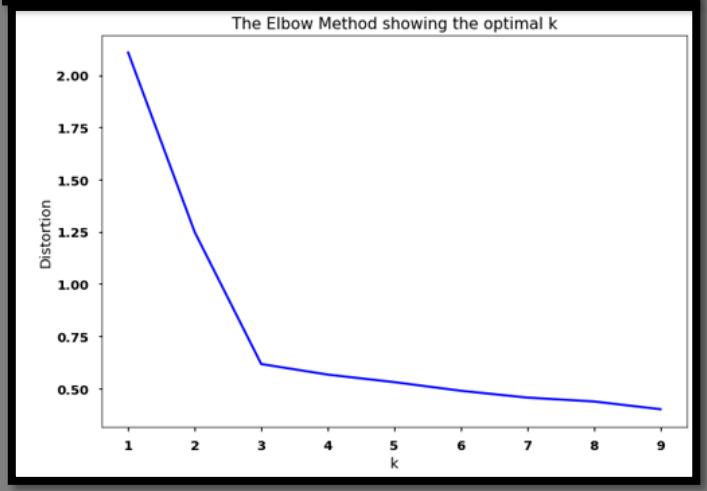
2.2 K_Means module Simple Code

최적의 K값

```
from scipy.spatial.distance import cdist
distortions = []
X=df_obj
K = range(1,10)
for k in K:
    kmeanModel = KMeans(n_clusters=k).fit(X)
    kmeanModel.fit(X)
    distortions.append(sum(np.min(cdist(X, kmeanModel.cluster_centers_, 'euclidean'), axis=1)) / X.shape[0])
# Plot the elbow
plt.plot(K, distortions, 'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('The Elbow Method showing the optimal k')
plt.show()
```







다른 data 분포

2. K_Mean Clustering

2.3 K_Means 실제 적용

전처리 과정 → 군집화 가능

영어: 띄어쓰기만으로 토큰 화가 수행되기 때문에 문제 없음.
한글: 자연어처리를 안하고 CountVectorizer 적용 시,
조사 등의 이유로 제대로 BoW 생성 X
Ex)"봄과"와 "봄이"를 다르게 인식하기 때문에
CountVectorizer 사용 전에 어간 추출을 해야함

```
from konlpy.tag import Mecab
def tokenize_sentense(text):
    mecab=Mecab()
    return mecab.morphs(text)
```

① 직접 전처리 과정

1. 자연어 처리(토큰화)
2. Word2Vec
3. 하나의 글 벡터 화
4. K Means적용

Data [['이', '건', '하나', '의', '글', '입니다',], ['이', '건', '다른', '글', '이죠'], ...]

```
# 2.Word2Vec

from gensim.models import Word2Vec as w2v
model = w2v(tokenized_data, size=100, window=2, min_count=50, iter=20, sg=1)
    # 포스태깅된 컨텐츠를 100차원의 벡터로 바꿔라.
    # 주변 단어(window)는 앞뒤로 두개까지 보되, 코퍼스 내 출현 빈도가 50번 미만인 단어는 분석에서 제외해라.
    # CPU는 쿼드코어를 쓰고 100번 반복 학습해라. 분석방법론은 CBOW와 Skip-Gram 중 후자를 선택해라.
# tokenized_data는 2D list로 저장된 값입니다.
# 위의 함수(tokenized_sentence)의 output을 넣어주면 됩니다.
```

Word2Vec =유사한 단어들을 비슷한 방향과 힘의 벡터를 갖도록 변환하여 사용하는 방법 중 하나

model : data의 각각 하나의 token('이','건','하나','다른' 등)이 word2vec으로 학습된 모델

2. K_Mean Clustering

2.3 K_Means 실제 적용

② Scikit-Learn 의 문서 전처리 기능

Scikit-Learn 의 feature_extraction 서브패키지와 feature_extraction.text 서브 패키지는 다음과 같은 문서 전처리 용 클래스를 제공한다.

- DictVectorizer:
  - 각 단어의 수를 세어 놓은 사전에서 BOW 벡터를 만든다.
- **CountVectorizer:**
  - 문서 집합에서 단어 토큰을 생성하고 각 단어의 수를 세어 BOW 인코딩한 벡터를 만든다.
    1. 문서를 토큰 리스트로 변환한다.
    2. 각 문서에서 토큰의 출현 빈도를 센다.
    3. 각 문서를 BOW 인코딩 벡터로 변환한다.
- TfidfVectorizer:
  - CountVectorizer와 비슷하지만 TF-IDF 방식으로 단어의 가중치를 조정한 BOW 벡터를 만든다
- HashingVectorizer:
  - 해시 함수(hash function)을 사용하여 적은 메모리와 빠른 속도로 BOW 벡터를 만든다.

BOW (Bag of Words)¶

문서를 숫자 벡터로 변환하는 가장 기본적인 방법

전체 문서 $\{d_1,d_2,...,d_n\}\{d1,d2,...,dn\}$ 를 구성하는 고정된 단어장(vocabulary) $\{t_1,t_2,...,t_m\}\{t1,t2,...,tm\}$

개별문서 di.에 단어장에 해당하는 단어들이 포함되어 있는지를 표시하는 방법

$x_{i,j}$=문서 d_i내의 단어 t_j의 출현 빈도

={0 만약 단어 t_j가 문서 d_i 안에 없으면

{1 만약 단어 t_j가 문서 d_i 안에 있으면

2. K_Mean Clustering

2.3 K_Means 실제 적용

실제 한글 뉴스기사 적용

1. Data set 가져오기

data 폴더속 news기사중 교육관련 카테고리 기사를 가져와 data set으로 지정

```
##한글 뉴스 기사 데이터 가져오기

import pandas as pd
df = pd.read_csv('./data/news.csv')

#category ==1은 교육관련기사
df = df[df['category']==1]

df.head(30)
```

	category	content	date	hate	id	image	keyword	like	rate	skip	sub_category	title
0	1	"2028학년도에 수시·정시를 통합하겠다는 방안은 수능을 무력화해 대입제도를 학생부...	2019-02-15 02:42:56.000000	0	4	https://pds.joins.com/news/component/htmlphoto...	대입,교수,수능,시기,학년도	0	0	0	0	"수시·정시 통합해야" vs "수능 무력화 의도"... 2028 대입 두고 논란
		최근 열린 삼보국가대표 선발전에										"공부하겠다는 내게 감독님이 열쇠 주

```
df.loc[df['category']==1,['content']]
```

	content
0	"2028학년도에 수시·정시를 통합하겠다는 방안은 수능을 무력화해 대입제도를 학생부...
1	최근 열린 삼보국가대표 선발전에서 1위를 한 뒤, 신재용 씨 제공\n\n고교 국가대...
2	/첨부용/연세대 수험생 대나무숲\n\n우체국 전산 오류로 등록금을 못 내 연세대 합...

2. Data set 특수문자제거

```
##preprocessing(특수문자제거)

import re
#re.sub() 함수는 문자열에서 매치된 텍스트를 다른 텍스트로 치환할 때 사용한다.
def preprocessing(sentence):
    sentence = re.sub('2028', 'ㅎㅎ', sentence)
    return sentence

df['content_cleaned'] = df['content'].apply(preprocessing)
content = df['content_cleaned'] #content에는 preprocessing이 완료된 교육기사들이 담겼다.
print(content)

df['content_cleaned']
# df.loc[df['category']==1,['content']]
```

```
0       "ㅎㅎ학년도에 수시·정시를 통합하겠다는 방안은 수능을 무력화해 대입제도를 학생부종합...
1       최근 열린 삼보국가대표 선발전에서 1위를 한 뒤, 신재용 씨 제공ㅠㅠㅠ고교 국가대...
2       /첨부용/연세대 수험생 대나무숲ㅠㅠㅠ우체국 전산 오류로 등록금을 못 내 연세대 합...
3       교육당국, 갑자기 통과 기준 점수 대폭 올려… 폐지 논란 재점화ㅠㅠㅠ자사고·외, 교...
4       [베리타스알파-권수진 기자] 14일 오후9시 마감한 상위9에대학(고려대 동국대 서강
```

2. K_Mean Clustering

2.3 K_Means 실제 적용

실제 한글 뉴스기사 적용

3. Data set 전처리과정

_CounterVectorizer()이용

4. 토큰화와 정규화가 마친 X를

K_means알고리즘에 적용

```
content_cleaned                                                    유치원 관련된 내용
공립유치원 육아지원센터 활용키로 개학 연기 없는 것으로 파악 이덕선 한국유치...      7
교육당국 엄정 대응 방침 긴급 돌봄 서비스 제공해 공백 최소화 검찰 고발 땐 ...7
개학 연기 사립유치원 명단 오늘 정오 시도교육청 홈피 공개 머니투데이 세종 ...7
일부 사립유치원의 개학연기와 관련해 교육 당국이 개학 연기 유치원 명단을 공개하기로...      7
돌봄 신청 교육지원청 연락처 서울시교육청은 관내에서 개학 연기를 결정한 유치원...      7
충북교육청 전경 한국유치원총연합회의 개학 연기 방침에 동참하는 충북 사립유치원은 ... 7
개학 연기 유치원 수 맞습니꺼 우리 동네는 더 많다 카든데 머니투데이 세종 ... 7
```

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.preprocessing import normalize
from sklearn.cluster import KMeans

# 군집화 할 그룹의 갯수 정의
n_clusters = 100

# CountVectorizer로 토큰화
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(content)

# l2 정규화
X = normalize(X)

# k-means 알고리즘 적용
kmeans = KMeans(n_clusters=10).fit(X)

# trained labels and cluster centers
labels = kmeans.labels_
centers = kmeans.cluster_centers_

# labels에 merge
df['labels'] = labels
#간단히 코드 몇 줄 만으로 뉴스기사에대한 clustering 완료
```

```
df.loc[df['labels']==7, ['content_cleaned', 'labels']]
```

	content_cleaned	labels
7925	공립유치원 육아지원센터 활용키로 개학 연기 없는 것으로 파악 이덕선 한국유치...	7
8043		

2. K_Mean Clustering

2.3 K_Means 실제 적용

CounterVectorzier

1. 문서를 토큰 리스트로 변환한다.

2. 각 문서에서 토큰의 출현 빈도를 센다.

3. 각 문서를 BOW 인코딩 벡터로 변환한다.

- stop_words : 문자열 {'english'}, 리스트 또는 None (디폴트)
  - stop words 목록.'english'이면 영어용 스탑 워드 사용.
- analyzer : 문자열 {'word', 'char', 'char_wb'} 또는 함수
  - 단어 n-그램, 문자 n-그램, 단어 내의 문자 n-그램
- token_pattern : string
  - 토큰 정의용 정규 표현식
- tokenizer : 함수 또는 None (디폴트)
  - 토큰 생성 함수 .
- ngram_range : (min_n, max_n) 튜플
  - n-그램 범위
- max_df : 정수 또는 [0.0, 1.0] 사이의 실수. 디폴트 1
  - 단어장에 포함되기 위한 최대 빈도
- min_df : 정수 또는 [0.0, 1.0] 사이의 실수. 디폴트 1
  - 단어장에 포함되기 위한 최소 빈도.

Thank You

감사합니다.