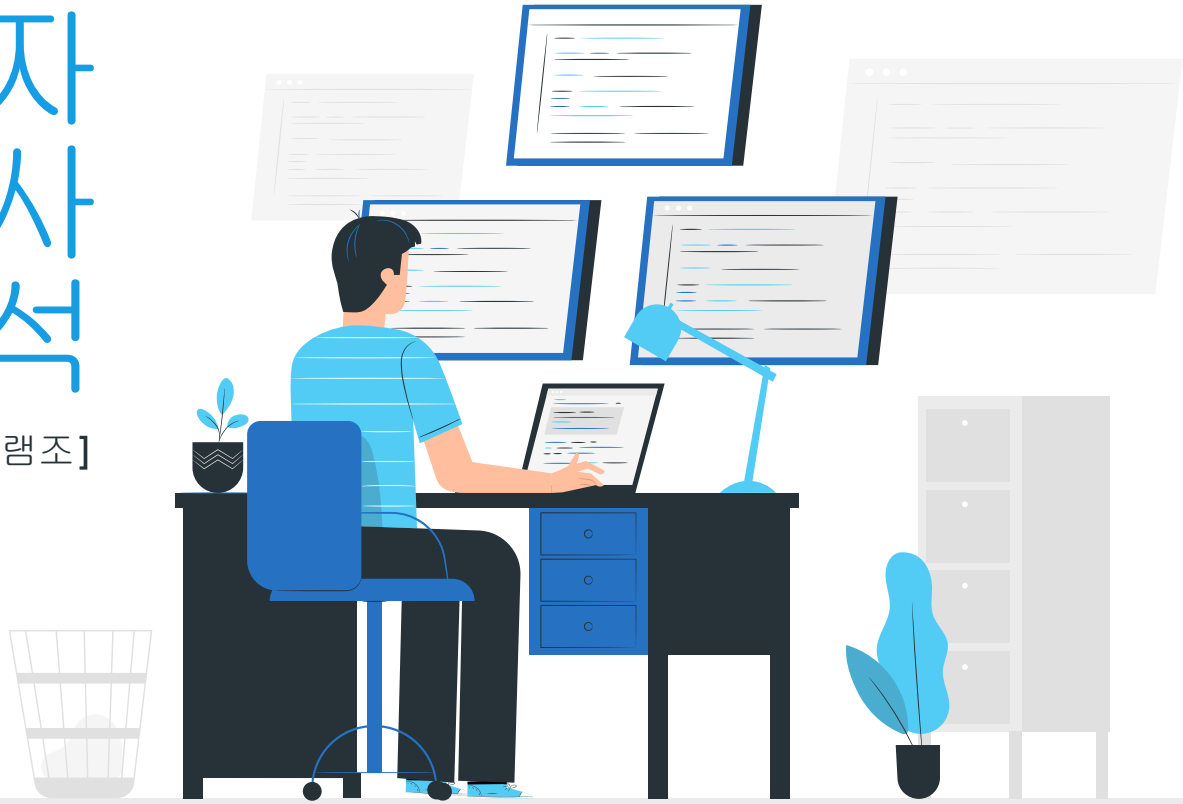
사용자 설문조사 결과 분석

[사그램조]

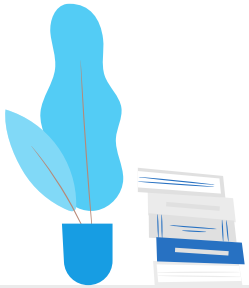
201402433 조승현 201704144 김수민
201704145 김주희 201402392 이상화



주제

프라이버시 보호 팁러닝 서비스 개발

소비 패턴을 분석해 적절한 금융 상품을 추천하는 어플리케이션 개발



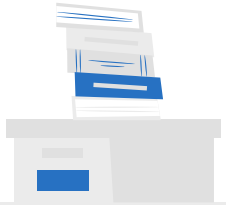
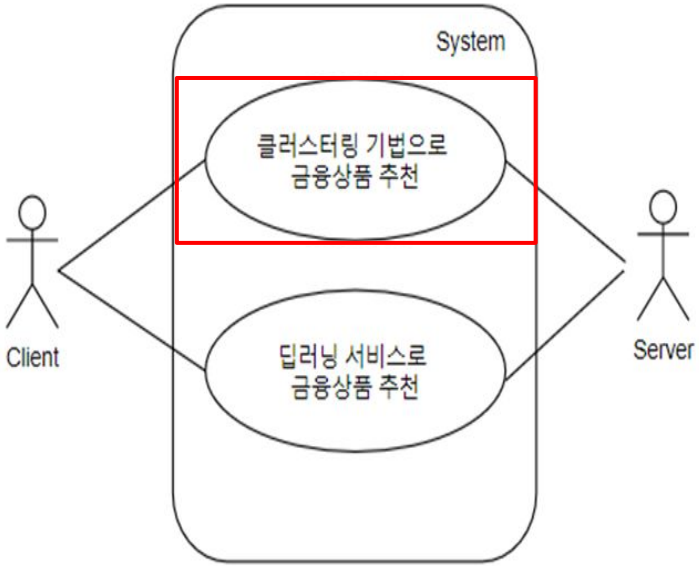


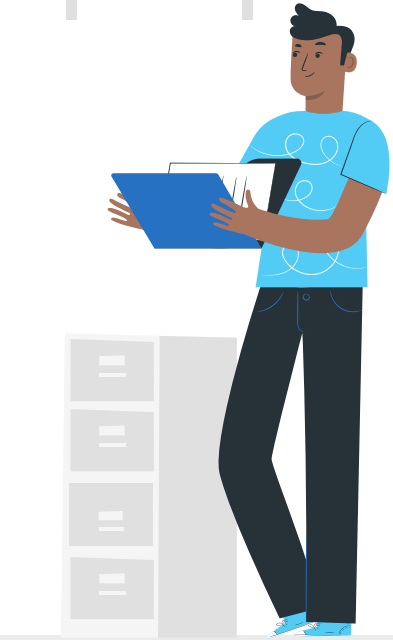
TABLE OF CONTENTS

01. 프로토타입 설명

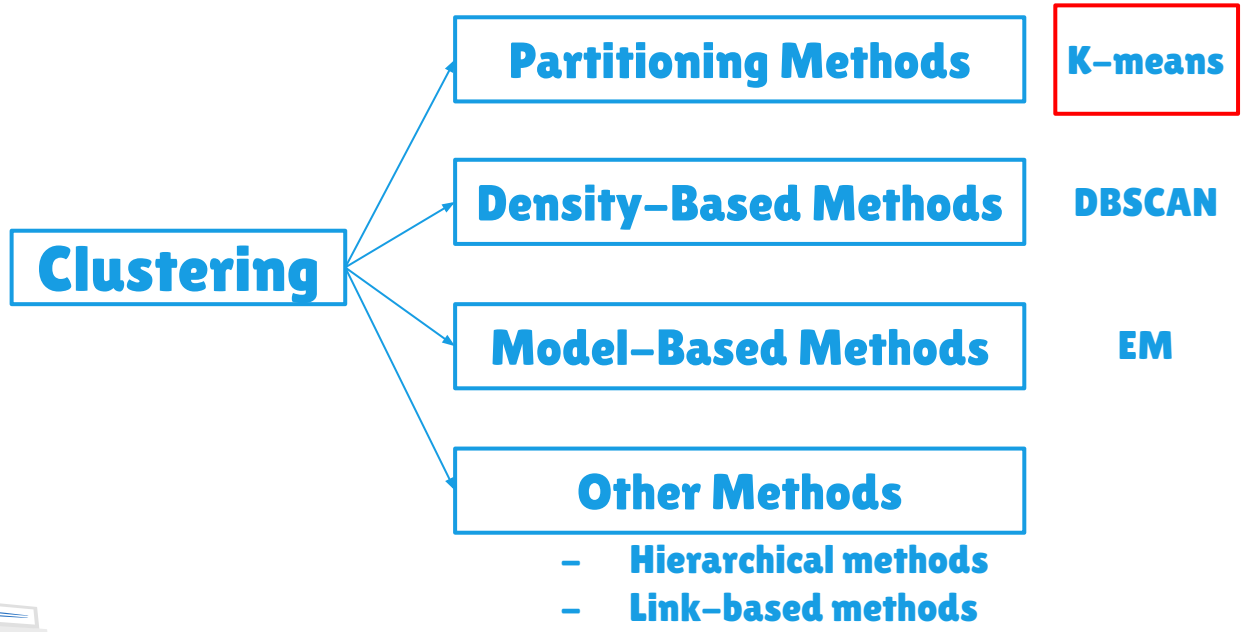
[사그램조] 201402433 조승현

프로토타입 설명





프로토타입 설명


Clustering
Partitioning Methods
K-means
Density-Based Methods
DBSCAN
Model-Based Methods
EM
Other Methods
- Hierarchical methods
- Link-based methods

프로토타입 설명

K-Means

fit
DP가 적용된 K-means를
계산해주는 함수

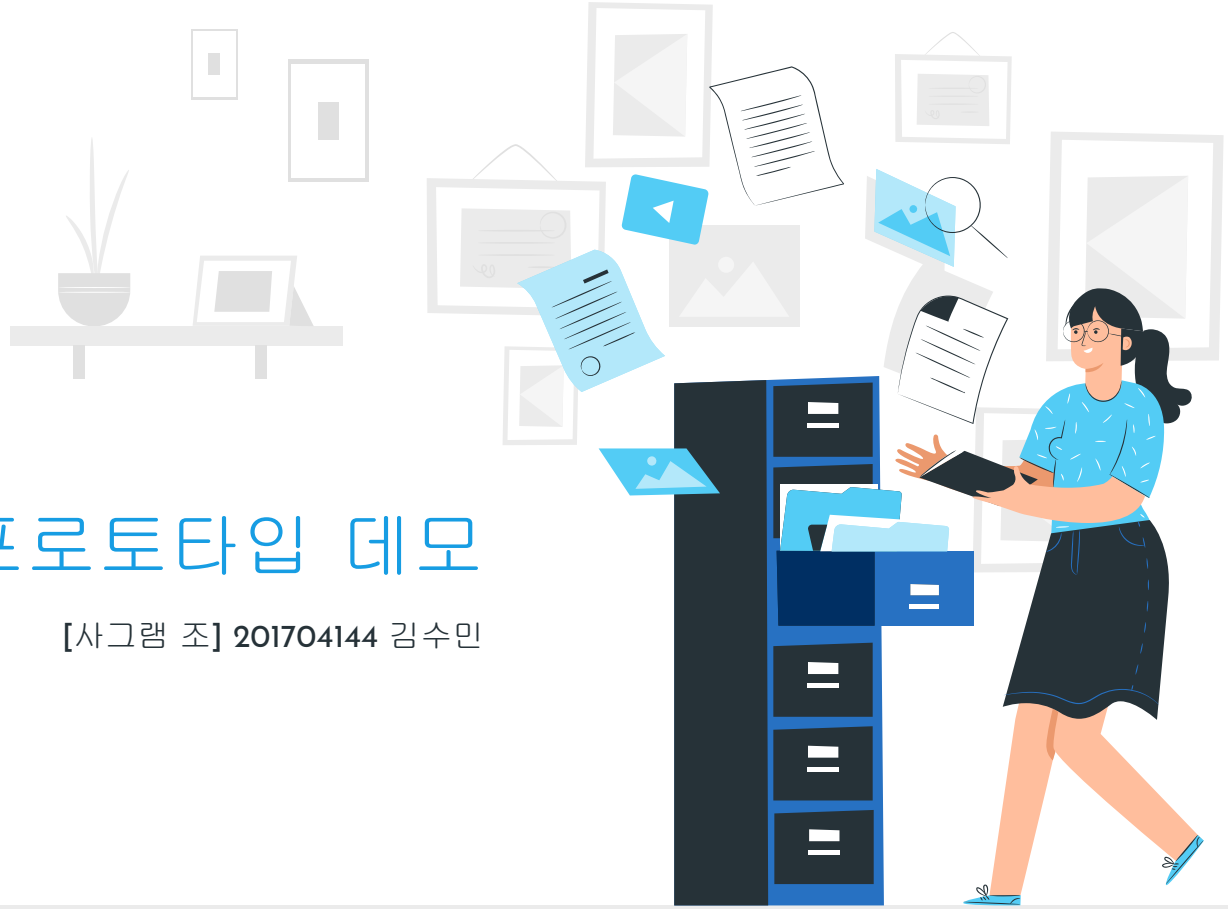
_init_centers
K-means의 center을 초기화
시켜주는 함수

_distances_labels
Current label의 거리를
계산해주는 함수

_update_centers
K-means의 center을 update
시켜주는 함수

_split_epsilon
Sum과 count perturbation
사이 epsilon 분할시키는 함수

_calc_iters
K-means의 최대 반복
횟수를 계산해주는 함수



02. 프로토타입 데모

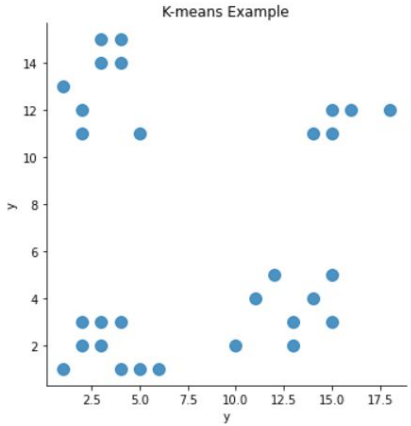
[사그램 조] 201704144 김수민

프로토타입 데모

차등프라이버시를 적용하지 않은 **K MEANS vs** 차등프라이버시를 적용한 **K MEANS**

```
sb.lmplot('x', 'y', data=df, fit_reg=False, scatter_kws={"s": 100})

plt.title('K-means Example')
plt.xlabel('x')
plt.xlabel('y')
```

```
Text(0.5, 6.79999999999999, 'y')
```



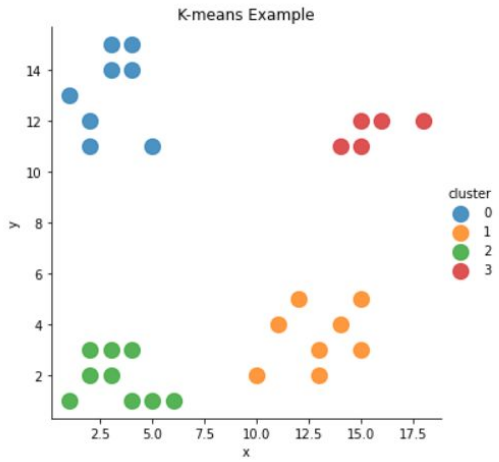
프로토타입 데모

차등프라이버시를 적용하지 않은 **K MEANS vs** 차등프라이버시를 적용한 **K MEANS**

```
In [40]: points = df.values

         REAL_kmeans = REAL_Kmeans(n_clusters=4).fit(points)

In [41]: REAL_kmeans.cluster_centers_

Out[41]: array([[ 3.        , 13.125     ,  3.25      ],
                [12.875     ,  3.5       ,  2.25      ],
                [ 3.33333333,  1.88888889,  3.66666667],
                [15.6       , 11.6       ,  3.        ]])

In [42]: REAL_kmeans.labels_

Out[42]: array([2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0,
                0, 0, 0, 3, 3, 3, 3, 3])

In [43]: df['cluster'] = REAL_kmeans.labels_
         df.head(30)

Out[43]:
```

```
sb.lmplot('x', 'y', data=df, fit_reg=False, scatter_kws={"s": 150}, hue = "cluster")

plt.title('K-means Example')
```

```
Text(0.5, 1.0, 'K-means Example')
```



프로토타입 데모

차등프라이버시를 적용하지 않은 K MEANS vs 차등프라이버시를 적용한 K MEANS

```
DP_kmeans = DP_KMeans(n_clusters=4).fit(points)
```

```
C:₩Users₩LG₩Anaconda3₩envs₩pysyft₩lib₩site-packages₩diffprivlib₩models₩k_
nd will be calculated on the data provided.  This will result in addition
al privacy leakage, specify `bounds` for each dimension.
  "privacy leakage, specify `bounds` for each dimension.", PrivacyLeakWar
```

```
DP_kmeans.cluster_centers_
```

```
array([[ 8.26795268, 12.06546938,  4.61758651],
       [16.6937239 ,  1.42871205,  1.04951336],
       [ 2.52553564,  2.43273052,  5.01460676],
       [11.27148078,  9.29469526,  2.55739668]])
```

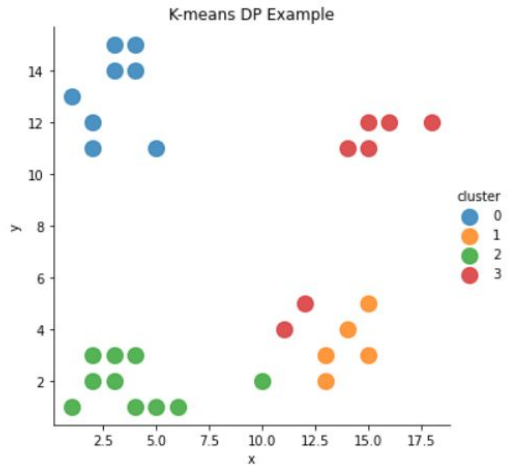
```
DP_kmeans.labels_
```

```
array([2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 3, 3, 3, 3, 3], dtype=int64)
```

```
df['cluster'] = DP_kmeans.labels_
df.head(30)
```

```
sb.lmplot('x', 'y', data=df, fit_reg=False, scatter_kws={"s": 150}, hue = "cluster")

plt.title('K-means DP Example')
```

```
Text(0.5, 1.0, 'K-means DP Example')
```



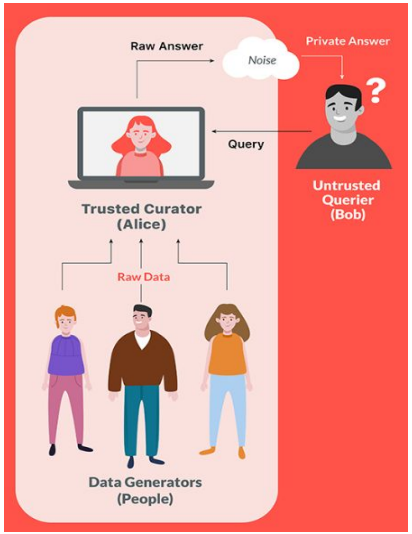
프로토타입 데모

차등프라이버시를적용하지 않은 K MEANS vs 차등프라이버시를적용한 K MEANS

```
def _update_centers(self, X, centers, labels, dims, total_iters):

    epsilon_0, epsilon_i = self._split_epsilon(dims, total_iters)
    geometric_mech = GeometricFolded().set_sensitivity(1).set_bounds(0.5, float("inf")).set_epsilon(epsilon_0)
    laplace_mech = LaplaceBoundedDomain().set_epsilon(epsilon_i)

    for cluster in range(self.n_clusters):
        if cluster not in labels:
            continue

        cluster_count = sum(labels == cluster)
        noisy_count = geometric_mech.randomise(cluster_count)

        cluster_sum = np.sum(X[labels == cluster], axis=0)
        noisy_sum = np.zeros_like(cluster_sum)

        for i in range(dims):
            laplace_mech.set_sensitivity(self.bounds[i][1] - self.bounds[i][0]) \
                .set_bounds(noisy_count * self.bounds[i][0], noisy_count * self.bounds[i][1])
            noisy_sum[i] = laplace_mech.randomise(cluster_sum[i])

        centers[cluster, :] = noisy_sum / noisy_count

    return centers
```

Laplacian Noise

Delta : 항상 0



<Global differential privacy>

프로토타입 데모

차등프라이버시를 적용하지 않은 **K MEANS vs** 차등프라이버시를 적용한 **K MEANS**

<<성능비교>>

```
REAL_VALUE=[1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,3,3,3,3,3,3,3,3,4,4,4,4,4]
```

```
#클러스터링 평가 지표로서 Rand Index
print("original k-means Rand Index value: ", rand_index(REAL_VALUE,REAL_kmeans.labels_))
print("DP k-means Rand Index value: ", rand_index(REAL_VALUE, DP_kmeans.labels_))
```

```
original k-means Rand Index value:  1.0
DP k-means Rand Index value:  0.9172413793103448
```

```
#Ajusted RAND Index
# ARI : 1(최적일 때), 0(무작위로 분류될 때)
from sklearn.metrics import adjusted_rand_score
print("original k-means Rand Index value: ", adjusted_rand_score(REAL_VALUE,REAL_kmeans.labels_))
print("DP k-means Rand Index value: ", adjusted_rand_score(REAL_VALUE, DP_kmeans.labels_))
```

```
original k-means Rand Index value:  1.0
DP k-means Rand Index value:  0.7710325467146241
```

03. 설문조사 결과 분석

[사그램조] 201704145 김주희

설문 개요

본 설문조사는 사그램조가 구현한 프로토타입에 대한 사용자들의 의견을 분석해보기 위하여 기획되었습니다.

이를 통해 문제점이나 개선해야 할 점을 참고하여 해결책을 찾아 보완해보고자 합니다.

- 응답자 수 : **15** 명
- 기간 : **2020.05.28 – 2020.05.29 (2**일간**)**
- 방법 : **Google** 설문지

설문 개요

[사그램 조] 졸업 프로젝트 - 사용자 설문 조사

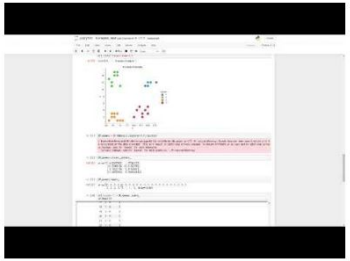
종합설계1 - 사그램조의 졸업 프로젝트 프로토 타입 사용자 설문 조사 입니다.

[주제] 프라이버시 보호 딥러닝 서비스 개발

- 소주제 : 사용자의 소비 내역을 분석해 적절한 금융 상품을 추천해 주는 어플리케이션

프로토 타입 : 차등 프라이버시(Differential Privacy)를 적용한 K-Means 클러스터링 기반 추천 시스템의 기초 모델

차등 프라이버시 보호를 적용한 K-Means 클러스터링 모델 구현 영상입니다.
영상을 시청하시고 아래의 설문에 응답해 주세요.



차등 프라이버시(Differential Privacy)를 적용한 K-Means 클러스터링 기반 추천 시스템의 기초 모델

- 차등 프라이버시(Differential Privacy) : 데이터에 노이즈를 추가하는 기술로, 사용자의 개인정보와 같은 민감한 데이터를 보호할 수 있는 프로세스이다.

- k-means model : 주어진 데이터를 k개의 클러스터로 묶는 알고리즘으로, 각 클러스터와 거리 차이의 분산을 최소화하는 방식으로 동작한다. 이를 통해 사용자의 데이터에 대해 동질 유형을 분류하고 분류 기반 추천을 수행할 수 있다.

- step 1 : 클러스터의 개수 k 값을 선택
- step 2 : data가 분포 된 공간 상에 클러스터 중심으로 가정할 임의의 center of cluster 선택
- step 3 :임의로 선택한 center와 개별 데이터 사이의 거리를 계산하여 가장 가깝게 있는 center를 기준으로 소속된 클러스터로 할당
- step 4 : 클러스터에 속하게 된 데이터들의 평균값을 새로운 클러스터 center로 지정
- step 5 : 3 ~ 4단계를 center가 변화하지 않을 때까지 반복

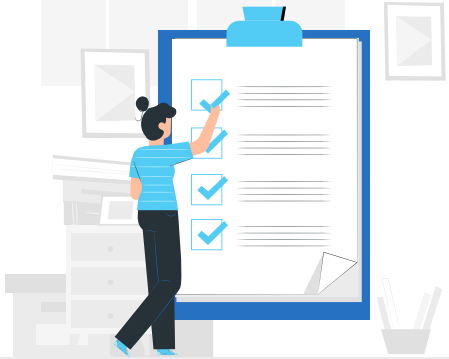
(클러스터링 평가 지표 ARI 값이 1에 가까울 수록 좋은 모델임)

설문 문항

01 해당 모델이 사용자의 데이터를 분석하여 적절한 상품을 추천해 주는 어플리케이션을 개발하는데 얼마나 도움이 된다고 생각하시나요?

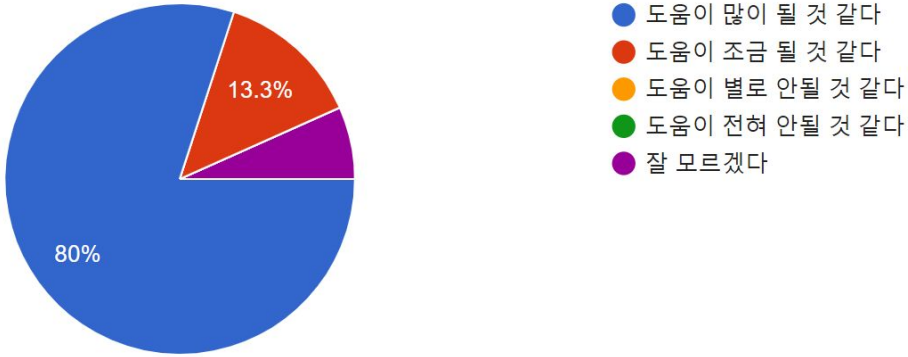
02 해당 모델의 문제점(부족한 점)이 있다면 무엇인가요?

03 해당 모델로 구현된 서비스를 사용할 의향이 있으십니까?



설문 결과

해당 모델이 사용자의 데이터를 분석하여 적절한 상품을 추천해 주는 어플리케이션을 개발하는데 얼마나 도움이 된다고 생각하시나요**?**



15명의 응답자 중, 도움이 많이 될 것 같다 12명 / 도움이 조금 될 것 같다 2명 / 잘 모르겠다 1명

도움이 될 것 같다는 응답 (93.3%) 多 !!

설문 결과

해당 모델의 문제점(부족한 점)이 있다면 무엇인가요?

- 데이터 분류시에 특징을 조금 더 디테일하게 분류할 수 있으면 좋겠다
- 오류가 자주 발생할 것 같다
- 3~4단계를 반복한다는 점이 비효율적이라고 생각한다. 데이터의 양이 많아지면 더욱 번거로운 일이 될 것으로 예상된다. 한 두번 정도의 실행으로 오차값을 줄일 수 있는 방법을 구현하는 것도 좋을 것 같다.
- 데이터의 차원 자체가 너무 커서 쉽지 않을 것 같다.
- 기존의 학습데이터 셋을 딥러닝 한 경우 테스트 데이터를 넣어서 딥러닝 하였을 때 오차율이 크게 나올 것 같다.

이 외의 응답은 "문제점(부족한 점)이 없다"

설문 결과

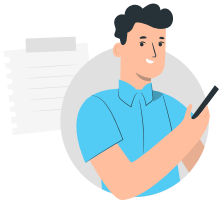
반복수행 하는 것이 비효율적. 데이터의 양이 많아지면 번거로워질 것 같은데 한 두번의 실행으로 줄일 수 있는 방법을 구현하는 것도 좋을 듯

네 맞습니다. K-Means 방법은 중심위치와 모든 데이터 사이의 거리를 계산해야 하기 때문에 데이터의 양이 많아지면 계산량도 늘어나게 됩니다.
이처럼 데이터의 수가 많은 경우에는 데이터를 미니배치 크기만큼 무작위로 분리하여 K-Means를 수행하는 **‘미니배치 K-Means’** 로 계산량을 줄일 수 있습니다. 계산량이 줄기 때문에 속도도 훨씬 빠른 장점을 갖습니다.

설문 결과

데이터의 차원 자체가 너무 커서 쉽지 않을 것 같다

사용할 데이터와 feature가 많으면 어떤 feature를 사용해야 할지 상당히 난감해지는데요. 이러한 고차원 데이터에 아무런 처리를 하지 않고 군집화를 시도하면 그다지 좋은 성능을 낼 수 없습니다.
따라서 차원 축소를 수행해 주어야 합니다. 차원을 축소해주는 기법인 PCA나 SVD를 이용할 수 있습니다.



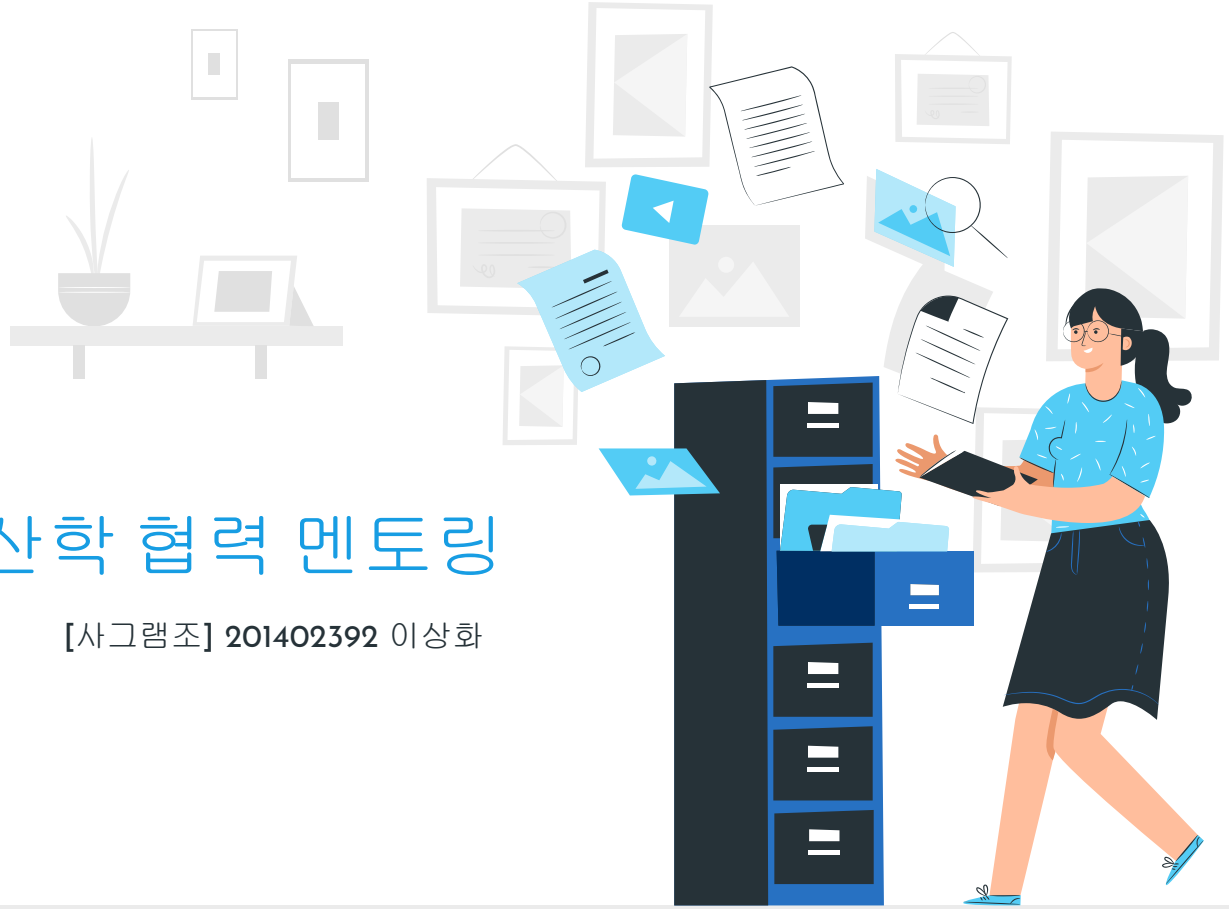
주성분 분석(PCA) : 여러 변수간에 존재하는 상관관계를 이용하여 이를 대표하는 주성분을 추출하여 줌

특이값 분해(SVD) : 임의의 고유값 분해를 직사각형 행렬에 대해 일반화한 방법

설문 결과

해당 모델로 구현된 서비스를 사용할 의향이 있으십니까?

- 학습된 데이터로 최적의 상품을 추천해준다는 점에서 사용할 의향이 있다.
- 개인정보는 보호하고, 시간을 단축하여 필요한 상품을 가입할 수 있을 것 같아 사용할 의향이 있다.
- 요즘 사람들이 평균 10분에 한 번 핸드폰을 본다는 결과가 있듯 핸드폰 사용시간이 늘어나면서 편리하게 이용할 수 있는 모바일 금융서비스의 이용률도 증가하였다. 이 서비스 역시 편리하게 사용할 수 있을 것 같아 사용할 의향이 있다.
- 관련 정보나 지식이 부족한 분들에게 좋을 것 같고 아무래도 금융 서비스이다보니 민감한 정보가 유출되는 걱정을 덜 수 있어서 좋은 모델이라고 생각한다.
- 금융 상품을 선택할 때 내 소비패턴을 판단하여 결정하기까지 쉽지 않았는데 딥러닝을 통해 분석해준다면 보다 빠르게 적절한 판단을 할 수 있을 것 같다.
- 데이터를 분류하여 추천해주는 과정에서 정보가 유출될 수도 있고 악용될 수도 있을 것이라는 불안감을 가질수도 있지만, 데이터를 암호화하여 추천된 상품이라 하면 안심되고 해당 서비스에 대한 신뢰감도 생길 것 같다.
- 정확도가 좀 더 높아지고 개인정보 보호만 확실해 진다면 사용할 것 같다.



04. 산학 협력 멘토링

[사그램조] 201402392 이상화

산학 협력 멘토링

- 화상 회의로 멘토링 진행
- 지도 교수, 멘토, 멘티 전원 참가

수행 결과

- 커스텀으로 만든 임의의 데이터 말고 실제 데이터 셋을 다뤄보는 것을 추천 -> 클러스터링 할 수 있는 금융 관련 오픈 데이터 셋을 찾아야 한다.

- 데이터 셋을 구할 때 금융 빅데이터 플랫폼 사이트에서 데이터 셋을 유료로 구매하는 방법이 있으며 데이터를 구하기 힘든 경우 데이터 셋을 지인이나 다른 사람들로부터 모으는 방법도 있다.

- 만들어진 서비스에 대한 평가를 어떻게 할 것인가를 고민해봐야 한다.

수행 결과

- 온디바이스는 주로 저사양의 엣지 디바이스(라즈베리파이, 아두이노 등등)를 의미하는데 엣지 디바이스상에서 딥러닝 모델을 운영하는 것이 쉽지 않다. 한번에 메모리에 올라갈 수 있도록 모델을 경량화(압축)하는 작업이 필요하며 성능 역시 중요하다.
  ->*성능이 덜 떨어지는 선에서 경량화를 하는 것이 주요 이슈*

- 차등 프라이버시와 연합 학습, 온디바이스를 전부 구현하는 것이 벅찬 목표일 수 있으므로 추후에 이 주제들을 간소화할 필요가 있다.

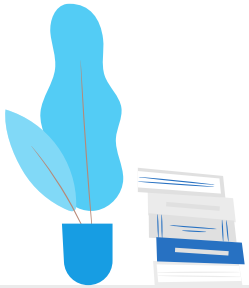
링크

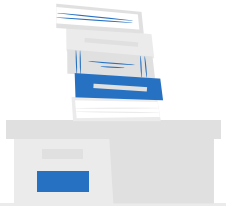
Youtube

https://www.youtube.com/watch?v=NBoVc_i7YfA

GitHub

https://github.com/pmcsh04/designsprint_4gram/tree/master/GP_Final2





감사합니다

종합설계 02분반 [사그램 조]