

9th Meet

□ 오늘 공부할 것은

06

차원 축소

01. 차원 축소(Dimension Reduction) 개요	399
02. PCA(Principal Component Analysis)	401
PCA 개요	401
03. LDA(Linear Discriminant Analysis)	415
LDA 개요	415
04. SVD(Singular Value Decomposition)	418
SVD 개요	418
사이킷런 TruncatedSVD 클래스를 이용한 변환	424
05. NMF(Non-Negative Matrix Factorization)	427
NMF 개요	427
06. 정리	429

Now, It's your turn !!!

40min ~ 50min

Chapter 06

차원 축소

(Dimension Reduction)

[06-01]
차원 축소
(Dimension Reduction)
개요

차원 축소
(Dimension Reduction)

피처 선택 / 특성 선택
(feature selection)

피처 추출 / 특성 추출
(feature extraction)

특정 피처에 종속성이 강한 불필요한 피처 제거
→ 데이터의 특징을 잘 나타내는 주요 피처만 선택

기존 피처를 저차원의 중요 피처로 압축해서 추출
→ 새롭게 추출된 피처는 기존의 피처와는 전혀 다른 값

▶ 잠재적인 요소 (Latent Factor) 추출

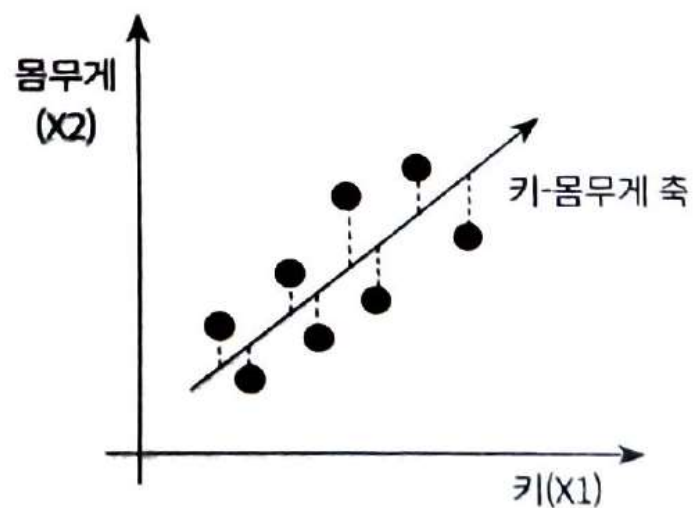
- ▶ PCA (Principal Component Analysis)
- ▶ LDA (Linear Discriminant Analysis)
- ▶ SVD (Singular Value Decomposition)
- ▶ NMF (Non-Negative Matrix Factorization)

[06-02]

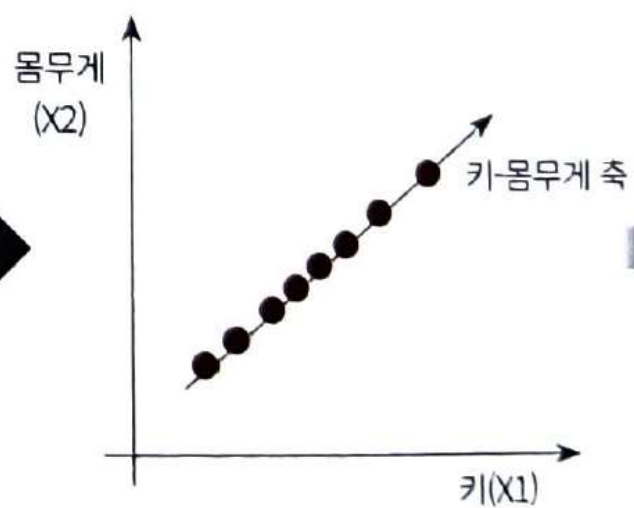
PCA

(Principal Component Analysis,
주성분 분석)

A. 데이터 변동성이 가장 큰 방향으로 축 생성



B. 새로운 축으로 데이터 투영



C. 새로운 축 기준으로 데이터 표현



표준 편차 (Standard Deviation)

$$s = \sqrt{\frac{\sum_{i=1}^n (X_i - \bar{X})^2}{(n-1)}}$$

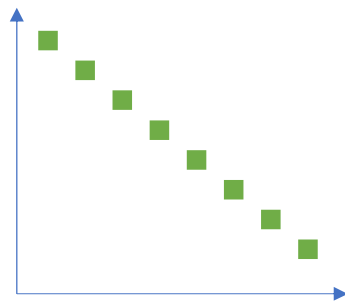
분산 (Variance)

$$s^s = \frac{\sum_{i=1}^n (X_i - \bar{X})^2}{(n-1)}$$

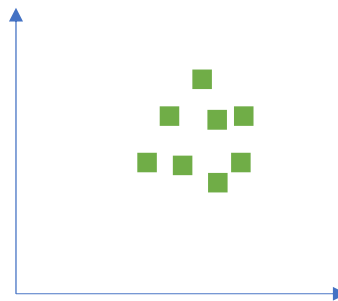
공분산 (Covariance)

Measure of the spread of data in a data set with respect to each other

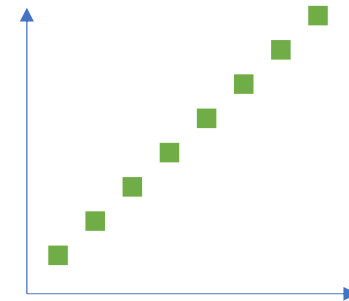
$$cov(X, Y) = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$



Large Negative
Covariance



Nearly Zero
Covariance



Large Positive
Covariance

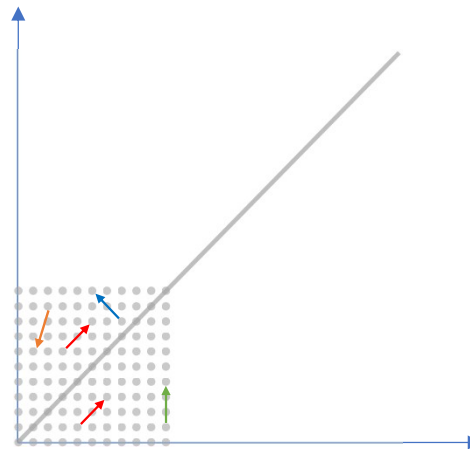
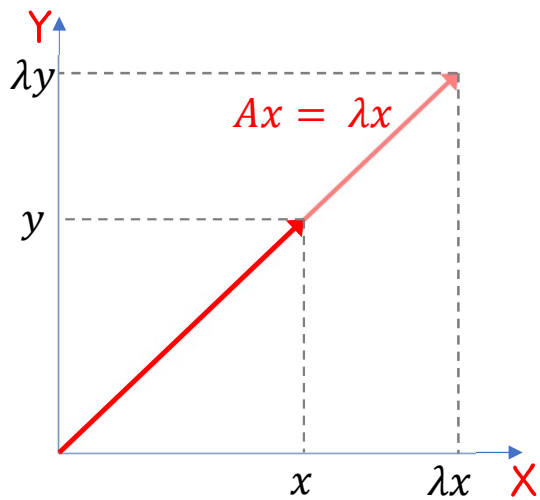
고유 벡터 (Eigen Vector)

행렬 A 의 선형 변환 결과가 자기 자신의 상수배가 되는 0이 아닌 벡터 ($n \times n$ 행렬만)

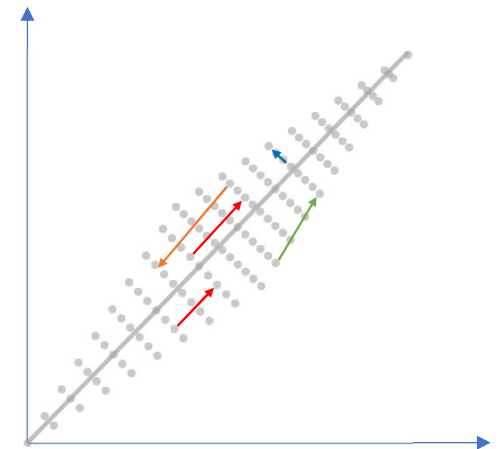
$$Av = \lambda v$$

v : Eigen Vector
 λ : Eigen Value

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} = \lambda \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}$$



Eigen Vector(v) : 데이터들의 방향
Eigen Value(λ) : 방향의 크기



□ PCA (Principal Component Analysis)

① mean=0으로 만들어 줌 (Z-score는 optional)

② Covariance matrix 구성

$$\begin{pmatrix} cov(X_1, X_1) & \cdots & cov(X_1, X_n) \\ \vdots & \ddots & \vdots \\ cov(X_n, X_1) & \cdots & cov(X_n, X_n) \end{pmatrix}$$

③ Eigen Vector와 Eigen Value를 구함

④ ①에서 만든 matrix와 Eigen Vector의 행렬 곱
(Eigen Value가 큰 순서대로...)

https://github.com/whatwant-school/python-ml/blob/main/09-week/09-week_01-PCA.ipynb

```
[6]: from sklearn.decomposition import PCA
```

```
pca = PCA(n_components=2)
```

```
pca.fit(scaled_df)
```

```
iris_pca = pca.transform(scaled_df)
```

```
iris_pca.shape
```

```
[6]: (150, 2)
```

```
[7]: pca_columns=['pca_component_1', 'pca_component_2']
```

```
df_pca = pd.DataFrame(iris_pca, columns=pca_columns)
```


```
df_pca['target'] = iris.target
```


```
df_pca.head(3)
```

```
[7]:
```

	pca_component_1	pca_component_2	target
0	-2.264703	0.480027	0
1	-2.080961	-0.674134	0
2	-2.364229	-0.341908	0

<https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients>

 [Datasets](#) [Contribute Dataset](#) [About Us](#)



Default of credit card clients

Donated on 1/25/2016

This research aimed at the case of customers' default payments in Taiwan and compares the predictive accuracy of probability of default among six data mining methods.

Dataset Characteristics Multivariate	Subject Area Business	Associated Tasks Classification
Feature Type Integer, Real	# Instances 30000	# Features 23


Dataset Information

Additional Information
This research aimed at the case of customers' default payments in Taiwan and compares the predictive accuracy of probability of default among six data mining methods. From the perspective of risk management, the result of predictive accuracy of the estimated probability of default will be more valuable than the binary result of classification - credible or...
[SHOW MORE](#)

Has Missing Values?
No

Introductory Paper

[DOWNLOAD](#)

 [IMPORT IN PYTHON](#)

[CITE](#)

3 citations
84535 views

Creators
I-Cheng Yeh

DOI
10.24432/C55S3H

License
This dataset is licensed under a [Creative Commons Attribution 4.0 International \(CC BY 4.0\)](#) license.

This allows for the sharing and adaptation of the datasets for any purpose, provided that the appropriate credit is given.

대만에서 고객의 채무 불이행 사례를 대상으로 하여, 채무 불이행 확률 예측

https://github.com/whatwant-school/python-ml/blob/main/09-week/09-week_02-Credit.ipynb

```
[11]: from sklearn.preprocessing import StandardScaler
      from sklearn.decomposition import PCA

      scaler = StandardScaler()
      scaled_df = scaler.fit_transform(X_features)

      pca = PCA(n_components=6)
      df_pca = pca.fit_transform(scaled_df)

      scores_pca = cross_val_score(rcf, df_pca, y_target, scoring='accuracy', cv=3)

      print('CV=3 인 경우의 PCA 변환된 개별 Fold세트별 정확도:', scores_pca)
      print('PCA 변환 데이터 셋 평균 정확도: {0:.4f}'.format(np.mean(scores_pca)))

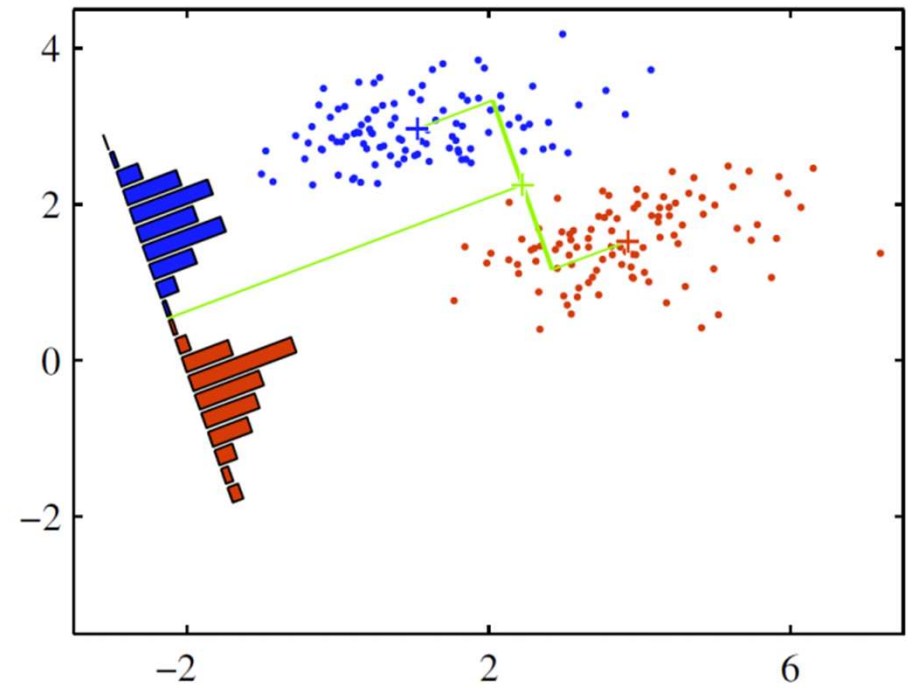
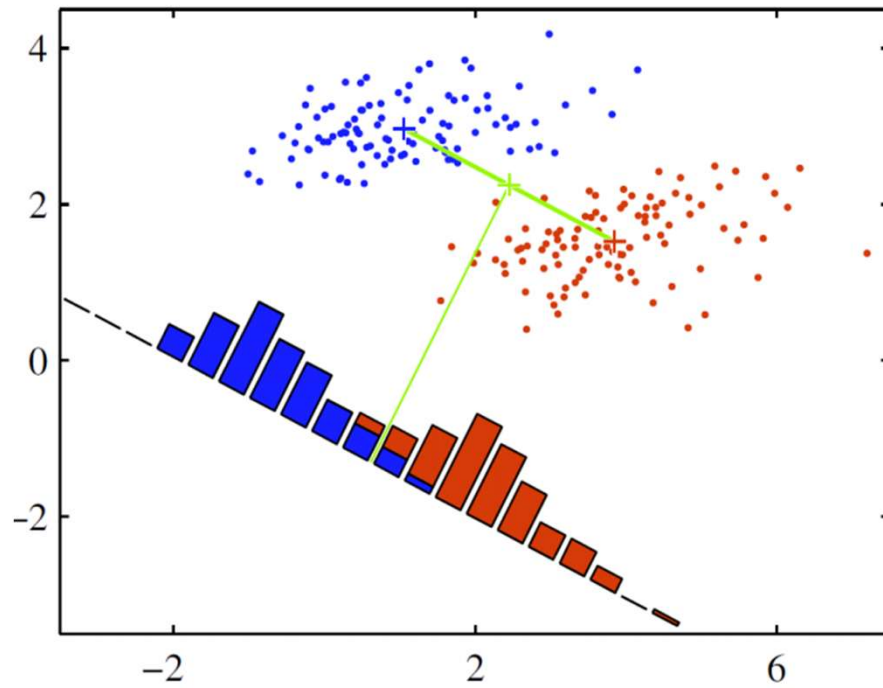
      CV=3 인 경우의 PCA 변환된 개별 Fold세트별 정확도: [0.7914 0.7975 0.8009]
      PCA 변환 데이터 셋 평균 정확도:0.7966
```

[06-03]

LDA

(Linear Discriminant Analysis,
선형 판별 분석법)

데이터 분포를 학습해 결정 경계(Decision Boundary)를 만들어 분류하는 모델



□ LDA (Linear Discriminant Analysis)

① 클래스 내부와 클래스 간 분산 행렬을 구한다. 개별 피처의 평균 벡터(mean vector) 기반으로 구한다.

② Eigen Vector(고유 벡터)로 분해

$$S_W^T S_B = [e_1 \dots e_n] \begin{bmatrix} \lambda_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_n \end{bmatrix} \begin{bmatrix} e_1^T \\ \vdots \\ e_n^T \end{bmatrix}$$

③ Eigen Value(고유 값)이 가장 큰 순으로 K개(LDA 변환 차수만큼) 추출

④ Eigen Value(고유 값)이 가장 큰 순으로 추출된 Eigen Vector(고유 벡터)를 이용해 새롭게 입력 데이터 변환

https://github.com/whatwant-school/python-ml/blob/main/09-week/09-week_03-LDA.ipynb

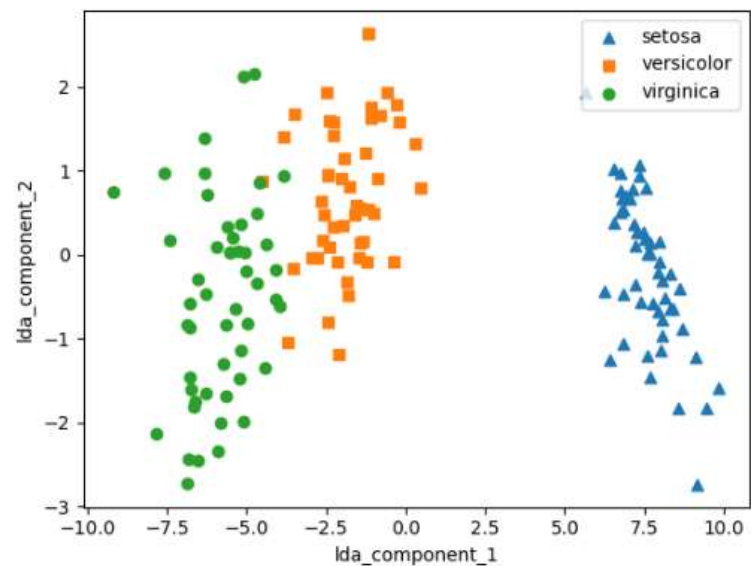
```
[4]: import pandas as pd
import matplotlib.pyplot as plt

lda_columns=['lda_component_1','lda_component_2']
lda_df = pd.DataFrame(lda_iris, columns=lda_columns)
lda_df['target'] = iris.target

markers=['^', 's', 'o']
for i, marker in enumerate(markers):
    x_axis_data = lda_df[lda_df['target']==i]['lda_component_1']
    y_axis_data = lda_df[lda_df['target']==i]['lda_component_2']

    plt.scatter(x_axis_data, y_axis_data, marker=marker, label=iris.target_names[i])

plt.legend(loc='upper right')
plt.xlabel('lda_component_1')
plt.ylabel('lda_component_2')
plt.show()
```



[06-04]

SVD

(Singular Value Decomposition,
특잇값 분해)

PCA : 정방행렬만 Eigen Vector로 분해 가능

vs.

SVD : 행과 열의 크기가 다른 행렬도 적용 가능

$$A = U\Sigma V^T$$

$$(m \times n) = (m \times m)(m \times n)(n \times n)$$

U : 특이 벡터 (Singular Vector)

V : 특이 벡터 (Singular Vector)

Σ : 대각 행렬 (대각 값만 0이 아니고, 나머지는 모두 0)

→ 특잇값 (Singular Value)

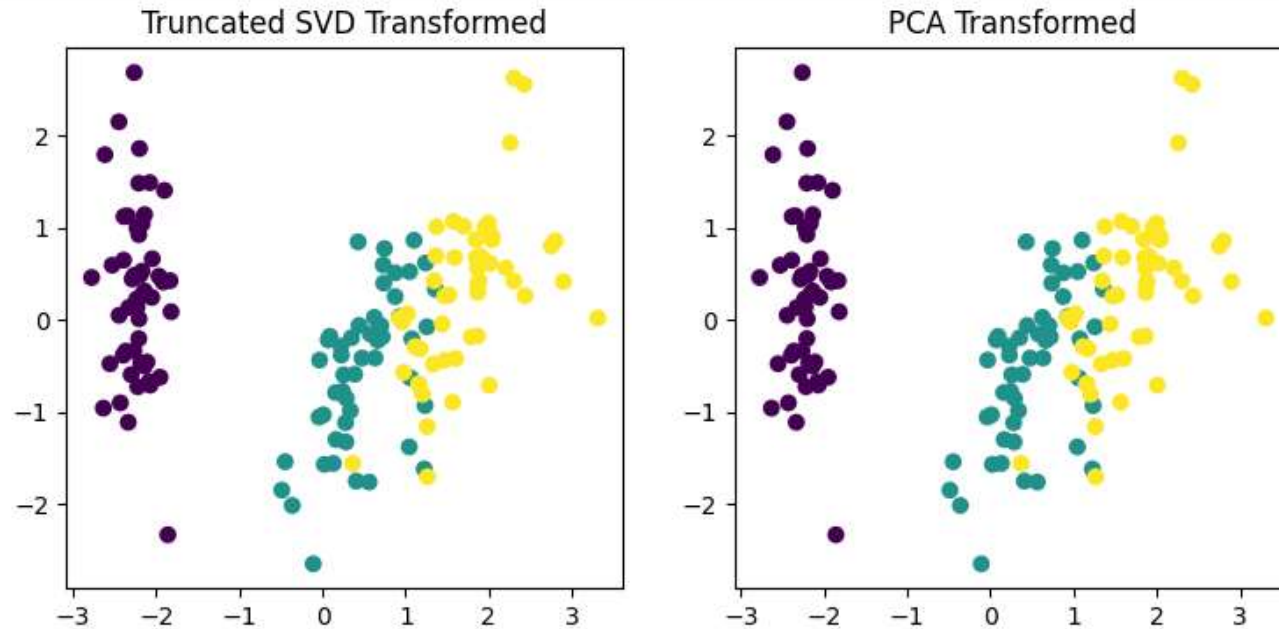
SVD

$$\begin{array}{c} \boxed{A} \\ (m \times n) \end{array} = \begin{array}{c} \boxed{U} \\ (m \times m) \end{array} \begin{array}{c} \boxed{\begin{array}{c} \text{0} \\ \text{0} \end{array}} \\ (m \times n) \end{array} \begin{array}{c} \boxed{V^T} \\ (n \times n) \end{array}$$

Truncated
SVD

$$\begin{array}{c} \boxed{A} \\ (m \times n) \end{array} = \begin{array}{c} \boxed{\begin{array}{c} \text{ } \\ \text{ } \end{array}} \boxed{U} \\ (m \times p) \end{array} \begin{array}{c} \boxed{\begin{array}{c} \text{ } \\ \text{0} \end{array}} \\ (p \times p) \end{array} \begin{array}{c} \boxed{\begin{array}{c} \text{ } \\ \text{ } \end{array}} \boxed{V^T} \\ (p \times n) \end{array}$$

https://github.com/whatwant-school/python-ml/blob/main/09-week/09-week_04-SVD.ipynb



```
[15]: print((iris_pca - iris_tsvd).mean())  
      print((pca.components_ - tsvd.components_).mean())
```

```
2.342304591026097e-15  
-1.3877787807814457e-17
```

[06-05]

NMF

(Non-Negative Matrix Factorization,
비-음수 행렬 분해)

V

	C1	C2	C3	C4	C5	C6
R1						
R2						
R3						
R4						

\approx

W

	LF1	LF2
R1		
R2		
R3		
R4		

\times

H

	C1	C2	C3	C4	C5	C6
LF1						
LF2						

https://github.com/whatwant-school/python-ml/blob/main/09-week/09-week_05-NMF.ipynb

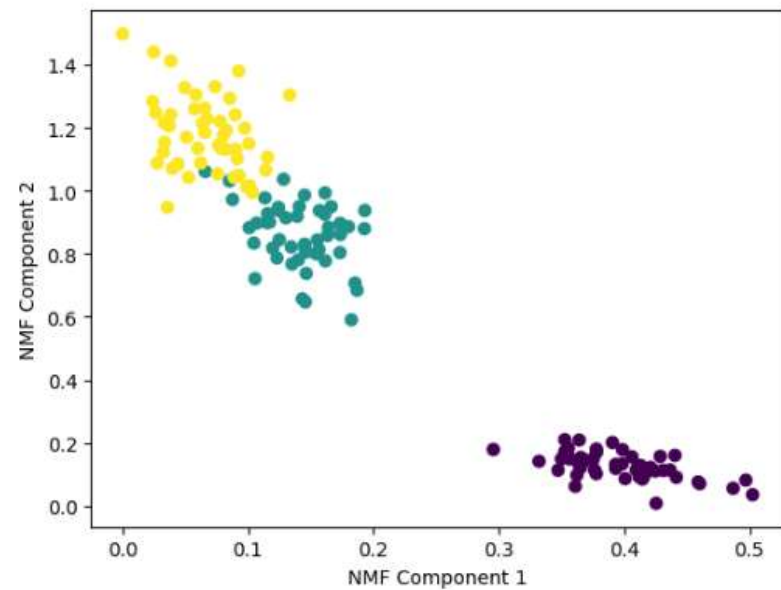
```
[2]: from sklearn.decomposition import NMF
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt

iris = load_iris()
iris_fts = iris.data

nmf = NMF(n_components=2)
nmf.fit(iris_fts)
iris_nmf = nmf.transform(iris_fts)

plt.scatter(x=iris_nmf[:,0], y=iris_nmf[:,1], c=iris.target)
plt.xlabel('NMF Component 1')
plt.ylabel('NMF Component 2')

plt.show()
```



You've really worked hard today

Next Week ~ ?

07

군집화

01. K-평균 알고리즘 이해	431
사이킷런 KMeans 클래스 소개	432
K-평균을 이용한 붓꽃 데이터 세트 군집화	433
군집화 알고리즘 테스트를 위한 데이터 생성	437
02. 군집 평가(Cluster Evaluation)	441
실루엣 분석의 개요	442
붓꽃 데이터 세트를 이용한 군집 평가	443
군집별 평균 실루엣 개수의 시각화를 통한 군집 개수 최적화 방법	445
03. 평균 이동	449
평균 이동(Mean Shift)의 개요	449
04. GMM(Gaussian Mixture Model)	455
GMM(Gaussian Mixture Model) 소개	455
GMM을 이용한 붓꽃 데이터 세트 군집화	457
GMM과 K-평균의 비교	459
05. DBSCAN	463
DBSCAN 개요	463
DBSCAN 적용하기 - 붓꽃 데이터 세트	467
DBSCAN 적용하기 - make_circles() 데이터 세트	471
06. 군집화 실습 - 고객 세그먼테이션	474
고객 세그먼테이션의 정의와 기법	474
데이터 세트 로딩과 데이터 클렌징	475
RFM 기반 데이터 가공	478
RFM 기반 고객 세그먼테이션	481
07. 정리	487

Who ~ ?

See you Next Weekend ~ ?