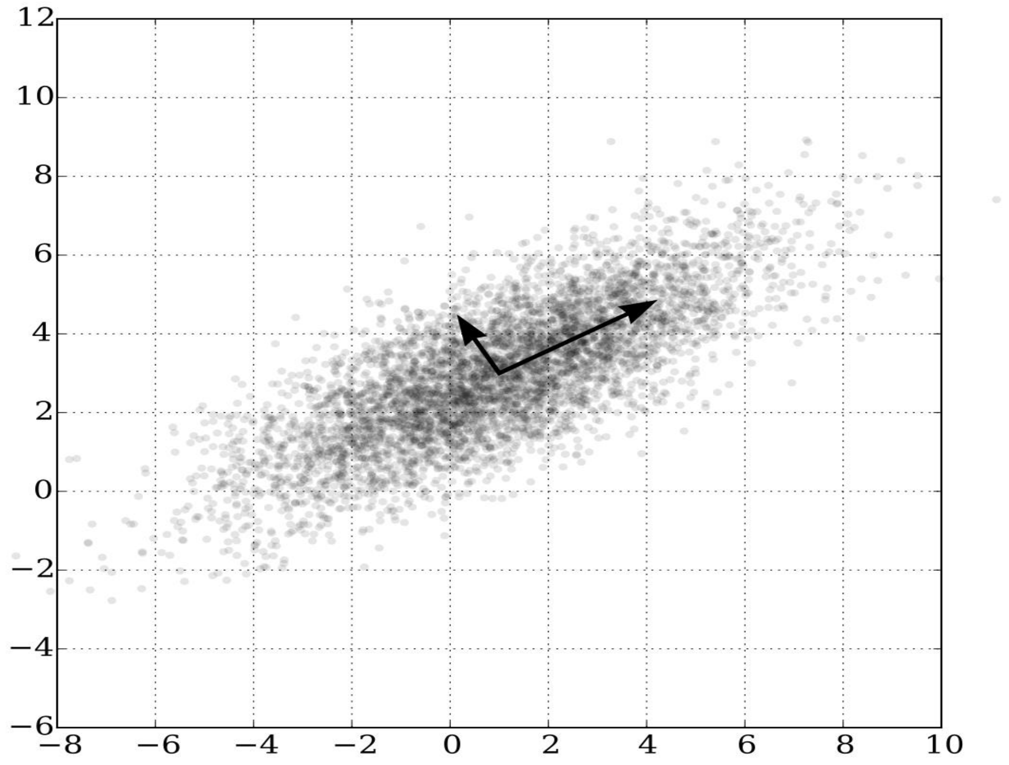
Unsupervised Learning

경희대학교 컴퓨터공학과
2019102191 신주영

Table of Contents

경희대학교
KYUNG HEE UNIVERSITY

Fundamental Concept



Fundamental Concept

- Using PCA, correlated multivariate data can be concisely represented as the main component.
- PCA is a method used to reduce variables in data.
- It is a representative dimension reduction method that can be applied to data that is correlated between variables.
- Dimension reduction means 'It is expressed as several variables while maintaining the characteristics of data with many variables.'
- It helps reduce complexity when analyzing multivariate data.

Fundamental Concept

- Two ways to reduce variables
  - Choose only important variables and do not use the remaining variables.
  - constructing a new variable from the original data variable
- PCA reduces variables by constructing a new variable from the original data variable
- The data represented in a high-dimensional space is represented by a lower-dimensional variable.
- We call lower-dimensional axis as a principal component

Fundamental Concept

1. Find the direction and importance necessary for data principal component analysis.
   - The direction of the line is the direction of the data.
   - Length represents importance.
     1. The direction of the data is determined by how much weight is assigned to the data variable when configuring a new variable.
     2. The importance of data is related to changes in variables.
2. The orthogonal line is used as a new axis and converted into original data.
   1. The changed data at this time is called the principal component score.
   2. It is called the first principal component and the second principal component from the value of the axis with the greatest importance among the principal components.

Algorithm

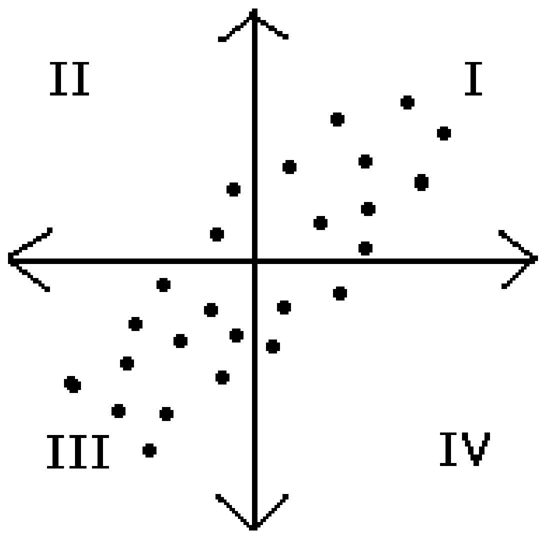
- The principal component analysis calculates the principal component in the following order.
    1. Calculate variance covariance matrix
    2. Solve eigenvalue problems and calculate eigenvalue of eigenvectors
    3. Construct data in each principal component direction

Algorithm
Covariance

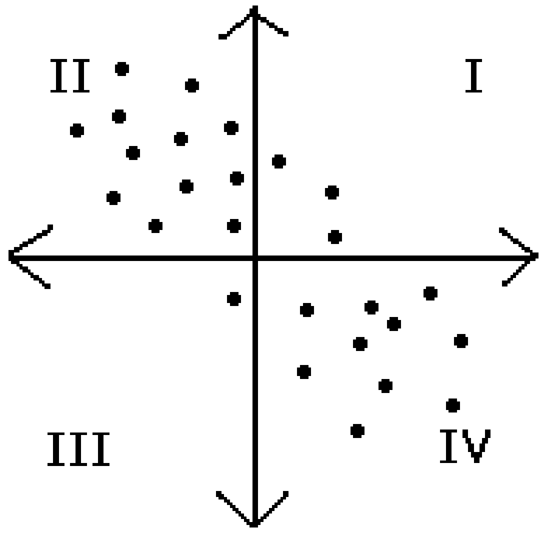
- The covariance is a value representing the relationship between data points x and y.
- The mean was calculated after multiplying the difference between the mean and the data point of each of x and y.
- $Cov(x,y) = s_{xy} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y) = E(X - \mu_x)(Y - \mu_y)$
- In this case, μ represents the average of each of X and Y.
- If $Cov(x,y) > 0$ than X increases, so does Y.
- If $Cov(x,y) < 0$ than X increases, Y decreases.
- If $Cov(x,y) = 0$ than X and Y has no correlation.

Algorithm

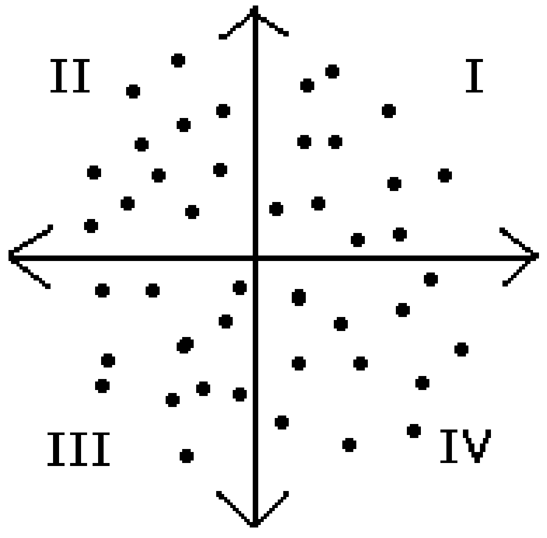
Covariance



(a) Positive Relationship



(b) Negative Relationship



(c) No Relationship

경희대학교
KYUNG HEE UNIVERSITY

Algorithm

Correlation

- An index that does not depend on the size of the variance represents a correlation coefficient.
- The covariance s_{xy} of the two data x and y is divided by the standard deviations s_x and s_y.
- $r_{xy} = \frac{s_{xy}}{s_x s_y}$
- The correlation coefficient represents the relationship between data like covariance, but since it is divided by each standard deviation, a value between -1 and 1 is obtained.

Algorithm

Covariance Matrix

- The covariance matrix is made into a matrix by calculating the covariance of each of the two or more data.
- $\Sigma = \begin{bmatrix} s_{x_1x_1} & \cdots & s_{x_1x_n} \\ \vdots & \ddots & \vdots \\ s_{x_nx_1} & \cdots & s_{x_nx_n} \end{bmatrix}$
- $s_{x_ix_j}$ represents the covariance of data point x_i and x_j

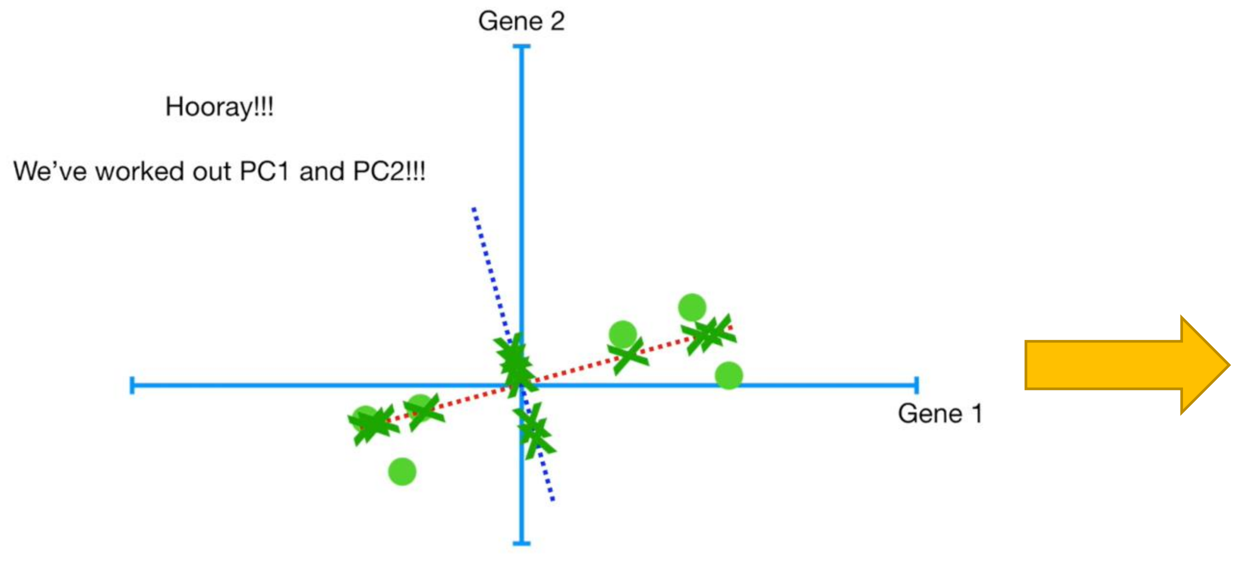
Algorithm

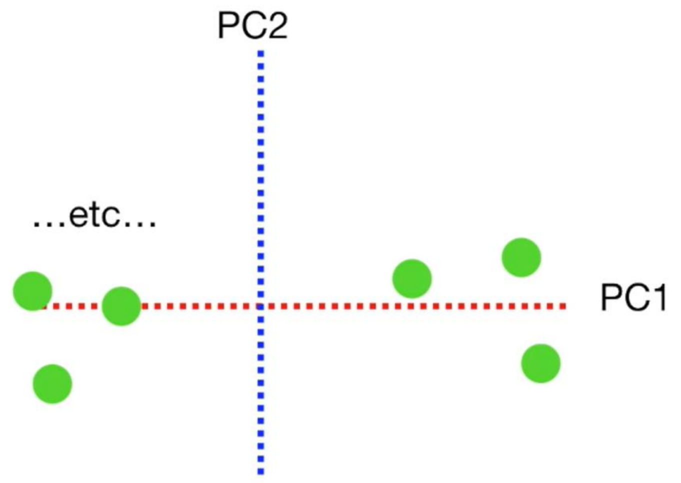
Correlation Matrix

- The correlation matrix is made into a matrix by calculating the correlation of each of the two or more data.
- $\Sigma = \begin{bmatrix} r_{x_1x_1} & \cdots & r_{x_1x_n} \\ \vdots & \ddots & \vdots \\ r_{x_nx_1} & \cdots & r_{x_nx_n} \end{bmatrix}$
- $r_{x_ix_j}$ represents the correlation of data point x_i and x_j

경희대학교
KYUNG HEE UNIVERSITY

Algorithm



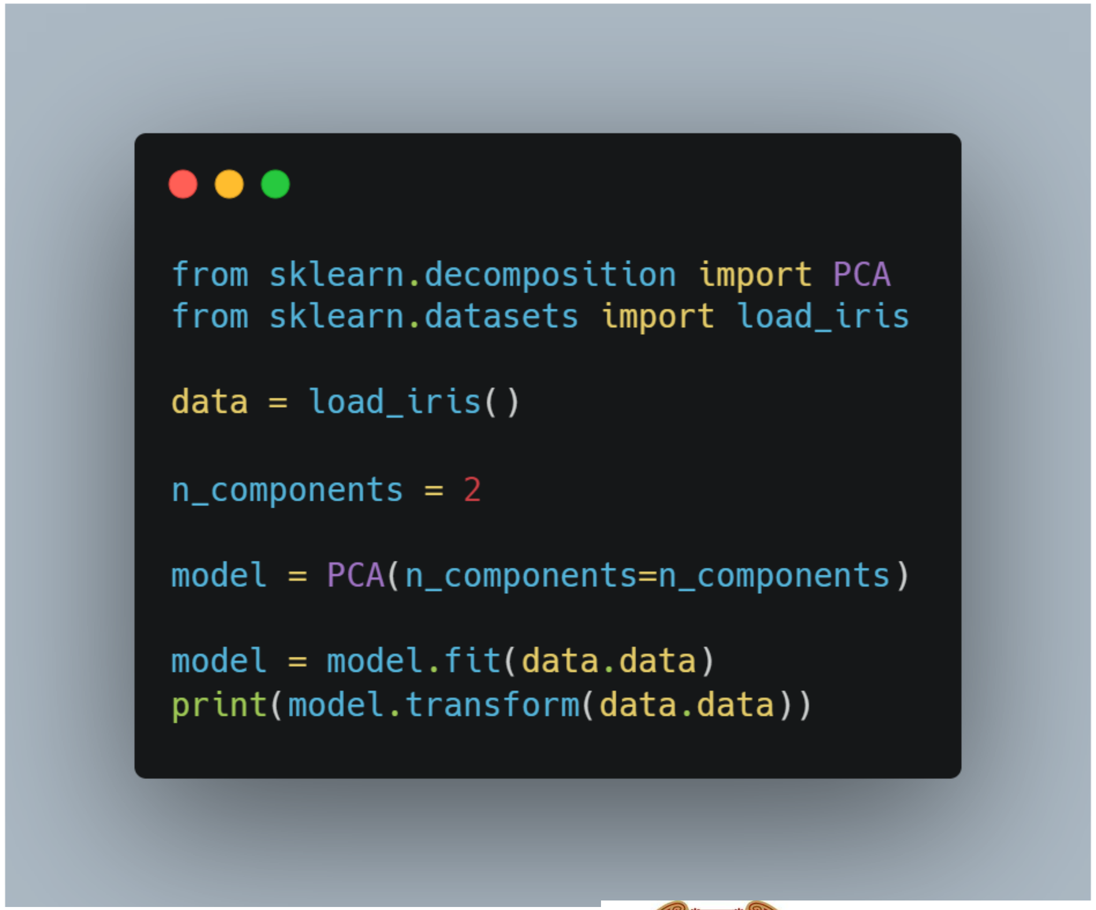


Algorithm

- The direction of the data is related to the eigenvector.
- The importance of data is related to eigenvalues.
- If the eigenvalue calculated for each principal component is divided by the total sum of several eigenvalues, the importance of the principal component can be expressed as a ratio, which is called the contribution rate.
- The value obtained by adding the contribution rate sequentially from the first main component is called the cumulative contribution rate.

Sample Code

- Import libraries
- Get iris dataset
- Make PCA model
- Train data into model
- Print transformed data



```
from sklearn.decomposition import PCA
from sklearn.datasets import load_iris

data = load_iris()

n_components = 2

model = PCA(n_components=n_components)

model = model.fit(data.data)
print(model.transform(data.data))
```

경희대학교
KYUNG HEE UNIVERSITY

Unsupervised Learning

Latent Semantic Analysis, LSA

경희대학교 컴퓨터공학과
2019102191 신주영

Fundamental Concept

- It is used in the field of data information retreival
- By using LSA, we can see the latent relevance of words from a large amount of document data.

Algorithm

1. Convert the sentence below to matrix X
   - 자동차로 회사에 출근했다.
   - 차로 출근했다.
   - 레스토랑에서 햄버거를 먹는다.
   - 레스토랑에서 파스타를 먹는다.

	문장1	문장2	문장3	문장4
자동차	1	0	0	0
회사	1	0	0	0
출근	1	1	0	0
차	0	1	0	0
레스토랑	0	0	1	1
햄버거	0	0	1	0
먹다	0	0	1	1
파스타	0	0	0	1

경희대학교
KYUNG HEE UNIVERSITY

Algorithm

2. Decompose the matrix X.
   2. The matrix X, which is an 8X4 matrix, is represented by the product of the 8X4 matrix U, the 4X4 matrix D, and the 4X4 matrix V^T.
      2. Matrix U: Matrix with transformation information of characteristics made of words
      3. Matrix D: Matrix with information importance
      4. Matrix V: Matrix with word features and document conversion information

$$X = UDV^T = \begin{matrix} 0.00 & -0.45 & -0.45 & 0.00 \\ 0.00 & -0.45 & -0.45 & 0.00 \\ \vdots & \vdots & \vdots & \vdots \\ -0.32 & 0.00 & 0.00 & -0.71 \end{matrix} \times \begin{matrix} 2.24 & 0 & 0 & 0 \\ 0 & 1.90 & 0 & 0 \\ 0 & 0 & 1.18 & 0 \\ 0 & 0 & 0 & 1.00 \end{matrix} \times \begin{matrix} 0.00 & 0.00 & -0.71 & -0.71 \\ -0.85 & -0.53 & 0.00 & 0.00 \\ -0.53 & 0.85 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.71 & -0.71 \end{matrix}$$

경희대학교
KYUNG HEE UNIVERSITY

Algorithm

- The matrix **D** is a diagonal matrix. **Diagonal elements list values in order of greater importance of information.**
- Therefore, if we reduce the dimension to three matrices, we should look at **D** carefully.
- Since the original data has four features, let's say that the dimension is reduced to two features.

Algorithm

1. From the diagonal element values of a matrix D with four features, choose two elements of greater importance to form a 2X2 diagonal matrix.
2. The other two matrices are also changed to correspond to the elements selected in D.
3. The matrix U deletes the entire elements of the third and fourth columns and replaces them with an 8X2 matrix.
4. The matrix V^T deletes all elements of the third and fourth rows to form a 2X4 matrix.

$$\hat{X} = \hat{U}\hat{D}\hat{V}^T = \begin{matrix} 0.00 & -0.45 \\ 0.00 & -0.45 \\ \vdots & \vdots \\ -0.32 & 0.00 \end{matrix} \times \begin{matrix} 2.24 & 0 \\ 0 & 1.90 \end{matrix} \times \begin{matrix} 0.00 & 0.00 & -0.71 & -0.71 \\ -0.85 & -0.53 & 0.00 & 0.00 \end{matrix}$$

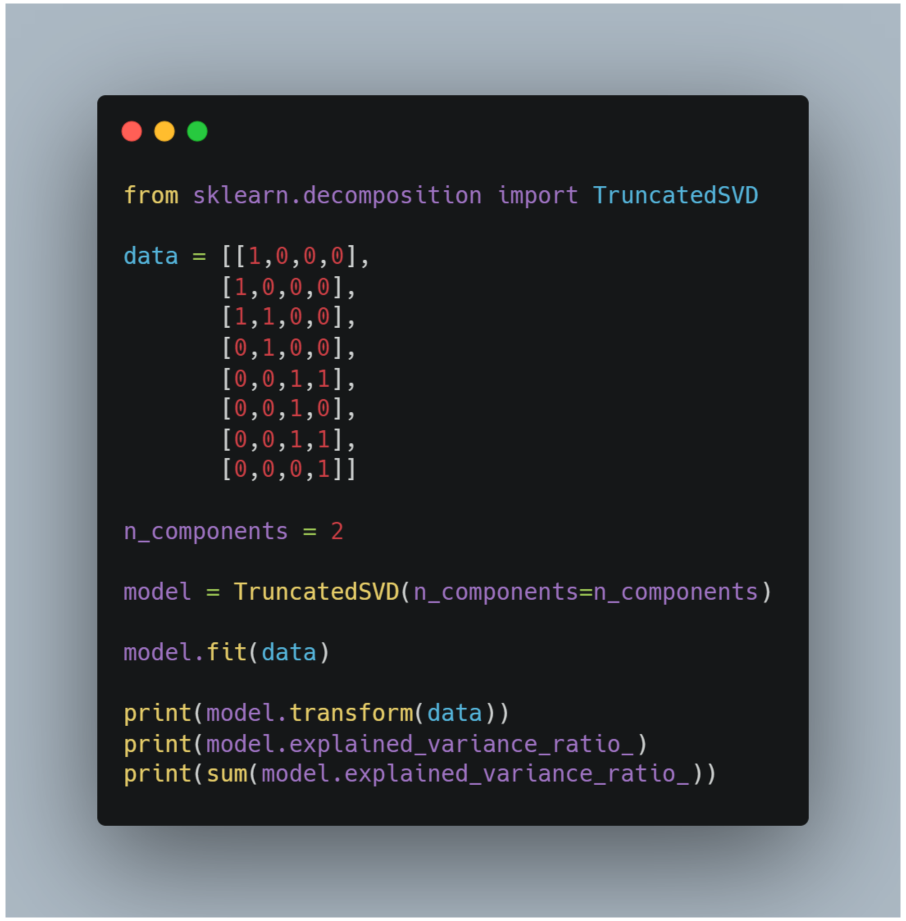
Algorithm

- Table of 2 features divided by latent variables

	A	B
자동차	0.00	0.85
회사	0.00	0.85
출근	0.00	1.38
차	0.00	0.53
레스토랑	1.41	0.00
햄버거	0.71	0.00
먹다	1.41	0.00
파스타	0.71	0.00

Sample Code

- Import library
- Set data
- Set dimension to reduce
- Print transformed data
- Print Contribution Rate



```
from sklearn.decomposition import TruncatedSVD

data = [[1,0,0,0],
        [1,0,0,0],
        [1,1,0,0],
        [0,1,0,0],
        [0,0,1,1],
        [0,0,1,0],
        [0,0,1,1],
        [0,0,0,1]]

n_components = 2

model = TruncatedSVD(n_components=n_components)

model.fit(data)

print(model.transform(data))
print(model.explained_variance_ratio_)
print(sum(model.explained_variance_ratio_))
```

Points to note

1. It is difficult to properly explain the relevance of words with the converted matrix. If each dimension is orthogonal or the matrix element is negative, it is difficult to properly explain the relevance.
2. The calculation can take a very long time.

Unsupervised Learning

Non-negative Matrix Factorization, NMF

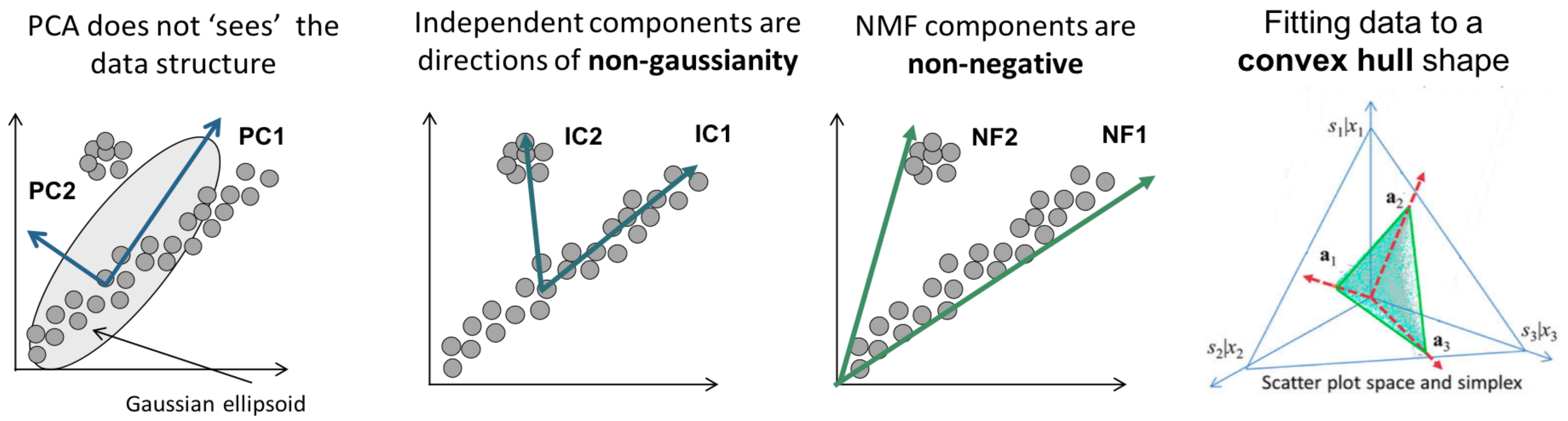
경희대학교 컴퓨터공학과
2019102191 신주영

Fundamental Concept

- NMF is a dimensionality reduction method in which features and labels are both positive.
- Easy to model when dealing with image data, etc
- It is an algorithm that is applied only when all the components of the original matrix are positive.
  - The original matrix element must be positive.
  - After matrix decomposition, the matrix element is positive.
  - In NMF, there is no restriction that each dimension is orthogonal.

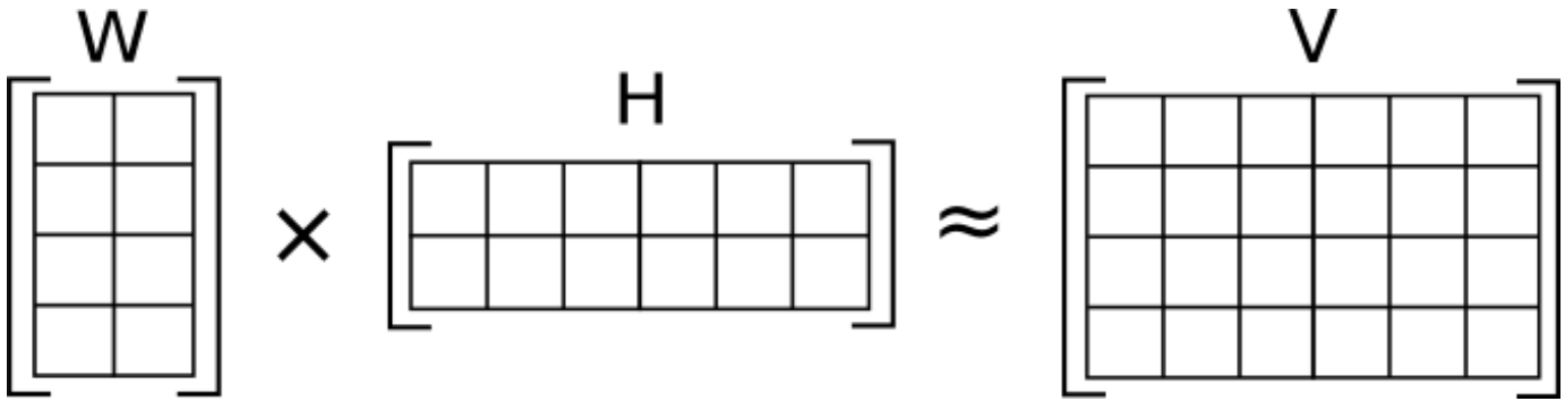
Fundamental Concept

- PCA always uses orthogonal eigenvectors, making it difficult to find associations between data, but NMF can grasp the nature of the data.


PCA does not 'sees' the data structure
PC1
PC2
Gaussian ellipsoid
Independent components are directions of **non-gaussianity**
IC2
IC1
NMF components are **non-negative**
NF2
NF1
Fitting data to a **convex hull** shape
$s_1|x_1$
$s_2|x_2$
$s_3|x_3$
$\mathbf{a}_1$
$\mathbf{a}_2$
$\mathbf{a}_3$
Scatter plot space and simplex

Fundamental Concept

- NMF do dimensionality reduction by decompose original data to 2 matrix

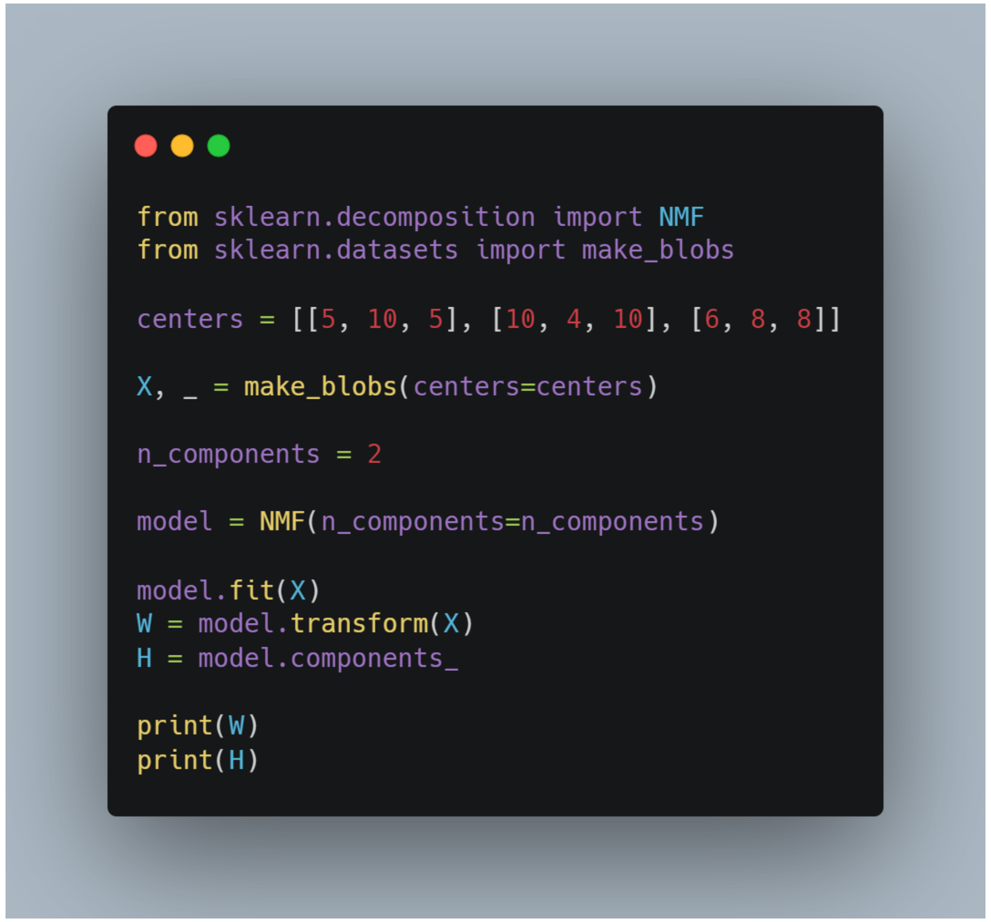


Fundamental Concept

- Calculation Process
  - Initialize W, H as a positive value
  - Treat H as a constant and renew W
  - Treat W as a constant and renew H
  - End calculation when W, H converges to the original matrix

Sample Code

- Imort libraries
- Set data
- Set dimension to reduce
- Make model
- Train data
- Print W matrix and H matrix



```python
from sklearn.decomposition import NMF
from sklearn.datasets import make_blobs

centers = [[5, 10, 5], [10, 4, 10], [6, 8, 8]]

X, _ = make_blobs(centers=centers)

n_components = 2

model = NMF(n_components=n_components)

model.fit(X)
W = model.transform(X)
H = model.components_

print(W)
print(H)
```

Unsupervised Learning

Latent Dirichlet Allocation, LDA

경희대학교 컴퓨터공학과
2019102191 신주영

경희대학교
KYUNG HEE UNIVERSITY

Fundamental Concept

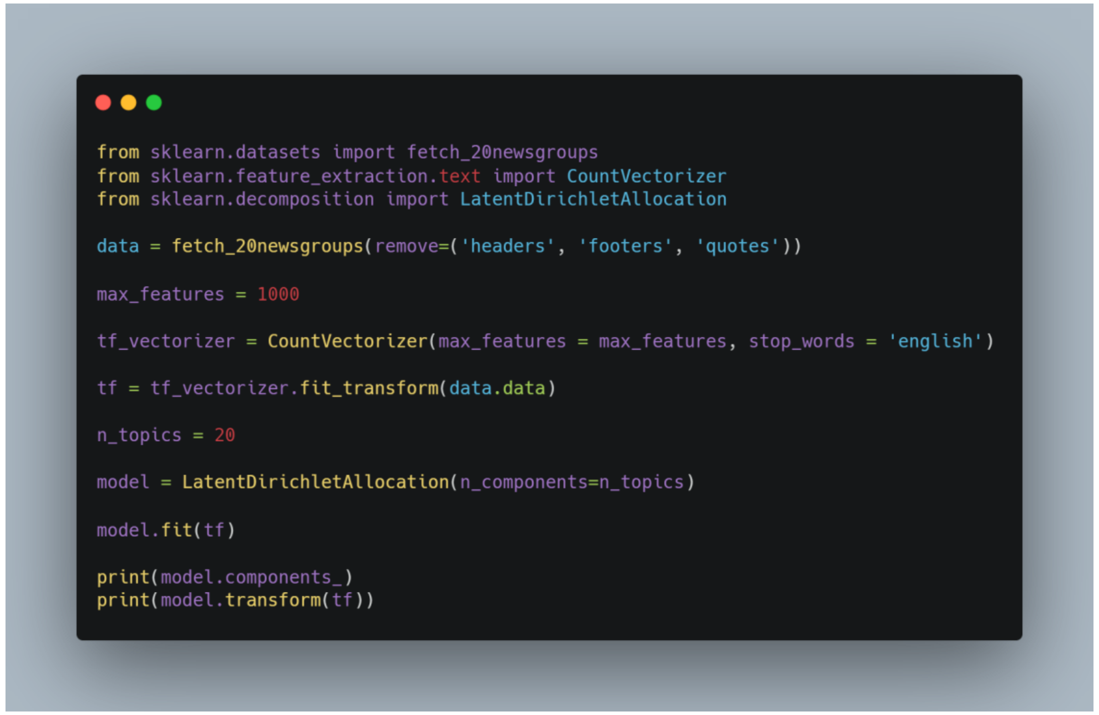
- One of the dimensional reduction methods suitable for modeling documents.
- Various topics can be covered by using words in the document as input data.
- It is used in NLP, etc
- Create latent topics with the words in the document to indicate what each document is composed of.
- You can explain that there are multiple topics in a document.

Algorithm

1. Assign a topic randomly to each word in a document
2. Calculate the probability of each topic in a document from the topic you assign to the word
3. Calculate the probability of words by topic from the topic you assign to the word
4. Reassign the topic to each word in the document with the probability of multiplying by 2 and 3
5. Repeat courses 2 through 4 until converging conditions

Sample Code

- Import libraries
- Make dataset
- Set max feature
- Set stop words
- Train data
- Print results



```
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation

data = fetch_20newsgroups(remove=('headers', 'footers', 'quotes'))

max_features = 1000

tf_vectorizer = CountVectorizer(max_features = max_features, stop_words = 'english')

tf = tf_vectorizer.fit_transform(data.data)

n_topics = 20

model = LatentDirichletAllocation(n_components=n_topics)

model.fit(tf)

print(model.components_)
print(model.transform(tf))
```

Disposal of stop word

- Topic 4
  :00 10 25 15 12 11 16 20 14 13 18 30 50 17 55 40 21 …
- Topic 6
  :people said know did don didn just time went like say think told
- **As described above, the distribution of topics that are difficult to intuitively explain the topic can be improved by learning to exclude stop word.**

경희대학교 KYUNG HEE UNIVERSITY

Unsupervised Learning

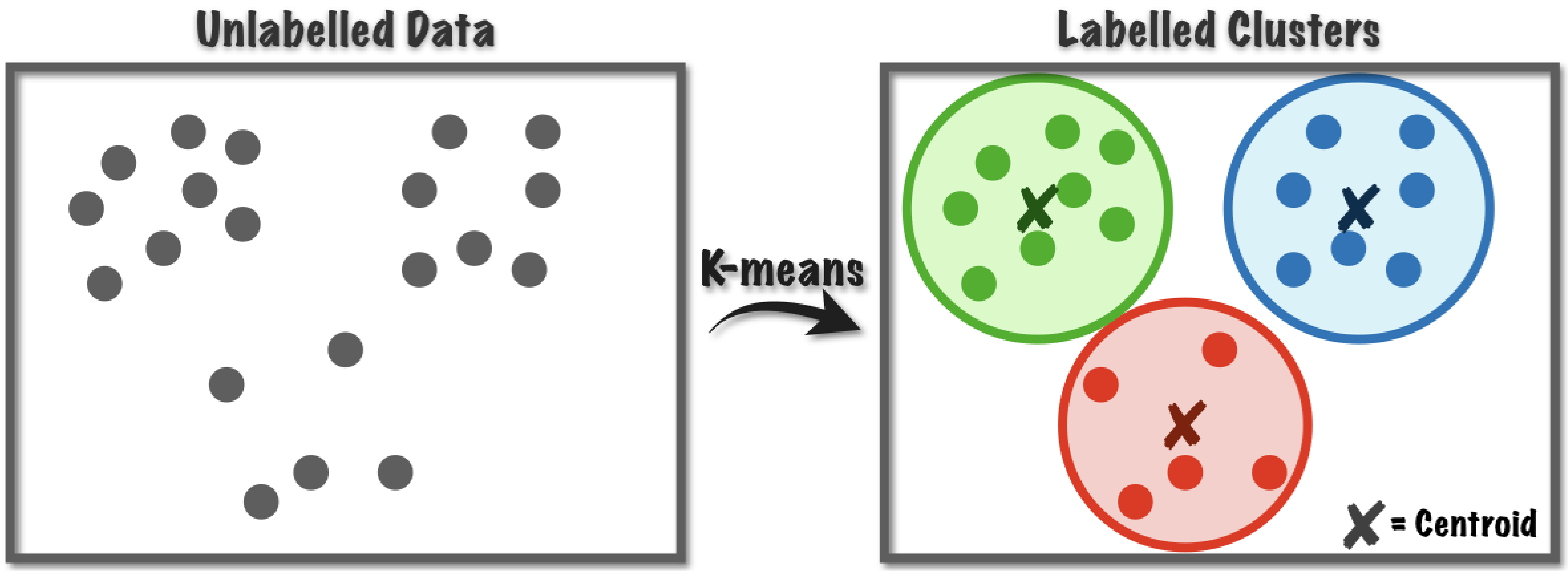
K-Means Algorithm

경희대학교 컴퓨터공학과
2019102191 신주영

Fundamental Concept

- A method of dividing data of similar characteristics into groups is called clustering.
- The group to which the data points belong is determined as the group whose distance between the data points and the center of each group is closest.
- Determining the center of this group is an important operation of the K-means algorithm.

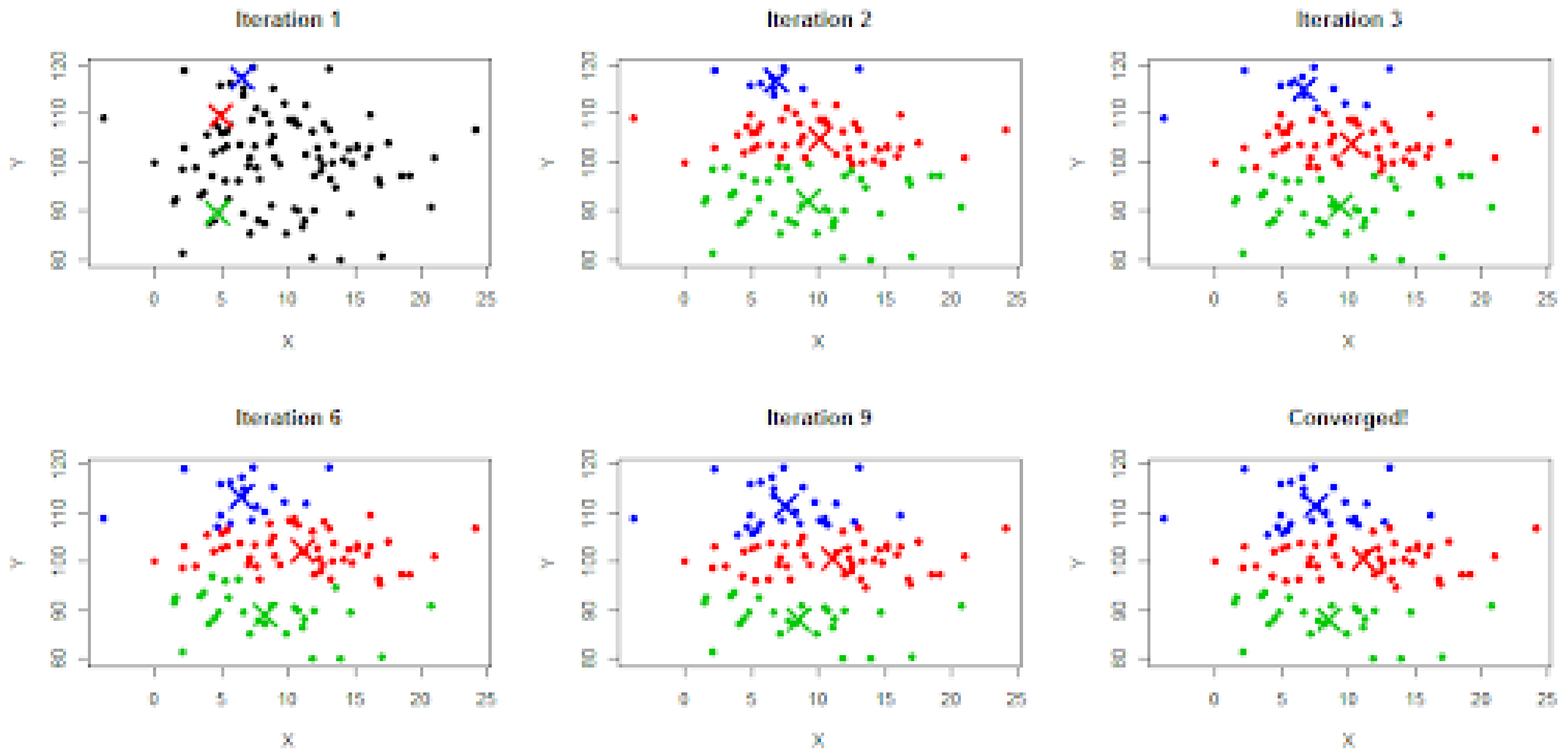
Fundamental Concept



Algorithm

- K-means algorithm's learning process
    1. Choose and center the appropriate number of data pointers by the number of groups
    2. Calculates the distance between a data point and each center to determine the nearest center as the group to which that data point belongs
    3. Average data points for each group and set them as new centers
    4. Repeat steps 2-3 until convergence

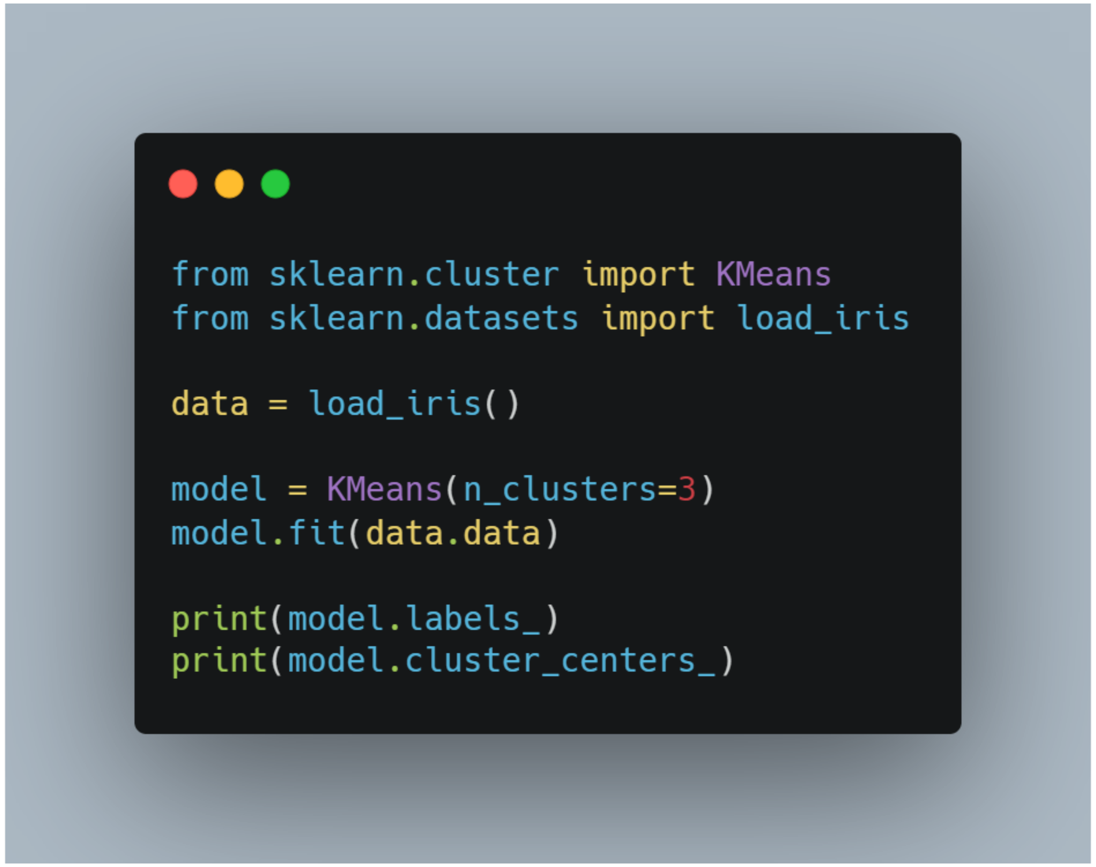
Algorithm



경희대학교
KYUNG HEE UNIVERSITY

Sample Code

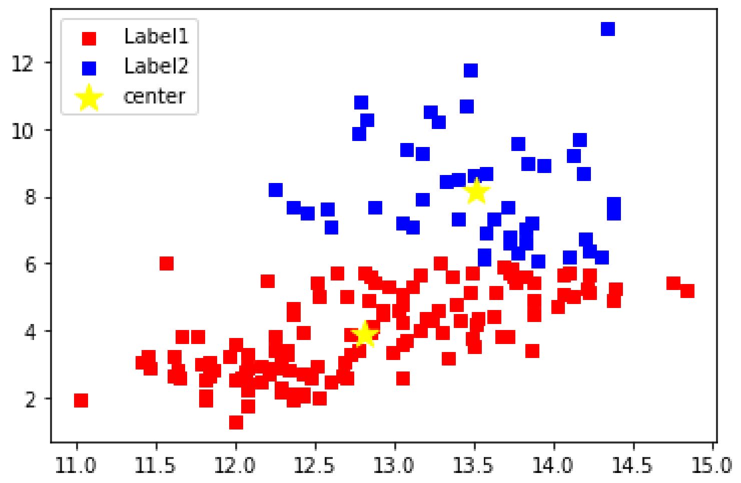
- Import libraries
- Load iris dataset
- Make K-Means model
- Set n_cluster
- Fit data
- Print labels
- Print cluster's center

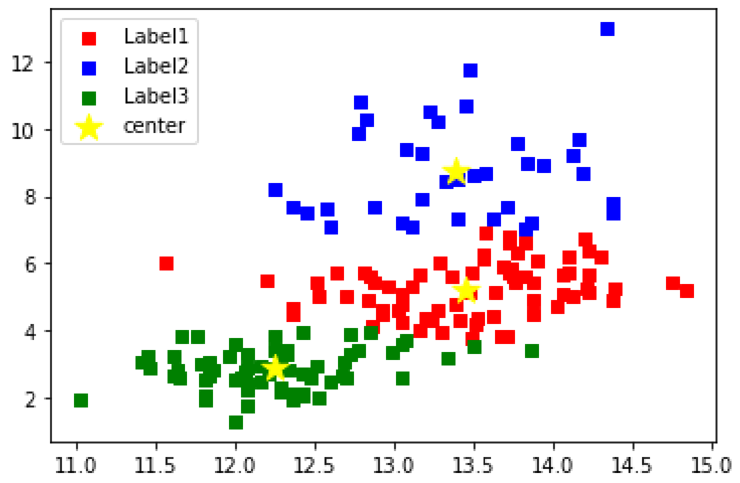


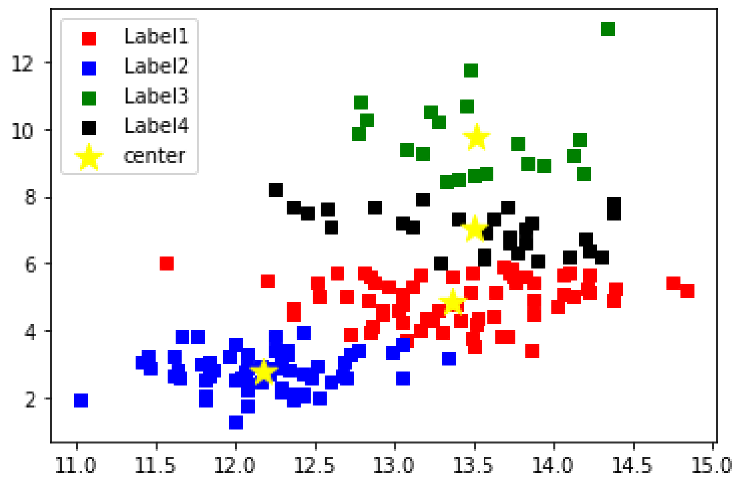
```
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

data = load_iris()

model = KMeans(n_clusters=3)
model.fit(data.data)

print(model.labels_)
print(model.cluster_centers_)
```

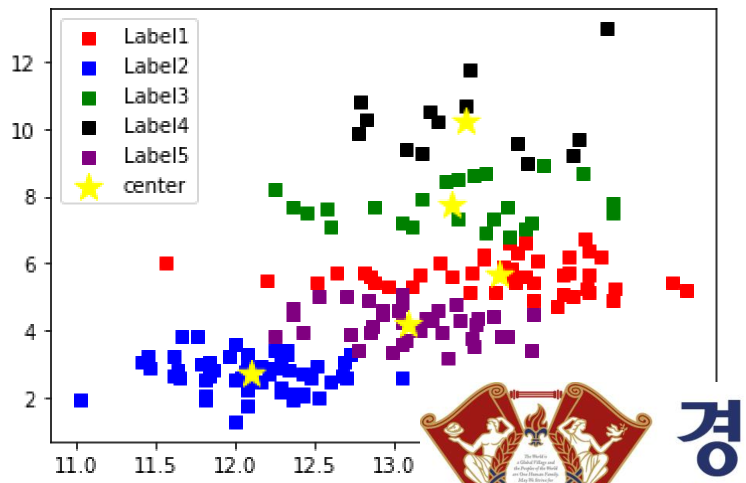
Visualization

according to n_clusters (using sepal length and sepal width)







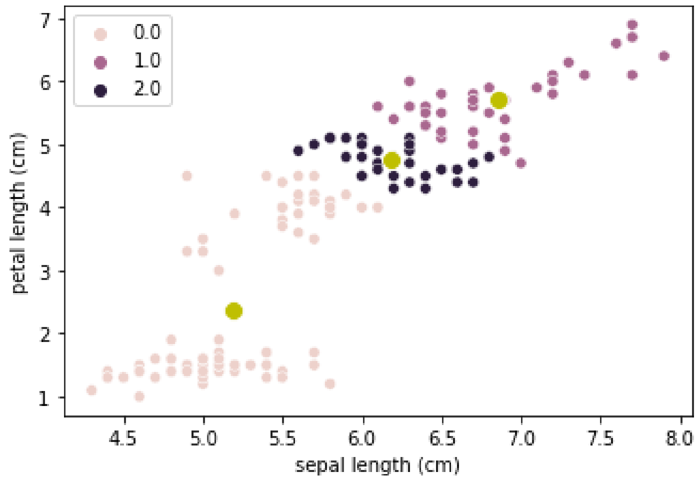


경희대학교
KYUNG HEE UNIVERSITY

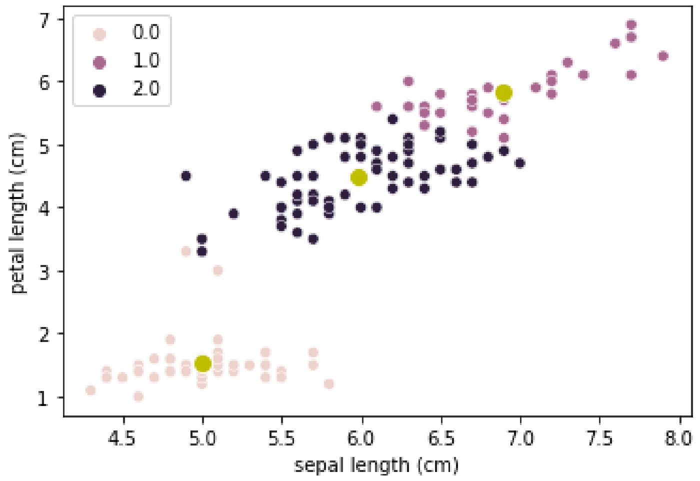
Visualization

according to iteration (using sepal length and petal length)

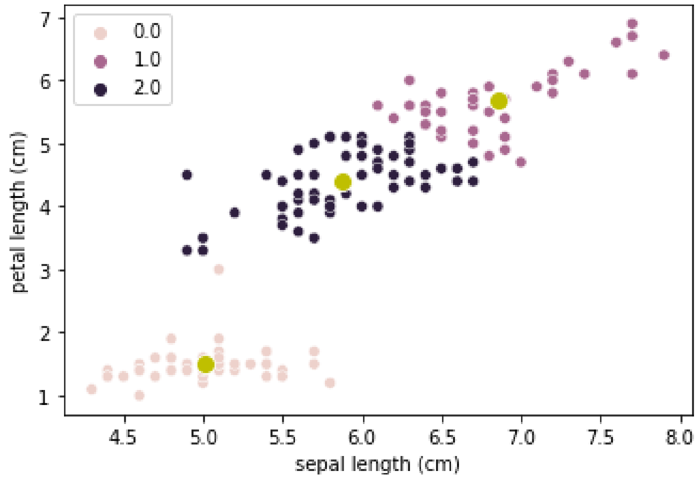
1st iteration



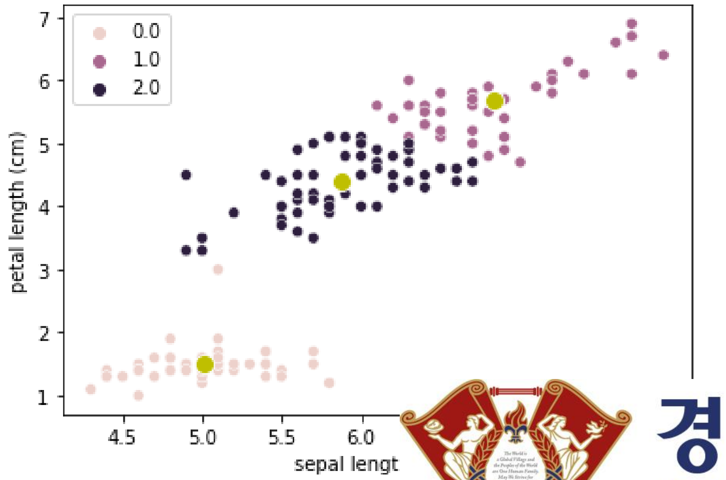
2nd iteration



3rd iteration



4th iteration



경희대학교
KYUNG HEE UNIVERSITY

Within-cluster sum of squares, WCSS

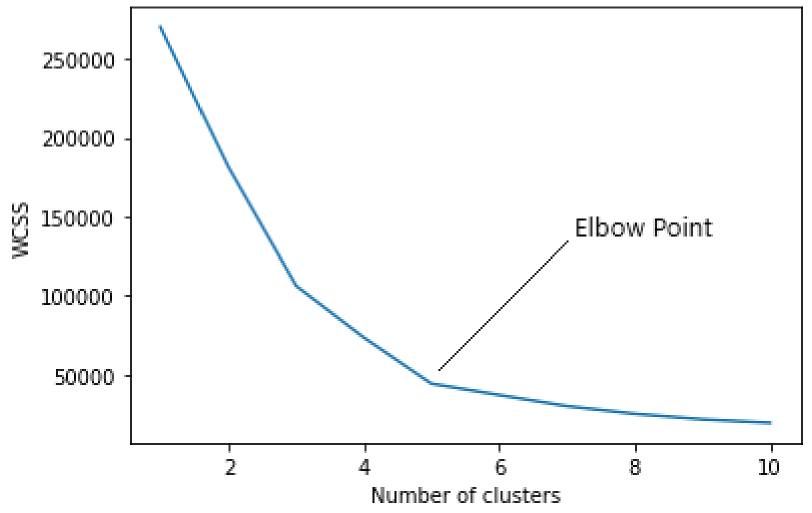
- The good or bad of clustering is evaluated quantitatively by calculating the WCSS.
- The smaller the distance between the center of the group and the data point belonging to the group, the smaller the WCSS value.

$$WCSS = \sum_{i \in n} (X_i - Y_i)^2$$

경희대학교
KYUNG HEE UNIVERSITY

WCSS elbow method

- The number of clusters can be determined using the WCSS elbow-method.
- Determine the number based on the point on the graph that looks like an elbow.



Over-fitting in unsupervised learning

- If there are too many clusters for a particular dataset, a generalized model is not created.
- For example, there are three actual types of iris varieties, but setting the cluster to five creates an unsuitable model.

Unsupervised Learning

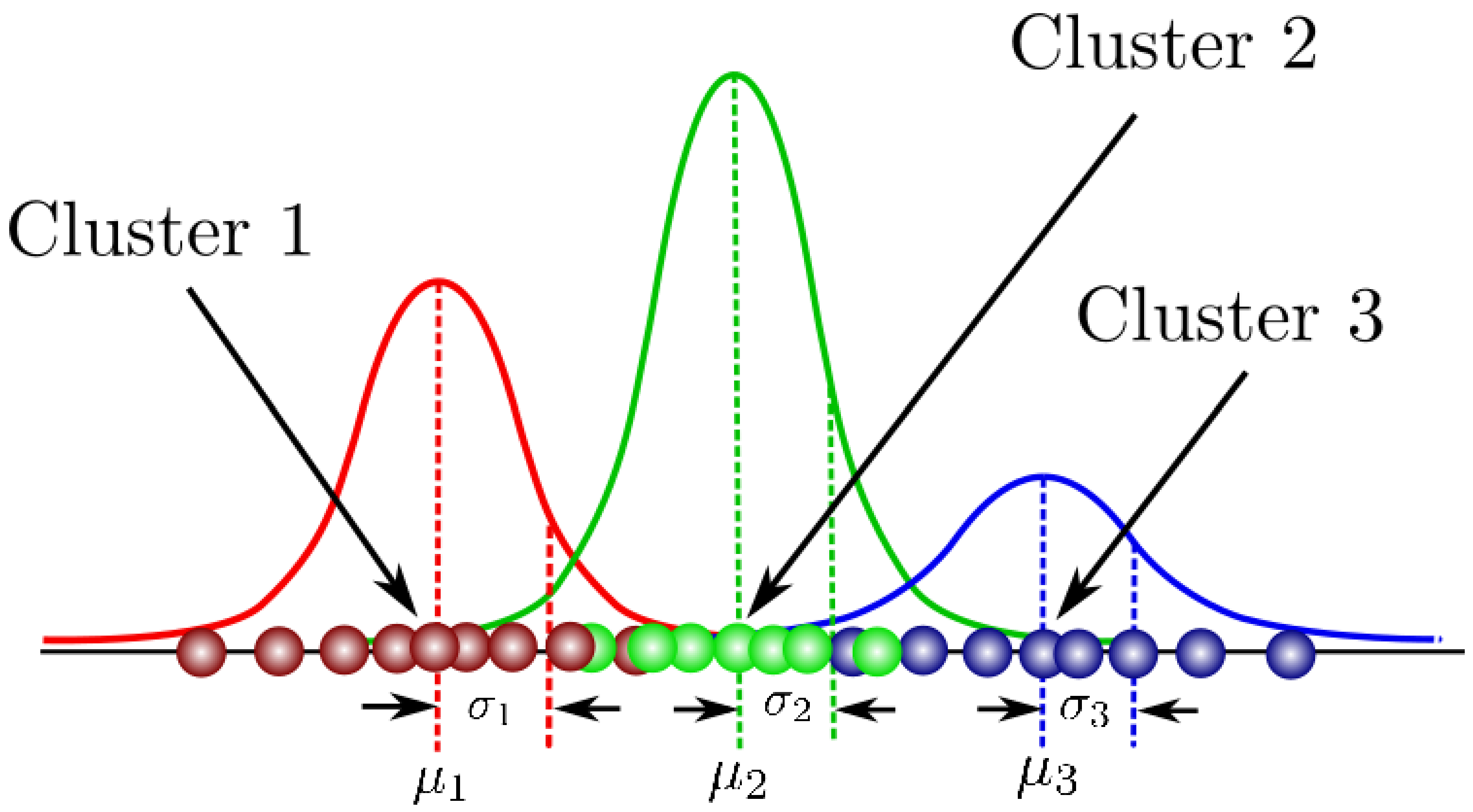
Gaussian mixture model

경희대학교 컴퓨터공학과
2019102191 신주영

Fundamental Concept

- It is used in clustering
- The data distribution is represented using the mean and variance of the data.
- The Gaussian distribution is mixed to represent complex data consisting of several groups.

Fundamental Concept



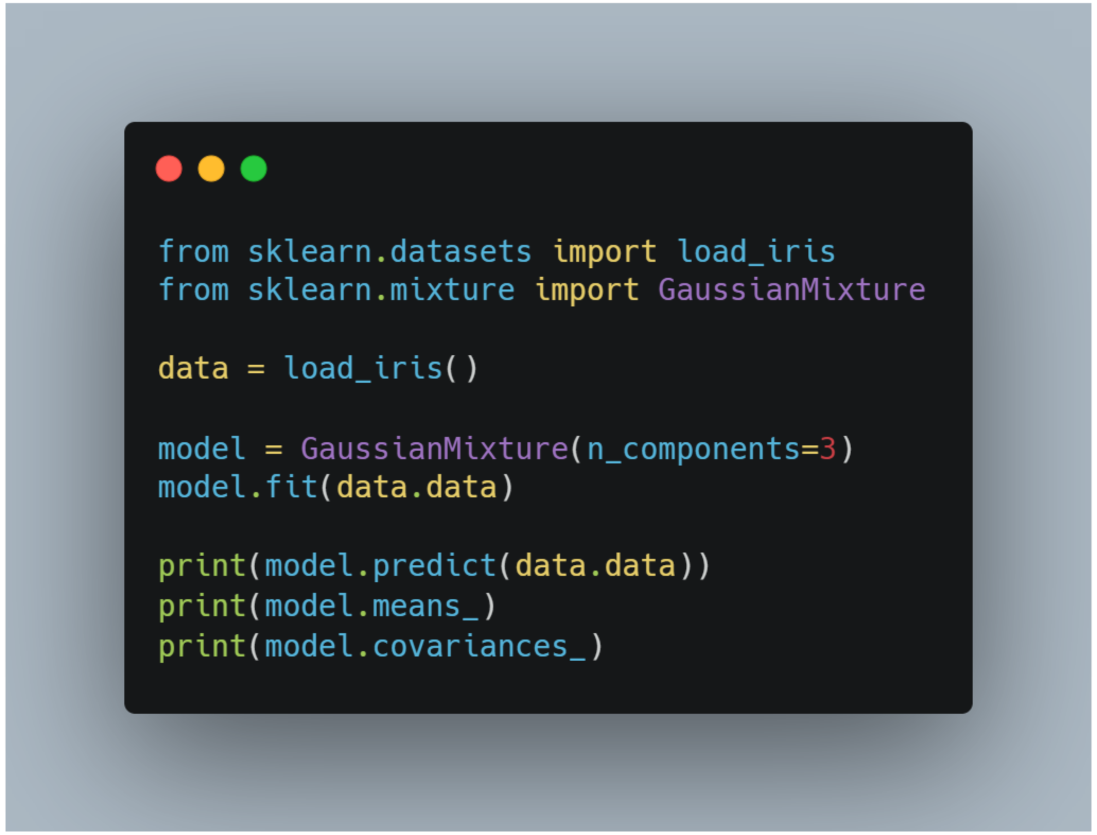
경희대학교
KYUNG HEE UNIVERSITY

Algorithm

- The Gaussian mixture model calculates the mean and variance for each Gaussian distribution at the data point.
  1. Initialize the mean and variance of each Gaussian distribution
  2. Calculate the weight of each data point from group to group
  3. Recompute parameters with weights obtained in course 2
  4. Repeat Steps 2 and 3 until each change in the average updated in Course 3 is sufficiently small

Sample Code

- Import libraries
- Make Gaussian mixture model instance
- Train data
- Print predicted data
- Print means
- Print covariances



```python
from sklearn.datasets import load_iris
from sklearn.mixture import GaussianMixture

data = load_iris()

model = GaussianMixture(n_components=3)
model.fit(data.data)

print(model.predict(data.data))
print(model.means_)
print(model.covariances_)
```

경희대학교
KYUNG HEE UNIVERSITY

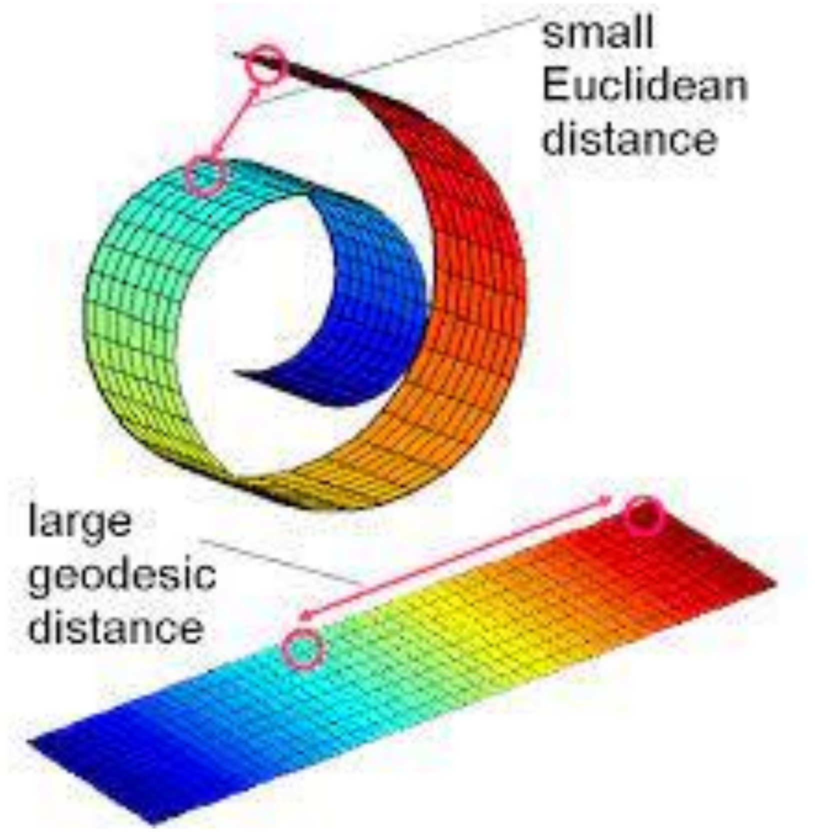
Unsupervised Learning

Local Linear Embedding, LLE

경희대학교 컴퓨터공학과
2019102191 신주영

A Dictionary Concept

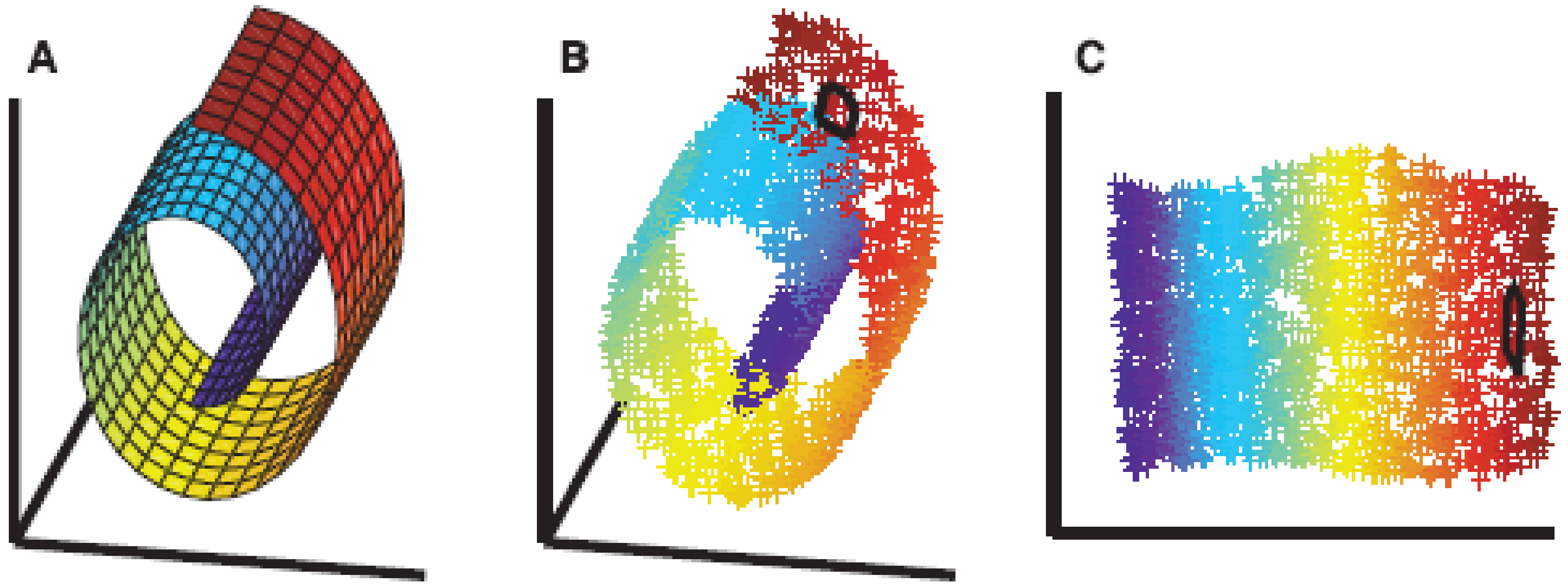
- manifold: The low-dimensional manifold is a low-dimensional shape bent or twisted in a high-dimensional space.



Fundamental Concept

- LLE is an algorithm that represents a curved or twisted structure in a high-dimensional space as a simple structure in a low-dimensional space.
- LLE is one of the manifold learnings that dimensionally reduce data of nonlinear structures.
- LLE takes out the 2D structure hidden in the 3D space and represents it as 2D data.

Fundamental Concept



경희대학교
KYUNG HEE UNIVERSITY

Fundamental Concept

- Three steps in LLE
  1. Construct kNN graph
  2. Calculate the reconstruction of weights for reconstructing every point by its neighbors
  3. Use the obtained weights to embed the points in the low dimensional subspace

Algorithm

- Training data: $\{x_i \in \mathbb{R}^d\}_{i=1}^{n}, X = [x_1, \dots, x_n] \in \mathbb{R}^{d \times n}$
- j–th neighbor of x_i : $x_{ij} \in \mathbb{R}^d, X_i = [x_{i1}, \dots, x_{ik}] \in \mathbb{R}^{d \times k}$
- Test dataset: $\{x_i^{(t)} \in \mathbb{R}^d\}_{i=1}^{n_t}, X^{(t)} = [x_1^{(t)}, \dots, x_{n_t}^{(t)}] \in \mathbb{R}^{d \times n_t}$
- Training neighbors of x_i: $X^{(t)}$: $X_i^{(t)} := [x_{i1}^{(t)}, \dots, x_{ik}^{(t)}] \in \mathbb{R}^{d \times k}$ x_i

경희대학교
KYUNG HEE UNIVERSITY

Algorithm

LLE step 1

- Linear regional relationship modeling.
- Fix the sample and find the optimal weight.

$$\hat{W} = argmin \sum_{i=1}^{m} \left(x^{(i)} - \sum_{j=1}^{m} w_{i,j} X^{(j)}\right)^{2}$$

[조건] $\begin{cases} w_{i,j=0} & X^{(j)}\text{가 } X^{(i)}\text{의 최근접 이웃 } k\text{개 중 하나가 아닐 때} \\ \sum_{j=1}^{m} w_{i,j} = 1 & i=1,2,3,\ldots,m\text{일 때} \end{cases}$

경희대학교
KYUNG HEE UNIVERSITY

Algorithm

LLE step 2

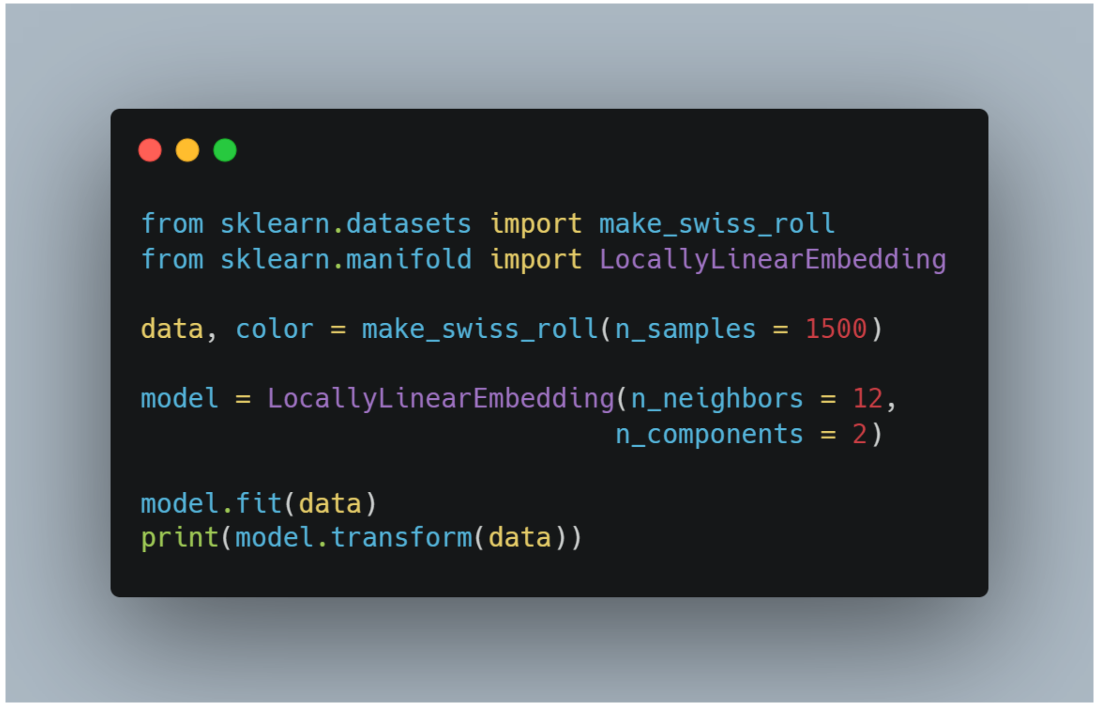
- Reduce the dimension of preserving relationships
- Fix the weights and find the optimal location of the sample image in the low-dimensional space.

$$Z = argmin \sum_{i=1}^{m} (Z^{(i)} - \sum_{j=1}^{m} \widehat{w}_{i,j} Z^{(j)})^{2}$$

경희대학교
KYUNG HEE UNIVERSITY

Sample Code

- Import libraries
- Make dataset
- Set n_neighbors and dimension
- Train data
- Print result



```python
from sklearn.datasets import make_swiss_roll
from sklearn.manifold import LocallyLinearEmbedding

data, color = make_swiss_roll(n_samples = 1500)

model = LocallyLinearEmbedding(n_neighbors = 12,
                               n_components = 2)

model.fit(data)
print(model.transform(data))
```

Unsupervised Learning

t-distributed stochastic neighbor embedding, t-SNE

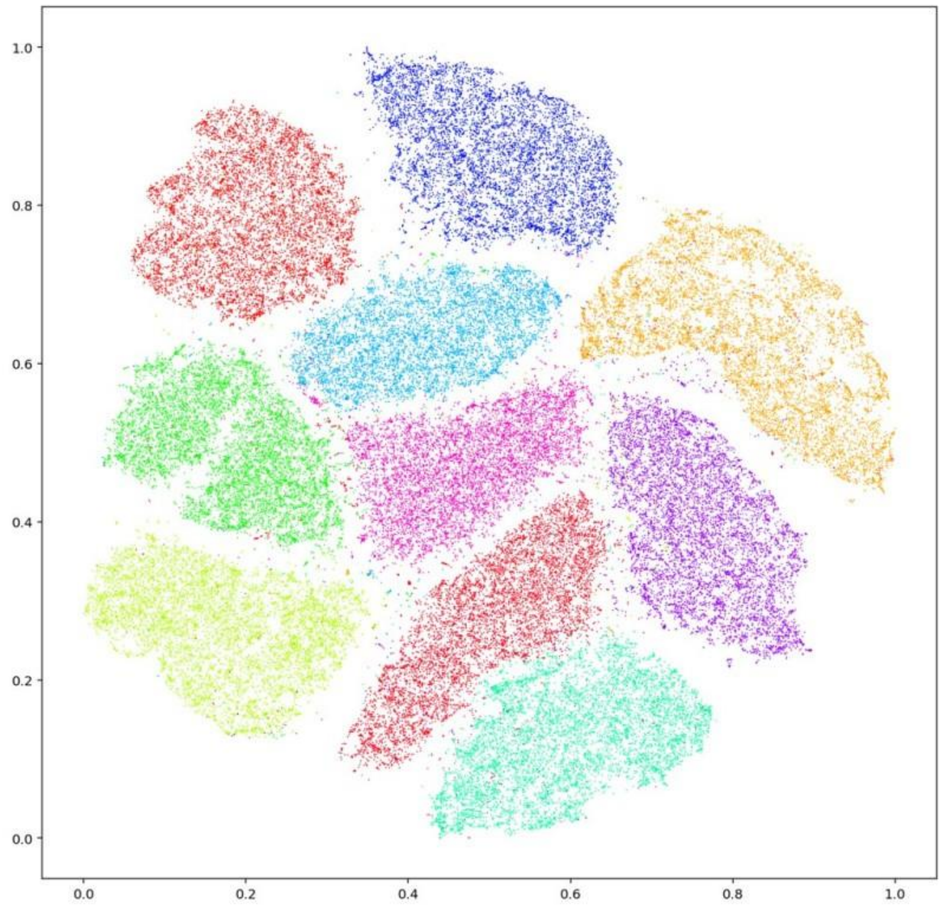
경희대학교 컴퓨터공학과
2019102191 신주영

경희대학교
KYUNG HEE UNIVERSITY

Fundamental Concept

- It is a method of dimensionally reducing complex data in high dimensions to two dimensions.
- t-SNE is one of manifold learning and aims to visualize complex data.

Fundamental Concept



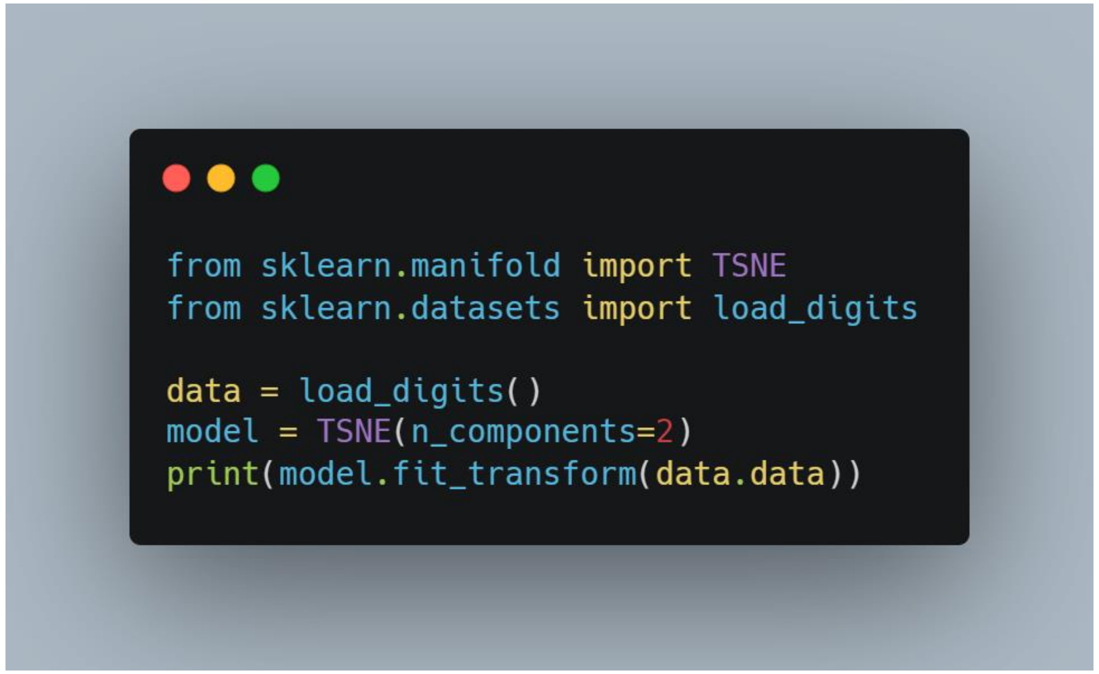
Algorithm

- t-SNE steps
  - The similarity of x_i and x_j for all i and j pairs is expressed as similarity using Gaussian distribution.
  - We randomly place the same number of points y_i as x_i in a low-dimensional space, and show the similarities of y_i and y_j for all i and j pairs using the t distribution.
  - If possible, the data point y_i is updated so that the similarity distribution defined in 1 and 2 is the same.
  - Repeat step 3 until the convergence condition.

경희대학교
KYUNG HEE UNIVERSITY

Sample Code

- Import libraries
- Load data
- Make t-SNE model
- Print result

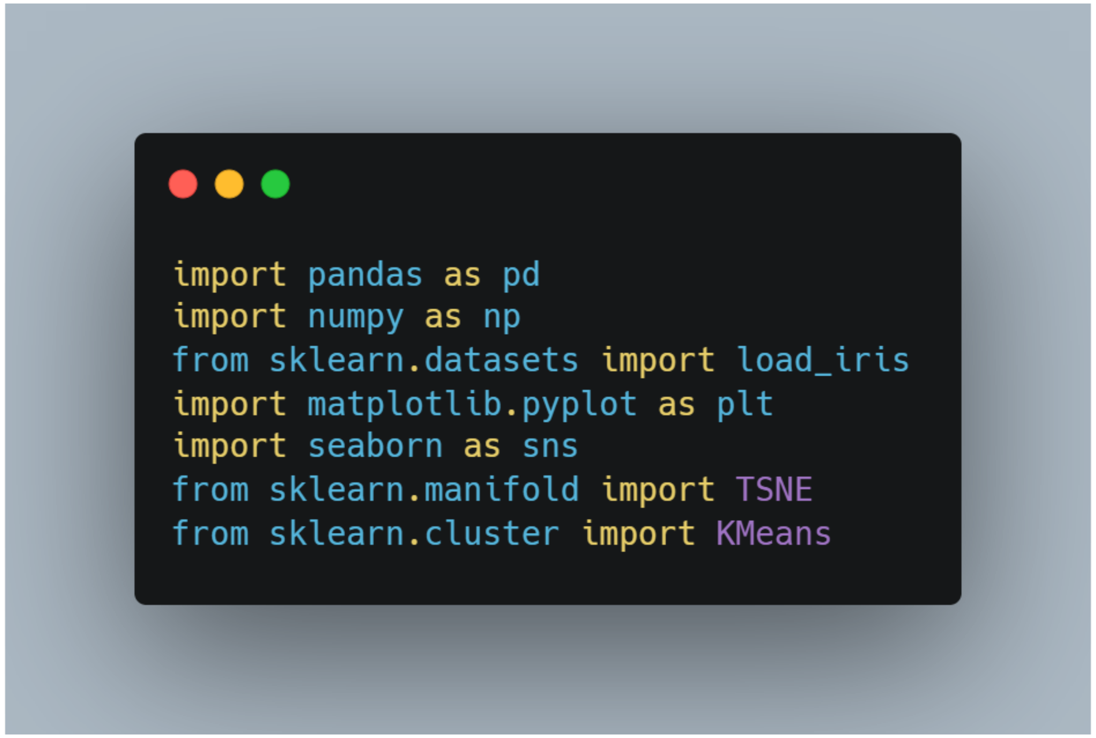


```python
from sklearn.manifold import TSNE
from sklearn.datasets import load_digits

data = load_digits()
model = TSNE(n_components=2)
print(model.fit_transform(data.data))
```

Application of t-SNE to K-Means

using iris dataset

- Import libraries
- Pandas
- Numpy
- Sklearn
- Matplotlib
- Seaborn

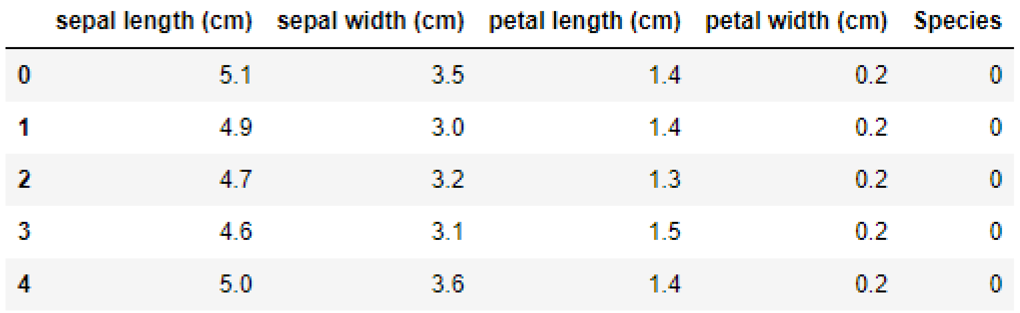


```python
import pandas as pd
import numpy as np
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.manifold import TSNE
from sklearn.cluster import KMeans
```

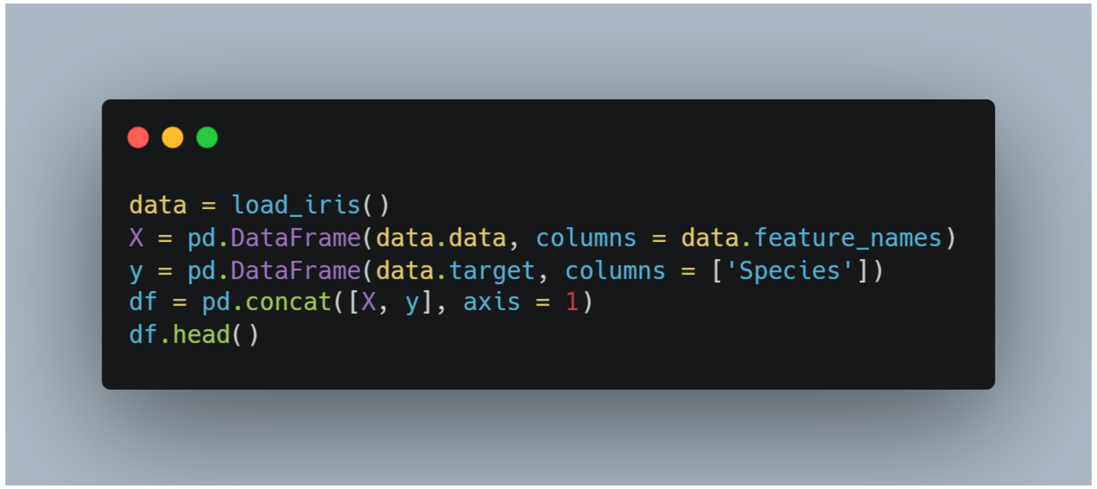
경희대학교
KYUNG HEE UNIVERSITY

Application of t-SNE to K-Means
using iris dataset

- Make dataset



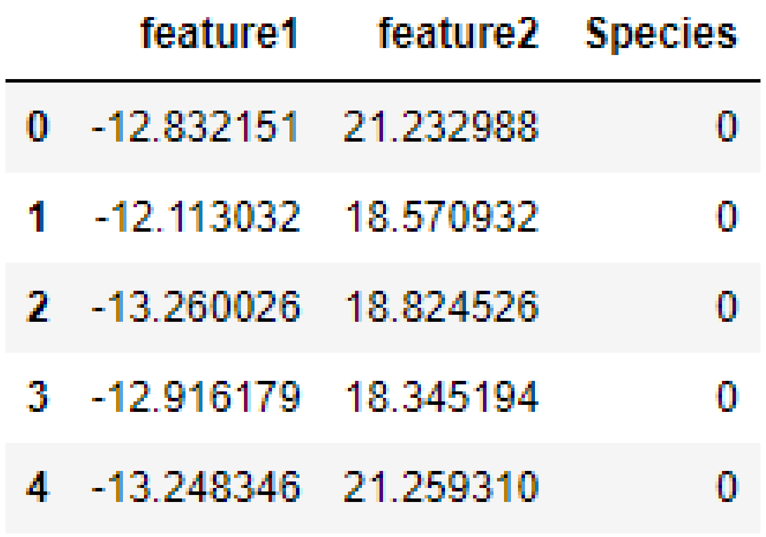
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0



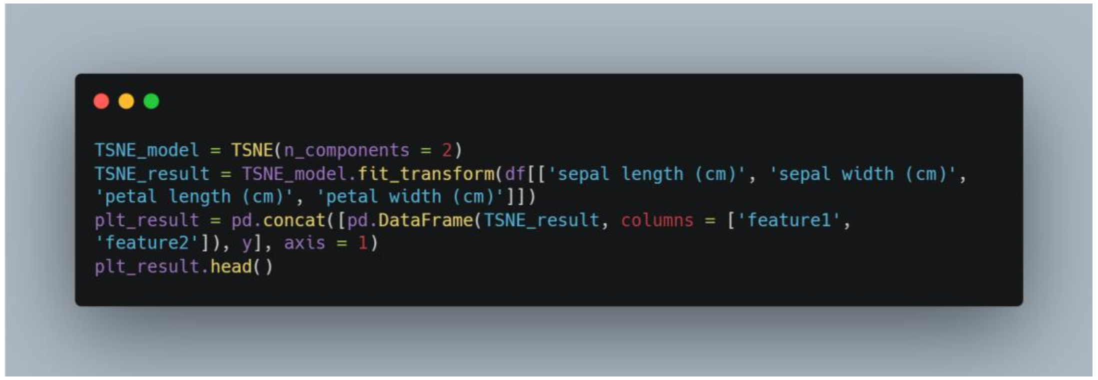
```
data = load_iris()
X = pd.DataFrame(data.data, columns = data.feature_names)
y = pd.DataFrame(data.target, columns = ['Species'])
df = pd.concat([X, y], axis = 1)
df.head()
```

Application of t-SNE to K-Means
using iris dataset

- t-SNE result



	feature1	feature2	Species
0	-12.832151	21.232988	0
1	-12.113032	18.570932	0
2	-13.260026	18.824526	0
3	-12.916179	18.345194	0
4	-13.248346	21.259310	0

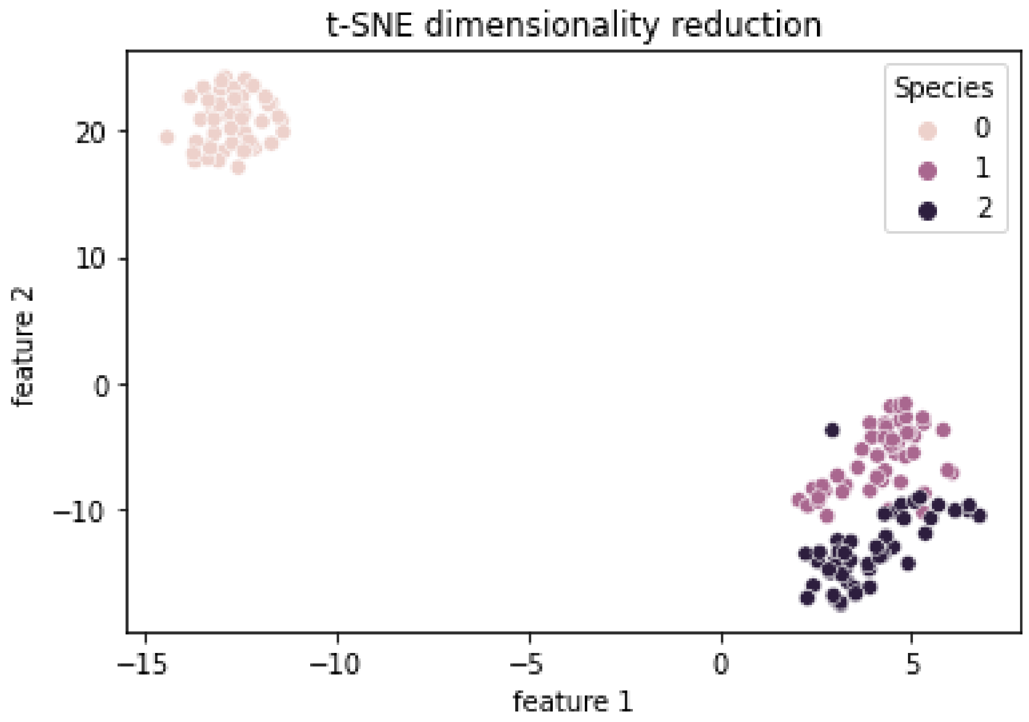


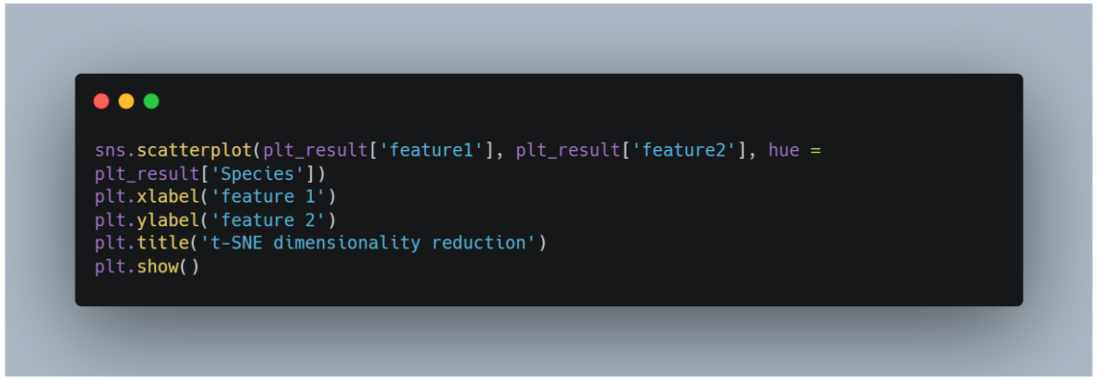
```
TSNE_model = TSNE(n_components = 2)
TSNE_result = TSNE_model.fit_transform(df[['sepal length (cm)', 'sepal width (cm)',
'petal length (cm)', 'petal width (cm)']])
plt_result = pd.concat([pd.DataFrame(TSNE_result, columns = ['feature1',
'feature2']), y], axis = 1)
plt_result.head()
```

Application of t-SNE to K-Means

using iris dataset

- Actual iris data distribution



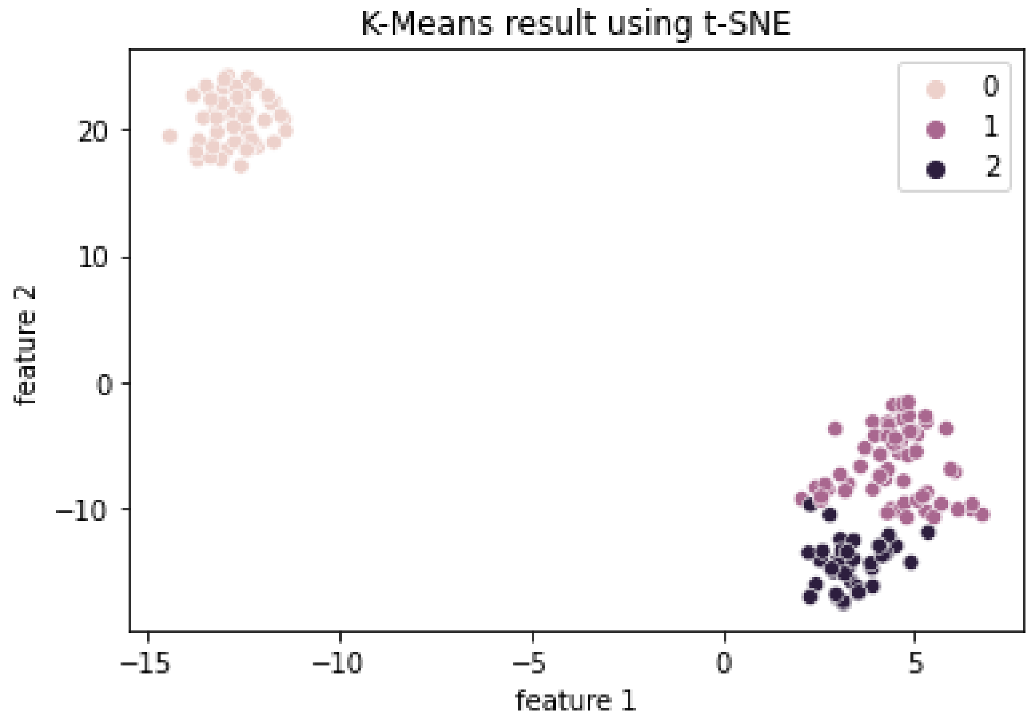


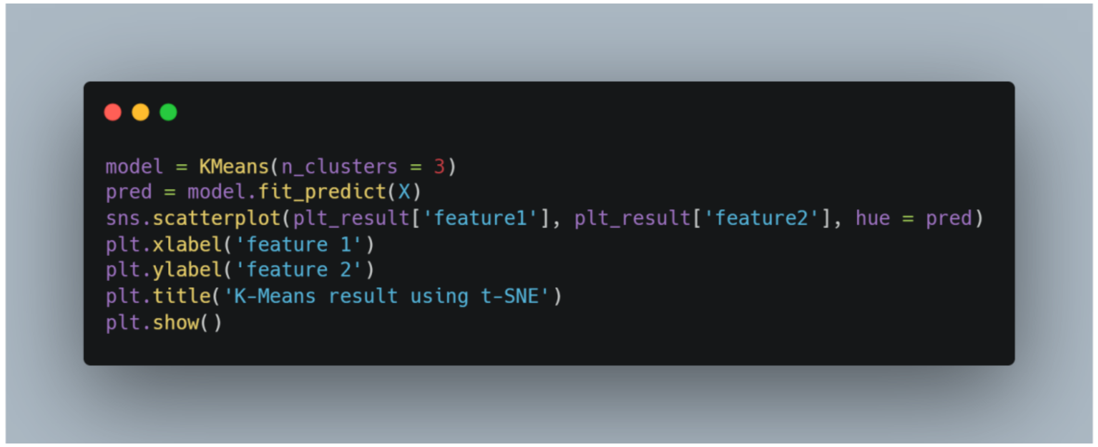
```
sns.scatterplot(plt_result['feature1'], plt_result['feature2'], hue =
plt_result['Species'])
plt.xlabel('feature 1')
plt.ylabel('feature 2')
plt.title('t-SNE dimensionality reduction')
plt.show()
```

Application of t-SNE to K-Means

using iris dataset

- Predicted iris data distribution





```
model = KMeans(n_clusters = 3)
pred = model.fit_predict(X)
sns.scatterplot(plt_result['feature1'], plt_result['feature2'], hue = pred)
plt.xlabel('feature 1')
plt.ylabel('feature 2')
plt.title('K-Means result using t-SNE')
plt.show()
```