Machine learning 04

Byung Chang Chung

Gyeongsang National University

bcchung@gnu.ac.kr

Contents

- Dimensionality reduction

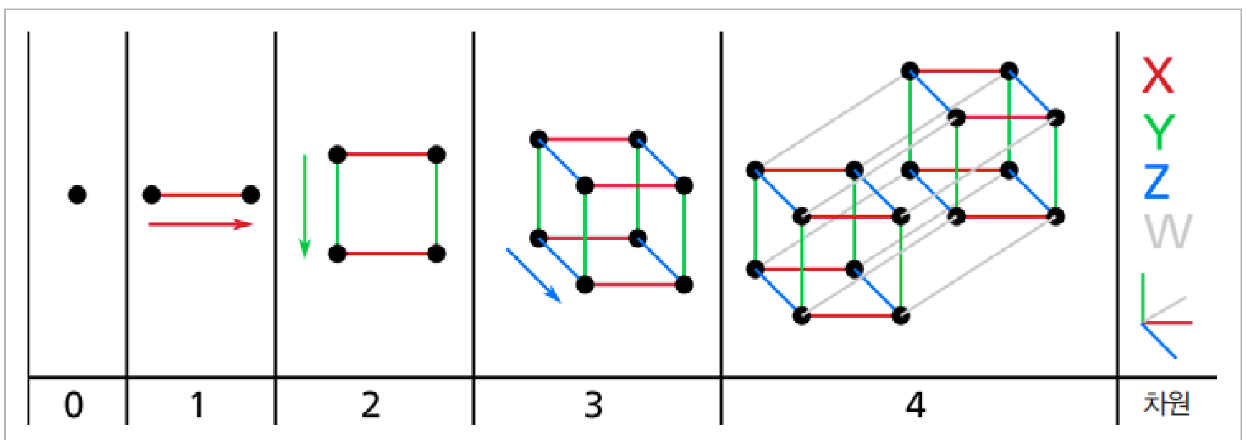
Curse of dimensionality

- Lots of samples
  - training sample has thousands or even millions of properties
  - not only slow down training but also make it difficult to find a good solution

Gyeongsang National University 경상국립대학교

Curse of dimensionality

- We cannot intuitively imagine high-dimension
  - if you select any two points
    - in 3D, the average distance is approximately 0.66
    - in 1,000,000D, the average distance is approximately 428.25



Curse of dimensionality

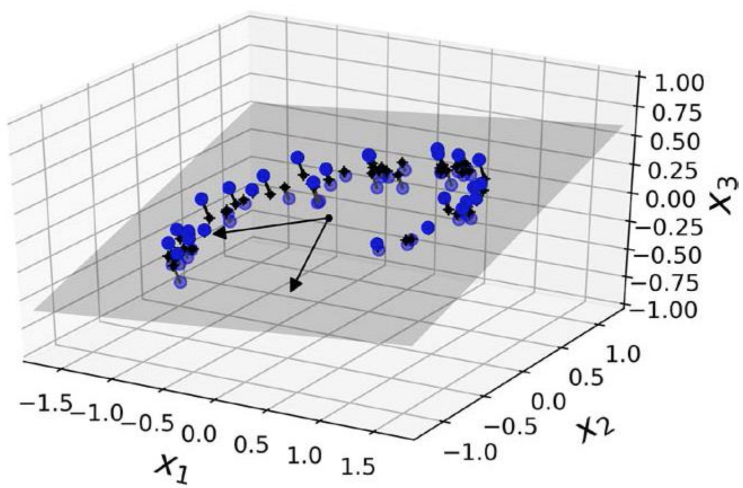
- How to approach?
  - theoretically, increasing the size of the training set until the training sample becomes sufficiently dense
  - the number of training samples required to reach a certain density increases exponentially as the number of dimensions increases

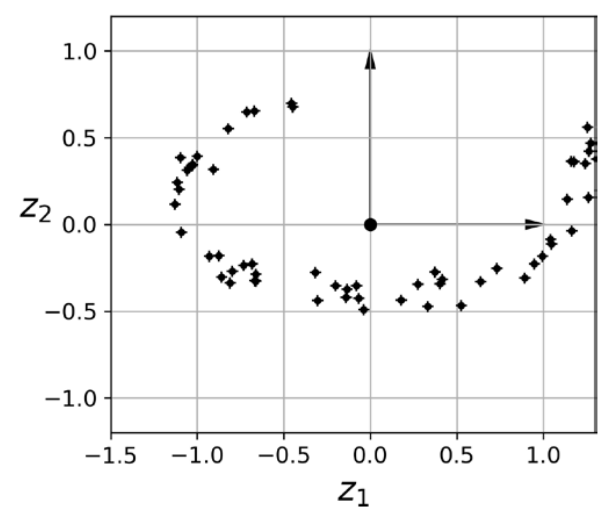
Projection

- Definition
  - in mathematics, a projection is a mapping of a set (or other mathematical structure) into a subset (or sub-structure), which is equal to its square for mapping composition

Projection

- Examples





Gyeongsang National University 경상국립대학교

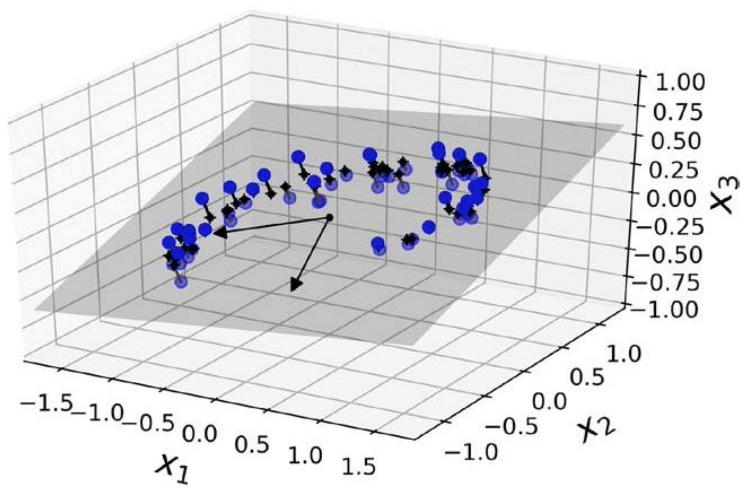
Projection

- Properties of data
  - training samples are not distributed evenly across all dimensions
  - many features remain almost unchanged
  - other properties are strongly associated with each other

Gyeongsang National University 경상국립대학교

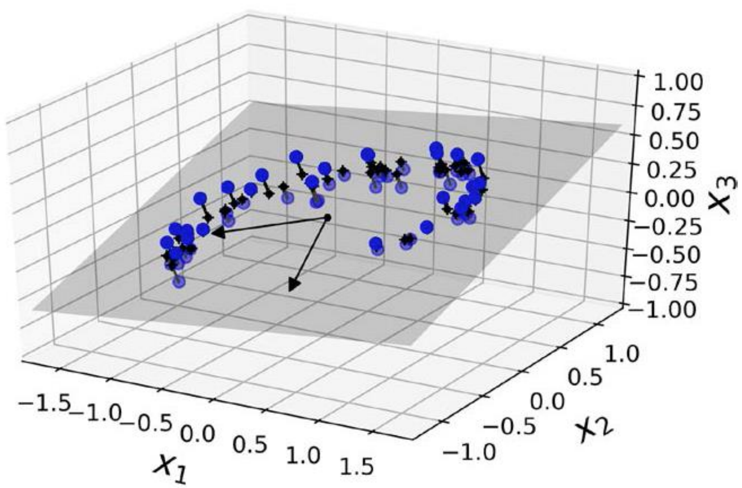
Projection

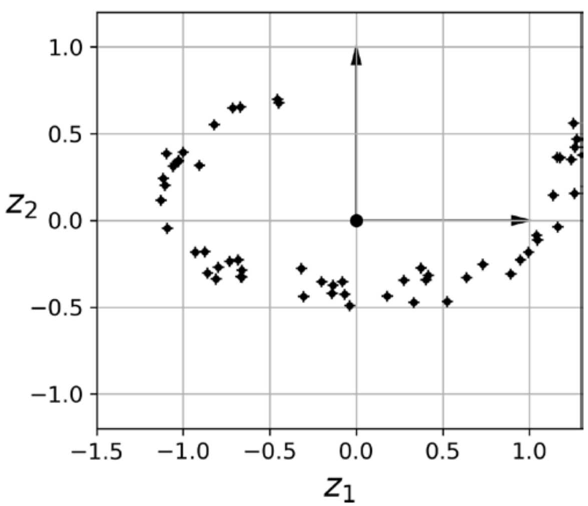
- Properties of data
  - resulting in all training samples lying (or close) in a low-dimensional subspace within a high-dimensional space



Projection

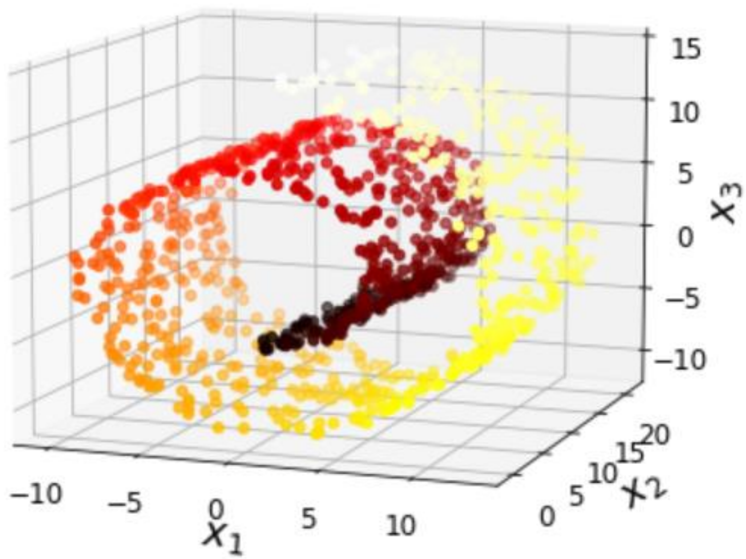
- Properties of data
  - projecting all training samples perpendicular to this subspace yields a below dataset





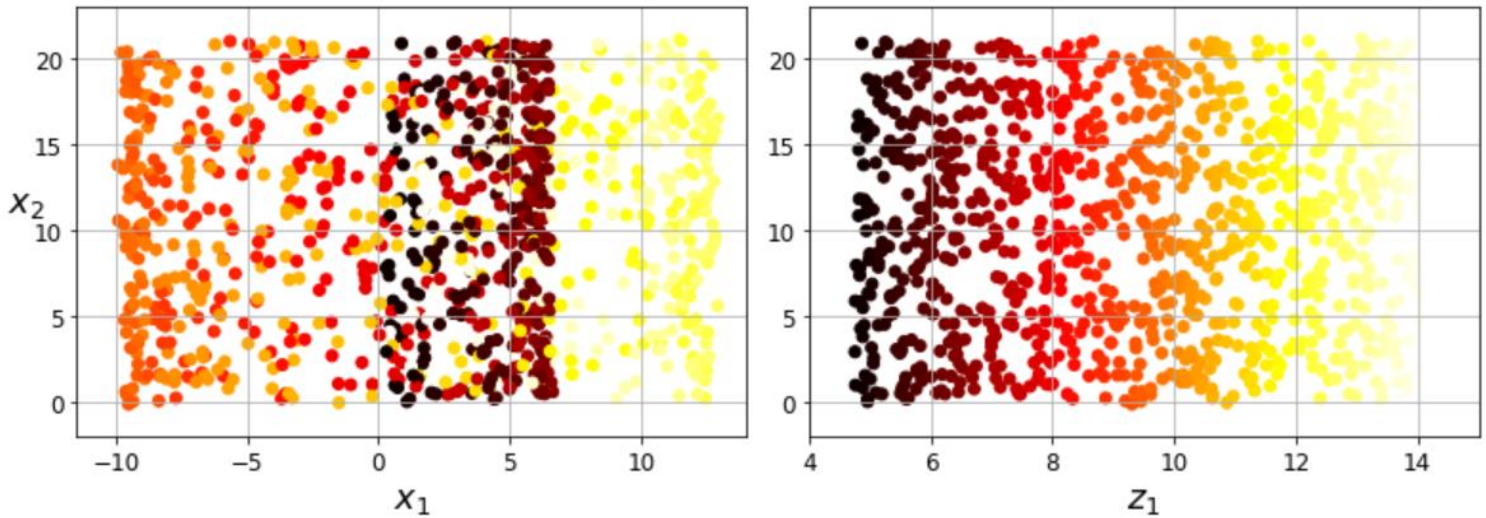
Projection

- Exceptions
  - Swiss roll dataset



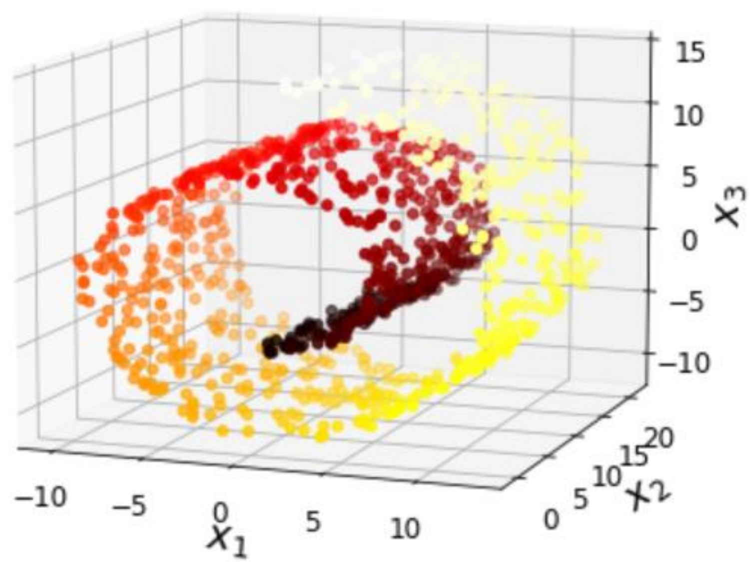
Projection

- Exceptions
  - projected Swiss roll dataset
    - which one is effective projection?



Manifold learning

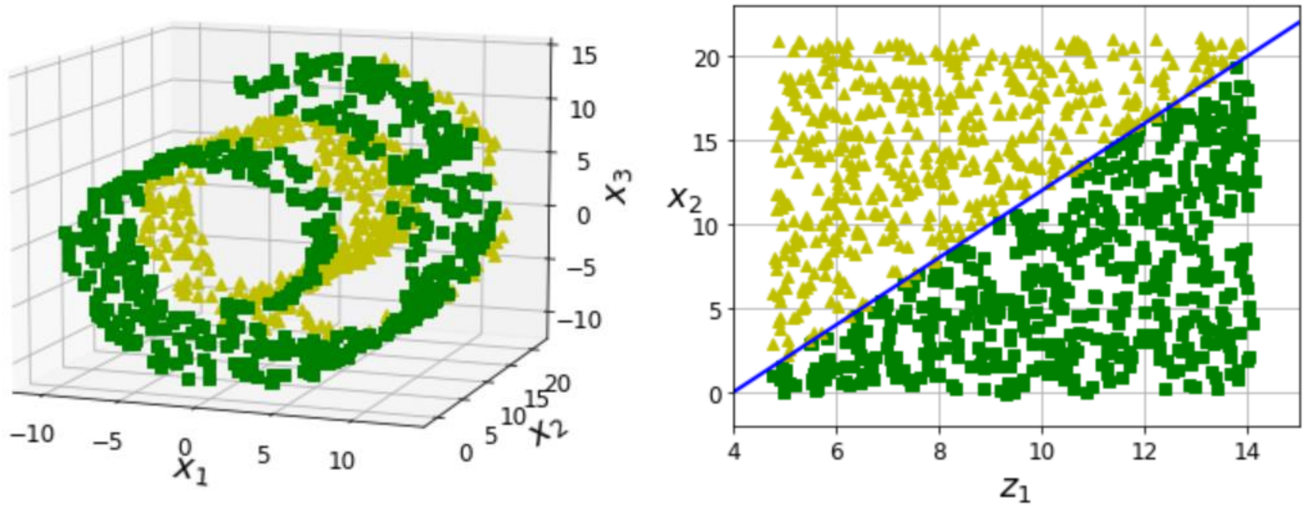
- Definition of manifold
  - in mathematics, a manifold is a topological space that locally resembles Euclidean space near each point



Gyeongsang National University 경상국립대학교

Manifold learning

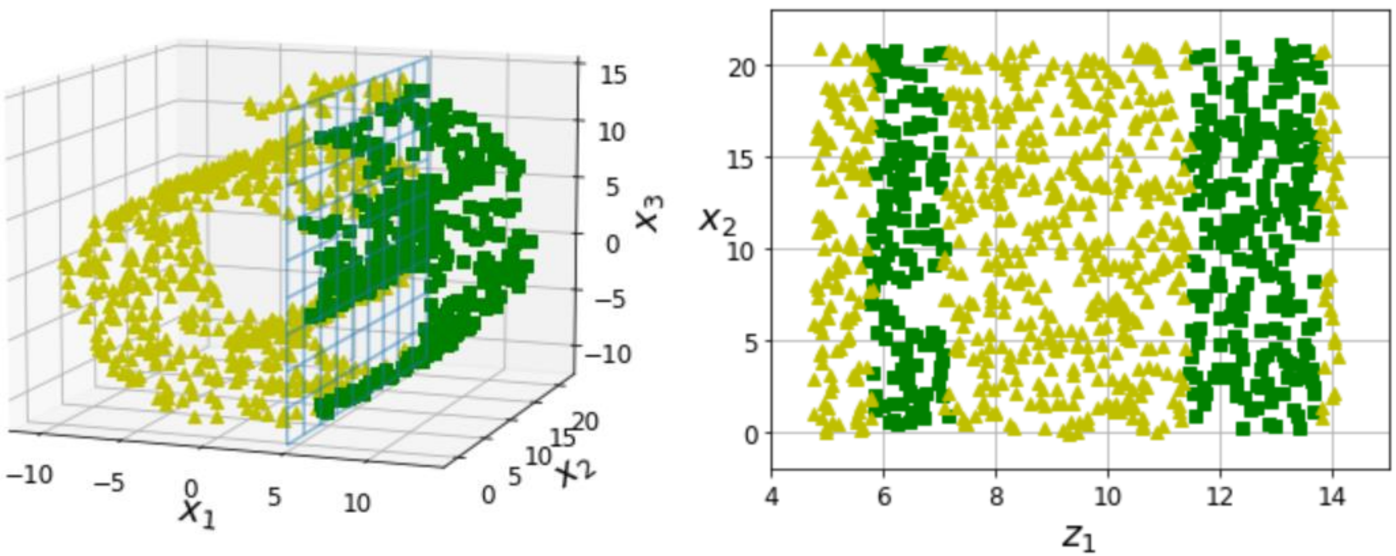
- Assumption
  - Based on the hypothesis that the high-dimensional dataset lies closer to the low-dimensional manifold



Gyeongsang National University 경상국립대학교

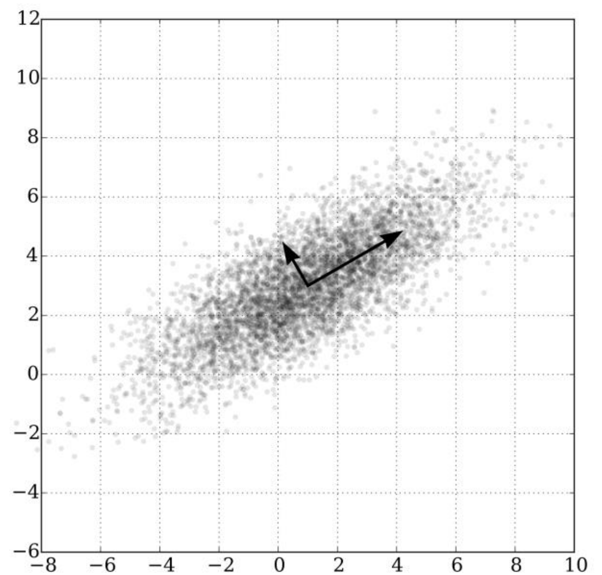
Manifold learning

- Assumption
  - is not always correct



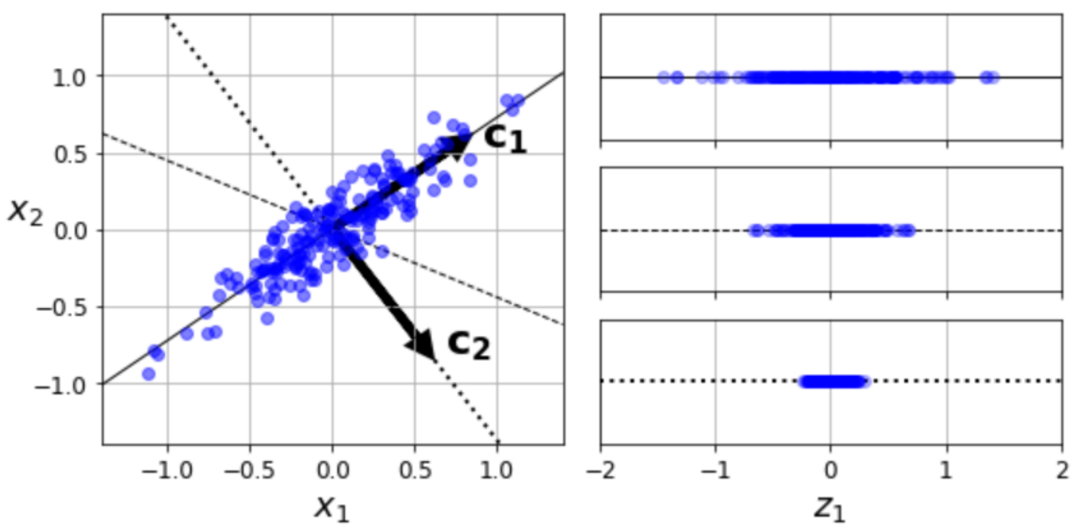
PCA

- Principal component analysis (주성분분석)
  - find hyperplane that closer to the given dataset
  - then project on the optimal hyperplane



PCA process

- How to select optimal hyperplane?
  - in terms of variance
    - dotted line
    - real line maximizes variance



Gyeongsang National University 경상국립대학교

PCA process

- Principal component
  - find axis which results largest variance
  - singular value decomposition in linear algebra

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

A: $m \times n$ rectangular matrix

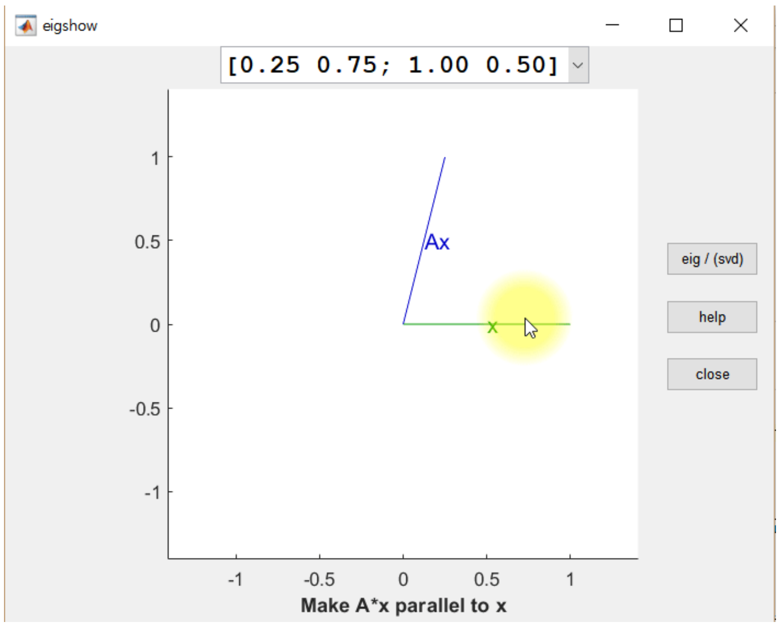
U: $m \times m$ orthogonal matrix

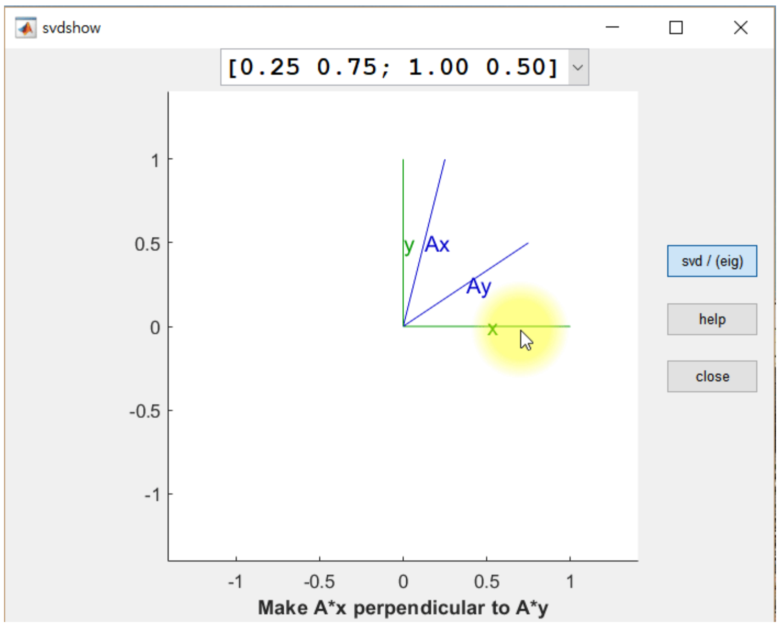
Σ: $m \times n$ diagonal matrix

V: $n \times n$ orthogonal matrix

PCA process

- Singular value decomposition





Gyeongsang National University
경상국립대학교

https://angeloyeo.github.io/2019/08/01/SVD.html

PCA process

- Projection to d-dimension

$$\mathbf{X}_{d\text{-proj}} = \mathbf{X}\mathbf{W}_d$$

  - $\boldsymbol{W}_d$ refers to the first d-rows of matrix $\boldsymbol{V}$

GNU
Gyeongsang
National
University
경상국립대학교

PCA implementation

- Implementation in scikit-learn

```python
from sklearn.decomposition import PCA

pca = PCA(n_components = 2)
X2D = pca.fit_transform(X)
```

  - reduce dimension to 2 (n_components = 2)

Gyeongsang National University 경상국립대학교

PCA implementation

- Finding appropriate dimension

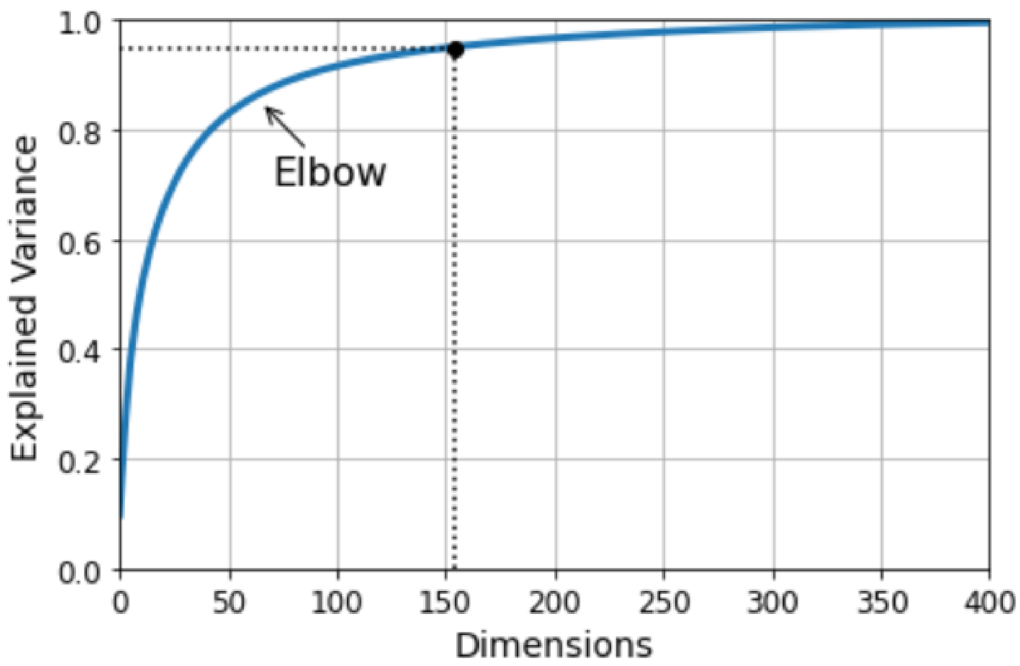
```
pca = PCA()
pca.fit(X_train)
cumsum = np.cumsum(pca.explained_variance_ratio_)
d = np.argmax(cumsum >= 0.95) + 1
```

- finding dimension which maintain the variance of dataset greater than 0.95

Gyeongsang National University 경상국립대학교

PCA implementation

- Finding appropriate dimension – visualization



PCA application

- PCA for extraction
  - reducing dimension => reducing amount of information

$$\mathbf{X}_{\text{recovered}} = \mathbf{X}_{d\text{-proj}} \mathbf{W}_d^T$$

PCA application

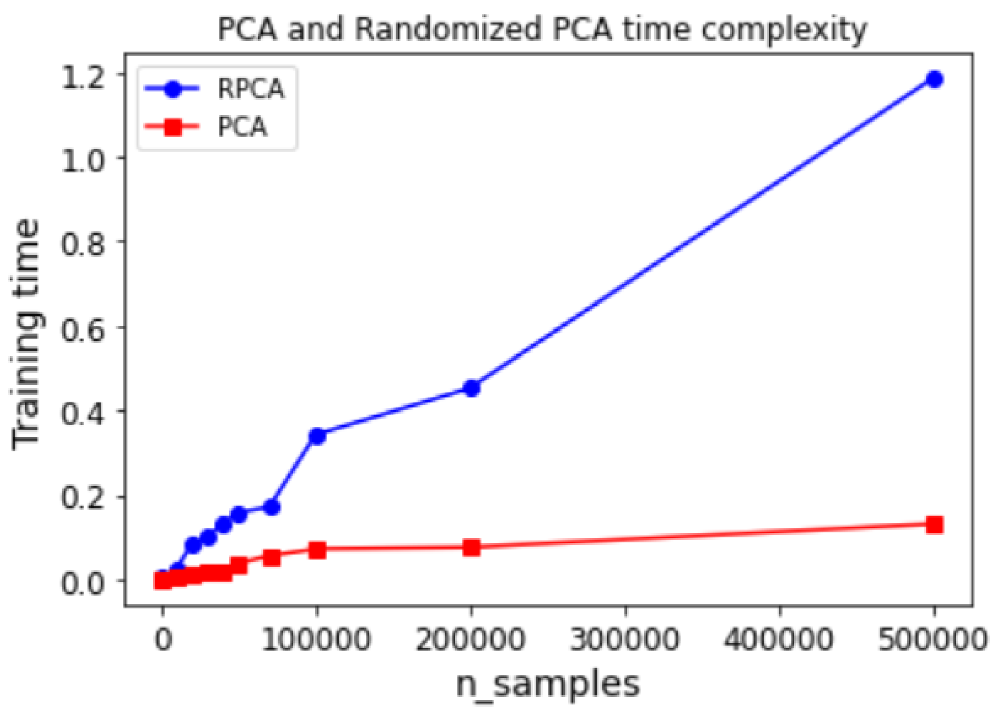
- Random PCA
  - finding randomized SVD can find principal component faster
  - SVD complexity $O(m \times n^2) + O(n^3)$
  - randomized SVD complexity $O(m \times d^2) + O(d^3)$

```
rnd_pca = PCA(n_components=154, svd_solver="randomized", random_state=42)
X_reduced = rnd_pca.fit_transform(X_train)
```

Gyeongsang National University 경상국립대학교

PCA application

- Time complexity of random PCA



Gyeongsang National University 경상국립대학교

PCA application

- Incremental PCA (IPCA)
  - how to make online?
    - divide dataset into minibatch
    - using IPCA

```
from sklearn.decomposition import IncrementalPCA

n_batches = 100
inc_pca = IncrementalPCA(n_components=154)
for X_batch in np.array_split(X_train, n_batches):
    print(".", end="") # 책에는 없음
    inc_pca.partial_fit(X_batch)

X_reduced = inc_pca.transform(X_train)
```

GNU Gyeongsang National University 경상국립대학교

Kernel PCA

- Nonlinear projection using kernel

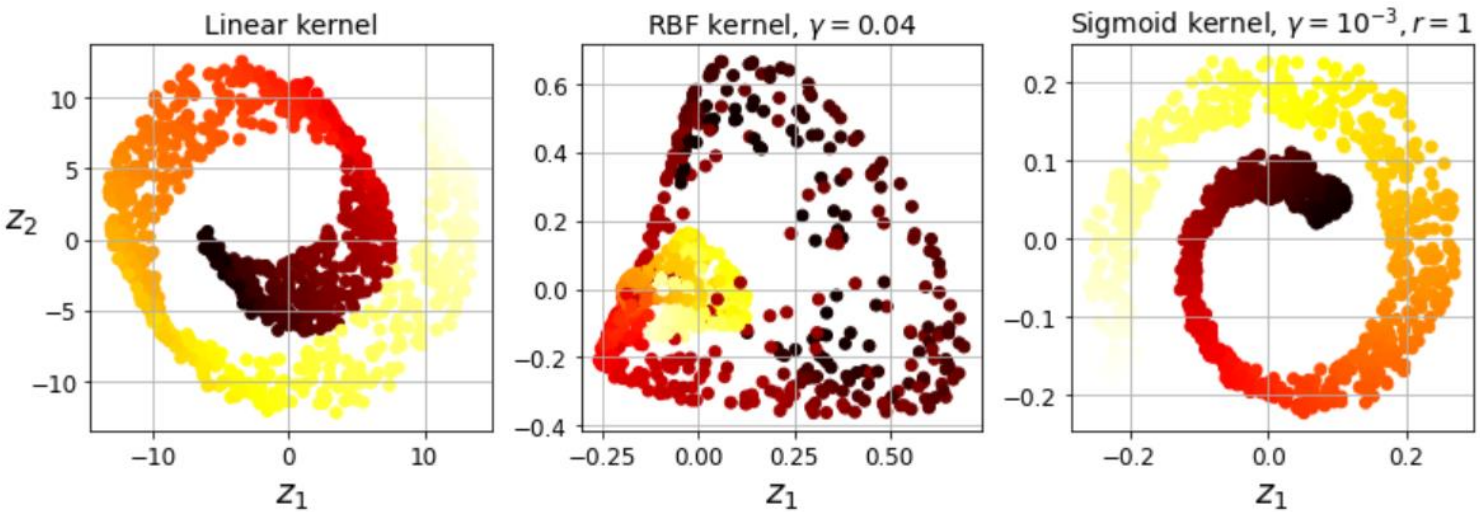
```
from sklearn.decomposition import KernelPCA

rbf_pca = KernelPCA(n_components=2, kernel="rbf", gamma=0.04)
X_reduced = rbf_pca.fit_transform(X)
```

  - kernel calculation can make non-linearity

Gyeongsang National University 경상국립대학교

Kernel PCA

- Nonlinear projection using kernel – visualization



Kernel PCA

- Finding proper kernel and hyperparameter
  - there are no clear performance metrics for selecting good kernels and hyperparameters since it is unsupervised
  - use grid search to select the appropriate kernel and hyperparameters for a imaginary regression problem

Kernel PCA

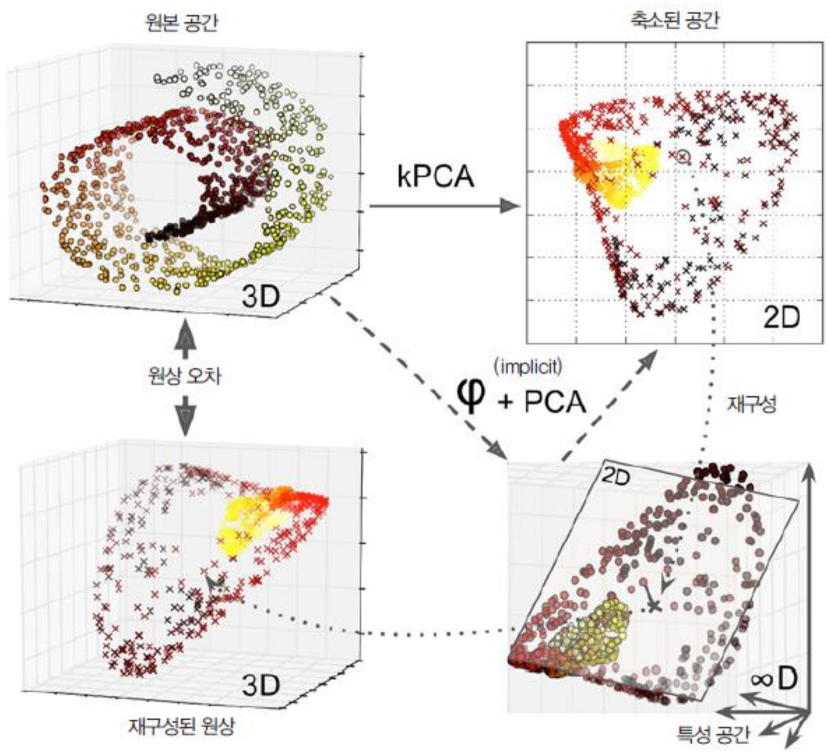
- Finding proper kernel and hyperparameter

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline

clf = Pipeline([
        ("kpca", KernelPCA(n_components=2)),
        ("log_reg", LogisticRegression(solver="lbfgs"))
    ])

param_grid = [{
        "kpca__gamma": np.linspace(0.03, 0.05, 10),
        "kpca__kernel": ["rbf", "sigmoid"]
    }]

grid_search = GridSearchCV(clf, param_grid, cv=3)
grid_search.fit(X, y)
```

GNU
Gyeongsang
National
University
경상국립대학교

Kernel PCA

- Other way
  - comparison with original data and reconstructed data



Gyeongsang National University 경상국립대학교

Locally linear embedding

- Manifold learning which is not limited to projection
  - measure how linearly each training sample relates to the closest neighbor (c.n.)
  - local relationship can be preserved
  - it works well to unfold the twisted manifold when there is not too much noise

Locally linear embedding

- Mathematical representation

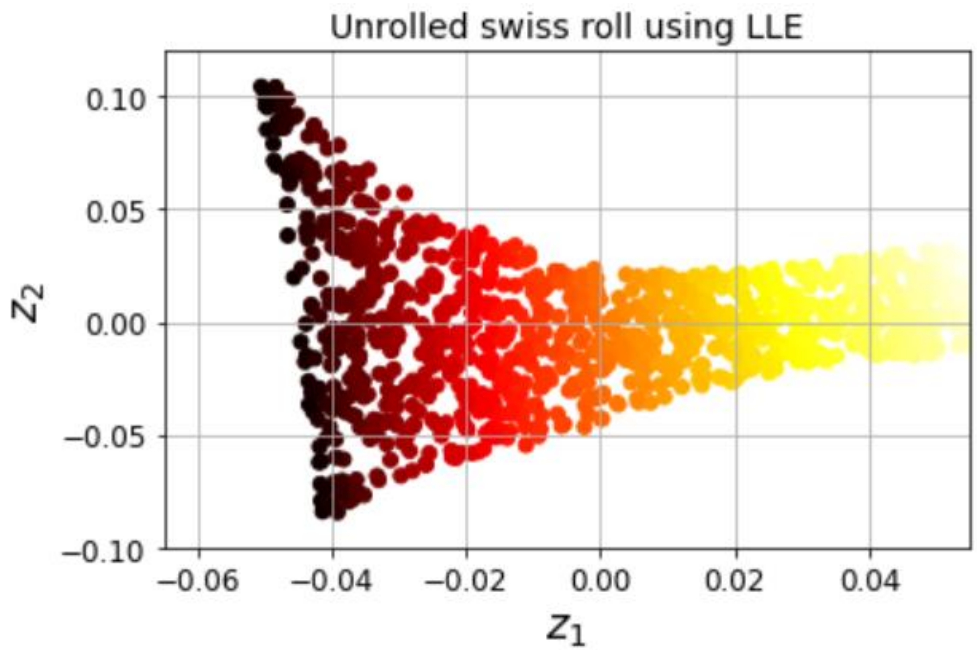
$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmin}} \sum_{i=1}^{m} \left(\mathbf{x}^{(i)} - \sum_{j=1}^{m} w_{i,j} \mathbf{x}^{(j)} \right)^2$$

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

$$\mathbf{Z} = \underset{\mathbf{Z}}{\operatorname{argmin}} \sum_{i=1}^{m} \left(\mathbf{z}^{(i)} - \sum_{j=1}^{m} \hat{w}_{i,j} \mathbf{z}^{(j)} \right)^2$$

Locally linear embedding

- LLE in scikit-learn



Gyeongsang National University 경상국립대학교

Other works with dimension

- Random projection
  - random linear projections to project data into low-dimensional space
- Multidimensional scaling (MDS)
  - reduce dimension while preserving distance between samples

Other works with dimension

- Isomap
  - make a graph by linking each sample with the nearest neighbor
  - reduce dimension while maintaining the geodesic distance

Gyeongsang National University 경상국립대학교

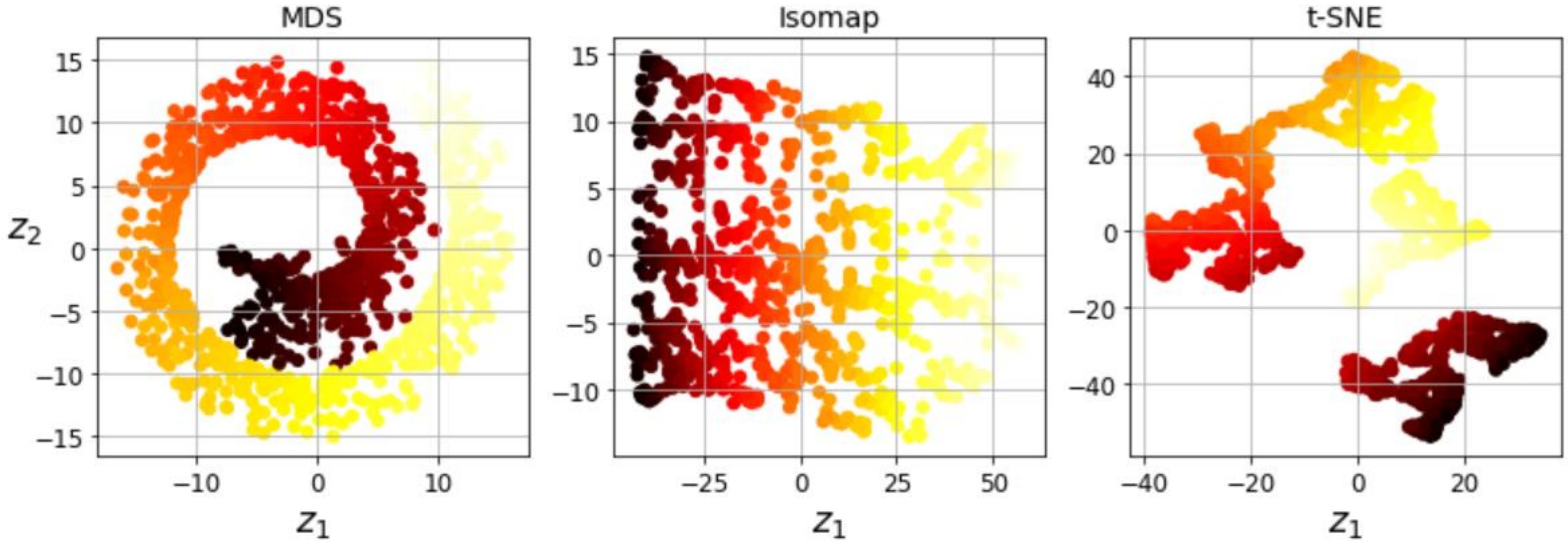
Other works with dimension

- t-distributed stochastic neighbor embedding (t-SNE)
    - reduce dimensions while keeping similar samples close and non-similar samples far away

- Linear discriminant analysis (LDA)
    - finding hyperplane that best separates the dataset
    - this hyperplane will be used in projection

Other works with dimension

- Results of dimension reduction algorithm



Feel free to question

Through e-mail & LMS

본 자료의 연습문제는 수업의 본교재인
한빛미디어, Hands on Machine Learning(2판)에서 주로 발췌함

Gyeongsang National University 경상국립대학교