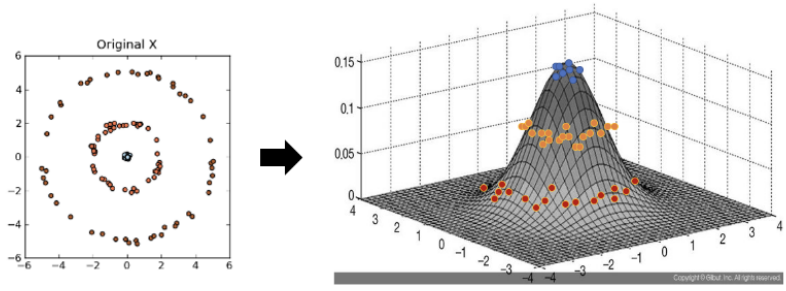
[Week10] 파머완 6장 예습과제

Dataset Decomposition Techniques

Kernel PCA

- 커널 기법 : 비선형 함수인 커널함수를 이용해 비선형 데이터를 고차원 공간으로 매핑하는 기술
- from sklearn.decomposition import KernelPCA



가우시안 커널 함수를 이용한 비선형 데이터의 고차원(Feature Space)으로의 매핑

Incremental PCA

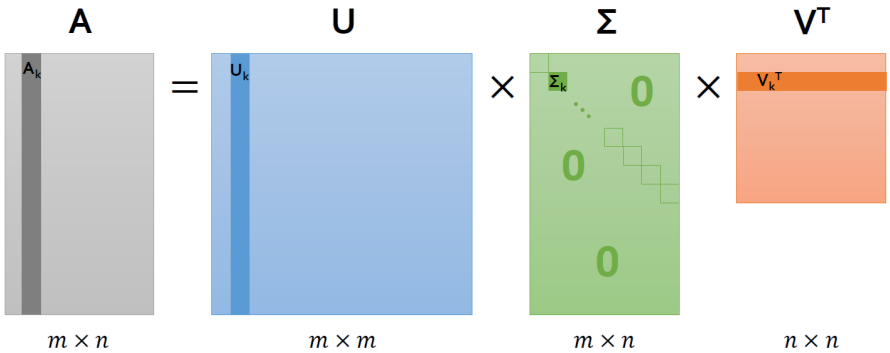
- 학습 데이터세트를 미니배치로 나눈 뒤 IPCA 알고리즘에 하나의 미니배치를 입력으로 넣어주는 방법
- from sklearn.decomposition import IncrementalPCA

Sparse PCA & Mini Batch Sparse PCA

- sparsity 제약 조건을 더해 확장해준 개념
- 희소한 데이터에서 차원 축소 & 변수 선택을 동시에 가능

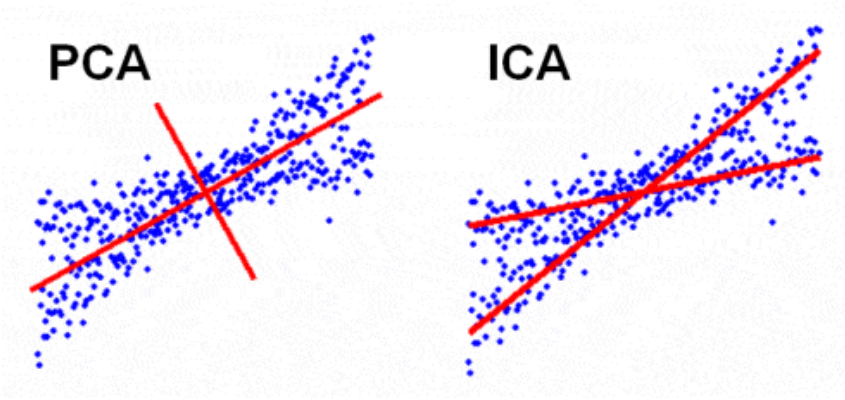
Truncated SVD

- sigma 행렬에 있는 대각원소, 즉 특이값 중 상위 일부 데이터만 추출해 분해하는 방식



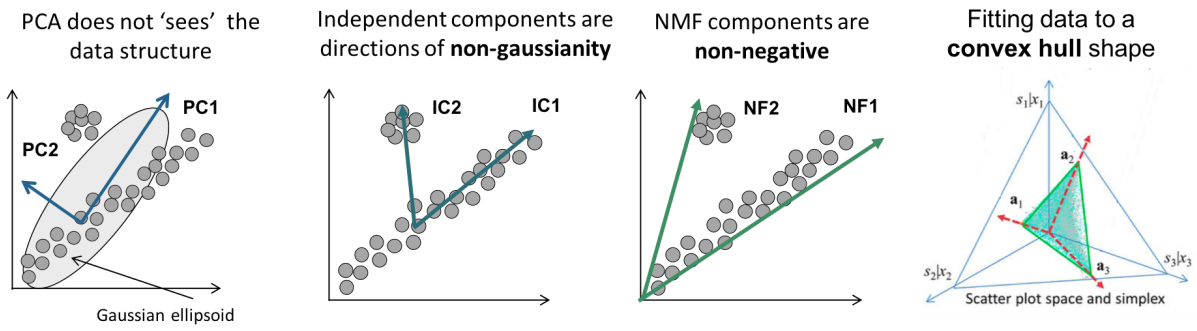
Independent Component Analysis - ICA

- 독립 성분 분석
- 가장 독립적인 축을 찾음 (cf. PCA는 데이터를 가장 잘 설명하는 축을 찾음)
- 독립성이 최대가 되는 벡터를 찾기 위해 독립성을 계산하는 알고리즘



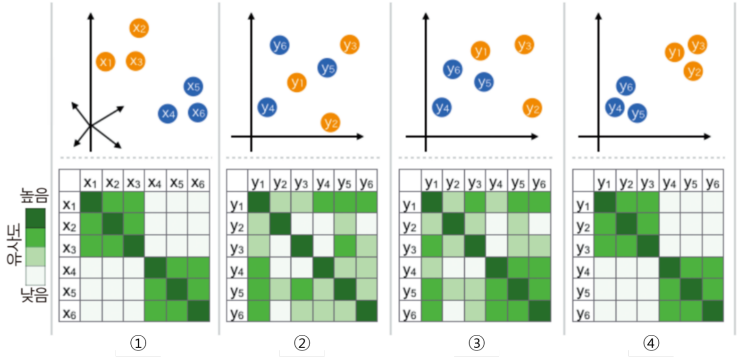
Non-negative Matrix Factorization

- 음수를 포함하지 않는 행렬 X를 음수를 포함하지 않는 행렬 W와 H의 곱으로 분해하는 알고리즘
- H : 각 행은 하나의 feature를 의미
- W : 각 행은 각각의 feature들을 얼마만큼 섞어 쓸 것인지에 관한 weight



t-SNE

- 매니폴드 학습의 하나



- 모든 i,j 쌍에 대하여 xi,xj의 유사도를 가우시안 분포를 이용하여 나타냄
- xi와 같은 개수의 점 yi를 낮은 차원 공간에 무작위로 배치하고, 모든 i,j 쌍에 관하여 yi,yj의 유사도를 `t-SNE`를 이용하여 나타냄
- 정의한 유사도 분포가 가능하면 같아지도록 데이터 포인트 yi를 갱신
- 수렴 조건까지 과정을 반복

The curse of dimensionality & Dimension reduction

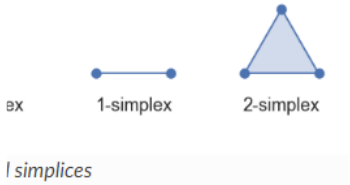
```
from matplotlib.pyplot as plt

plt.figure(figsize=(10,10))
for i, marker in enumerate(markers):
  mask = train_y == i
  plt.scatter(x_tsne[mask,0], x_tsne[mask,1], label=i, s=10, alpha=1, marker=marker)
plt.legend(bbox_to_anchor=(1,00,1), loc='upper left', fontsize=15)
```

UMAP

UMAP은 어떻게 작동할까? (Uniform Manifold Approximation and Projection) - 1

공부해야 할 필요성이 생겨서 글을 남기면서 공부하려고 한다. https://data-newbie.tistory.com/134?category=687142 https://umap-learn.readthedocs.io/en/latest/how_umap_works.html umap 과 t-sne의 차이를 볼 수 있는

https://data-newbie.tistory.com/169



l simplices

- 가까운 데이터를 이어줄 때 clustering 이용

```
!pip install umap-learn
!pip install umap-learn[plot]

import umap.plot
mapper = umap.UMAP().fit(train_x)
umap.plot.connectivity(mapper, show_points=True)

umap.plot.points(mapper, labels=train_y, theme='fire')
```

```
import plotly
import plotly.express as px
from umap import UMAP

umap_3d = UMAP(n_components=3, init='random', random_state=0)
x_umap = umap_3d.fit_transform(train_x)
umap_df = pd.DataFrame(x_umap)
train_y_sr = pd.Series(train_y,name='label')
print(type(x_umap))
new_df = pd.concat([umap_df,train_y_sr],axis=1)
fig = px.scatter_3d(
    new_df, x=0, y=1, z=2,
    color='label', labels={'color': 'number'}
)
fig.update_traces(marker_size=1)
fig.show()
```