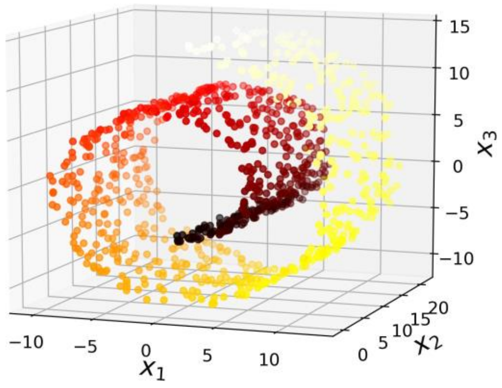
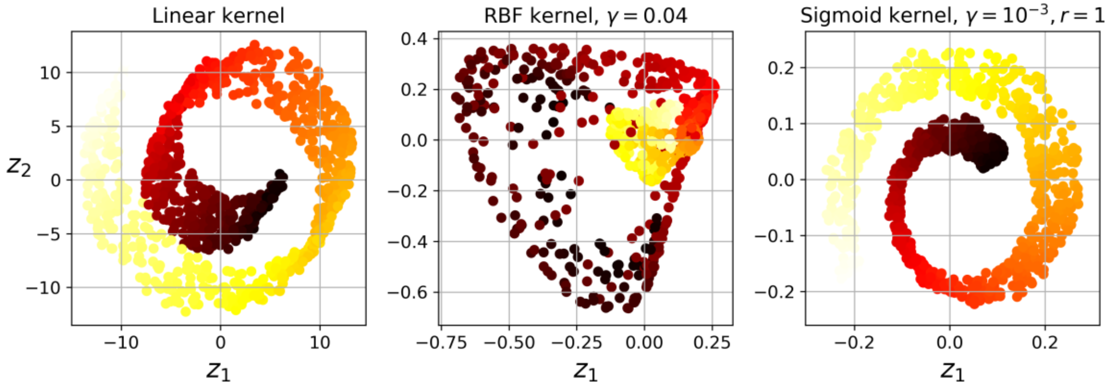
8.4 Kernel PCA



Swiss roll dataset



Swiss roll reduced to 2D using kPCA with various kernels

Kernel PCA

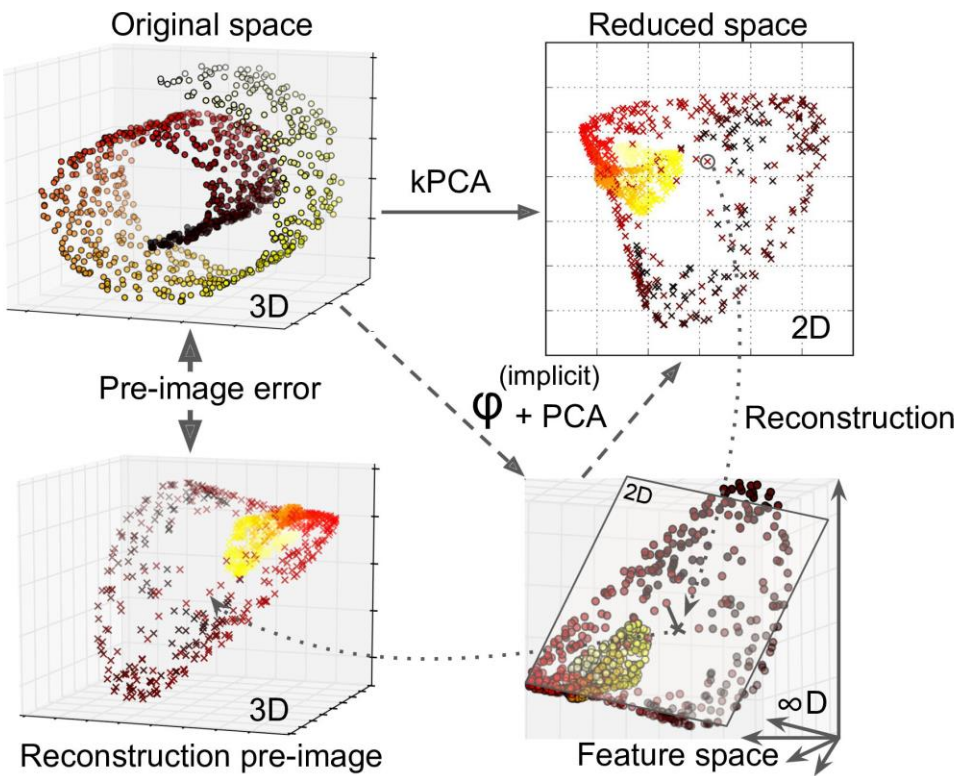
- **비지도 학습 →** 명확한 성능측정 기준 없음.
- 주로 지도학습의 전처리 단계로 사용됨.
  - 커널 및 하이퍼파라미터(HP) 선택 → 로지스틱 회귀 적용
  - GridSerarchCV 사용하여 가장 좋은 커널 및 가장 좋은 HP 선택.

8.4 Kernel PCA

```python
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())
    ])

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)
```

```python
>>> print(grid_search.best_params_)
{'kpca__gamma': 0.043333333333333335, 'kpca__kernel': 'rbf'}
```

8.4 Kernel PCA



Kernel PCA and the reconstruction pre-image error

Kernel PCA

- **(완전한) 비지도 학습**
  - 명확한 성능측정 기준 없음.
  - 가장 낮은 재구성 오차를 만드는 커널과 HP 선택.

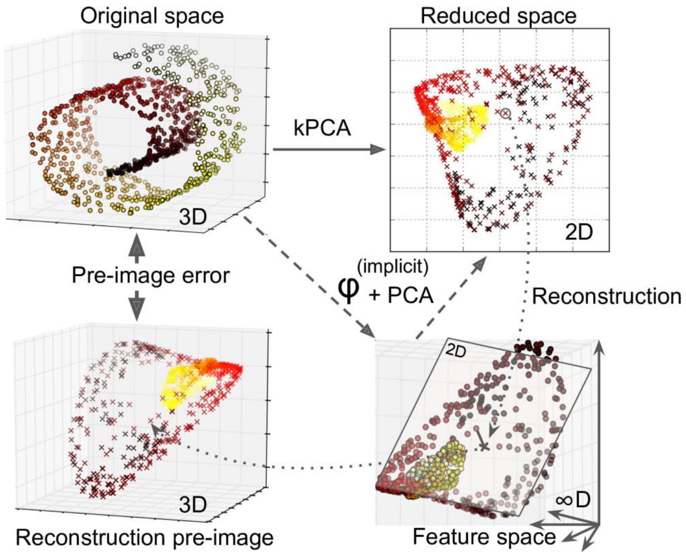
커널 트릭

1. 특성 맵을 사용하여 훈련세트를 무한 차원의 특성공간에 매핑 → 변환된 데이터셋을 선형 PCA를 사용하여 2D투영
2. 축소된 공간 샘플에 대한 PCA를 역전시키면, 재구성된 데이터포인트는 원본공간이 아닌 특성공간에 위치
3. 이 특성공간은 무한차원이므로 재구성된 포인트를 계산할 수 없고, 재구성에 따른 실제 에러를 계산할 수 없음.
4. 재구성된 포인트에 가깝게 매핑된 원본공간의 포인트를 찾을 수 있음

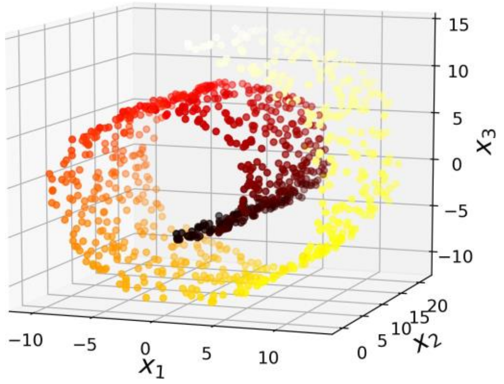
8.4 Kernel PCA

```python
rbf_pca = KernelPCA(n_components = 2, kernel="rbf", gamma=0.0433,
                    fit_inverse_transform=True)
X_reduced = rbf_pca.fit_transform(X)
X_preimage = rbf_pca.inverse_transform(X_reduced)
```

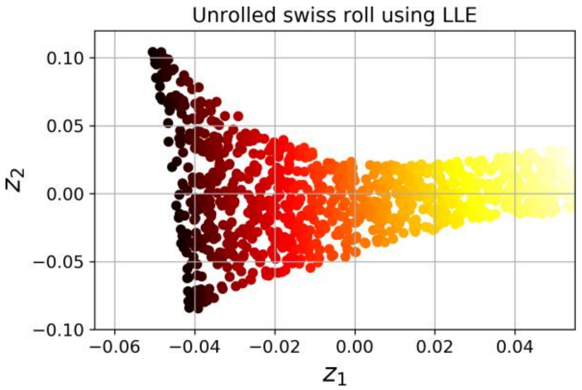
```python
>>> from sklearn.metrics import mean_squared_error
>>> mean_squared_error(X, X_preimage)
32.786308795766132
```



8.5 Locally Linear Embedding (LLE)



Swiss roll dataset



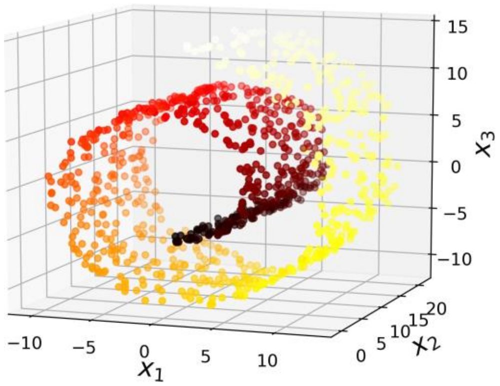
Unrolled Swiss roll using LLE

LLE

- **또 다른 강력한 비선형 차원 축소 방법.**
- 투영에 의존하지 않는 매니 폴드 학습
- 각 훈련샘플이 가장 가까운 이웃에 얼마나 선형적으로 연관되어 있는지 측정
- 그런다음, 국부적인 관계가 가장 잘 보존되는 훈련세터의 저차원 표현을 찾음.

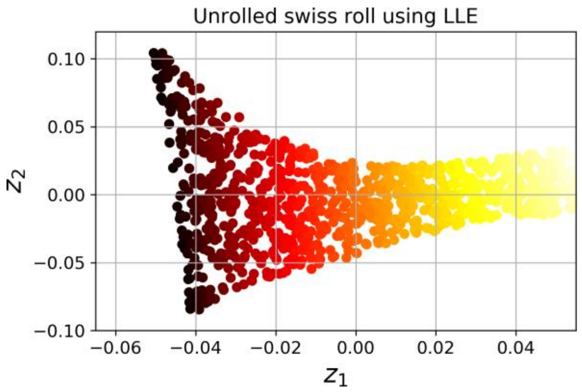
- 이 방법은 잡음이 너무 많지 않은 경우 꼬인 매니폴드를 펼치는데 잘 작동함.

8.5 Locally Linear Embedding (LLE)



Swiss roll dataset

```python
from sklearn.manifold import LocallyLinearEmbedding

lle = LocallyLinearEmbedding(n_components=2, n_neighbors=10)
X_reduced = lle.fit_transform(X)
```



Unrolled Swiss roll using LLE

- **스위스 롤이 완전히 펼쳐졌음.**
- 지역적으로 샘플간 거리가 잘 보존되어 있음.
- 그러나 크게 보면 샘플간 거리가 잘 유지되어 있지 않음.
- 스위스 롤의 오른쪽은 압축, 왼쪽은 확장되어 있음.
- 그러나 매니폴드를 모델링 하는데 잘 작동!!

8.5 Locally Linear Embedding (LLE)

Equation 8-4. LLE step 1: linearly modeling local relationships

n차원 공간

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

$$\text{subject to } \begin{cases} w_{i,j} = 0 & \text{if } \mathbf{x}^{(j)} \text{ is not one of the } k \text{ c.n. of } \mathbf{x}^{(i)} \\ \sum_{j=1}^{m} w_{i,j} = 1 & \text{for } i = 1, 2, \cdots, m \end{cases}$$

가중치 정규화

훈련샘플 $x^{(i)}$에 대해 가장 가까운 k개의 샘플을 찾는다.

선형함수로 $x^{(i)}$ 를 재구성 → $\sum_{j=1}^{m} w_{i,j} \mathbf{x}^{(j)}$

$x^{(i)}$와 $\sum_{j=1}^{m} w_{i,j} \mathbf{x}^{(j)}$ 사이의 제곱거리가 최소로 되는 $w_{i,j}$를 찾는다.

$x^{(j)}$가 $x^{(i)}$의 가장 가까운 이웃중 하나가 아니면 $w_{i,j}$=0.

Equation 8-5. LLE step 2: reducing dimensionality while preserving relationships

d차원 공간

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

$w_{i,j}$을 모두 담은 가중치 행렬은 은 훈련 샘플 사이에 있는 지역선형 관계를 담고 있음

이러한 관계가 잘 보존되도록 훈련샘플을 저 차원 공간으로 mapping수행.

$z^{(i)}$ 는 d차원 공간에서 $x^{(i)}$ 의 상(image)이라고 할때

$z^{(i)}$ 와 $\sum_{j=1}^{m} \hat{w}_{i,j} \mathbf{z}^{(j)}$사이의 거리가 최소화 되어야 함.

첫번째 단계와 비슷해 보이지만, 2번째 단계에서는 가중치는 고정하고 저차원 공간에서 샘플 이미지의 최적 위치를 바꾸며 찾아감.

8.5 Locally Linear Embedding (LLE)

Equation 8-4. LLE step 1: linearly modeling local relationships

n차원 공간

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

$$\text{subject to } \begin{cases} w_{i,j} = 0 & \text{if } \mathbf{x}^{(j)} \text{ is not one of the } k \text{ c.n. of } \mathbf{x}^{(i)} \\ \sum_{j=1}^{m} w_{i,j} = 1 & \text{for } i = 1, 2, \cdots, m \end{cases}$$

가중치 정규화

훈련샘플 $x^{(i)}$에 대해 가장 가까운 k개의 샘플을 찾는다.

선형함수로 $x^{(i)}$ 를 재구성 → $\sum_{j=1}^{m} w_{i,j}\mathbf{x}^{(j)}$

$x^{(i)}$와 $\sum_{j=1}^{m} w_{i,j}\mathbf{x}^{(j)}$ 사이의 제곱거리가 최소로 되는 $w_{i,j}$를 찾는다.

$x^{(j)}$가 $x^{(i)}$의 가장 가까운 이웃중 하나가 아니면 $w_{i,j}$=0.

Equation 8-5. LLE step 2: reducing dimensionality while preserving relationships

d차원 공간

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

$w_{i,j}$을 모두 담은 가중치 행렬은 은 훈련 샘플 사이에 있는 지역선형 관계를 담고 있음

이러한 관계가 잘 보존되도록 훈련샘플을 저 차원 공간으로 mapping수행.

$z^{(i)}$ 는 d차원 공간에서 $x^{(i)}$ 의 상(image)이라고 할때

$z^{(i)}$ 와 $\sum_{j=1}^{m} \widehat{w}_{i,j}\mathbf{z}^{(j)}$사이의 거리가 최소화 되어야 함.

첫번째 단계와 비슷해 보이지만, 2번째 단계에서는 가중치는 고정하고 저차원 공간에서 샘플 이미지의 최적 위치를 바꾸며 찾아감.

8.6 Other Dimensionality Reduction Techniques

랜덤 투영
- 랜덤한 선형 투영을 사용해 데이터를 저차원 공간으로 투영

다차원 스케일링 (MDS)
- 샘플간의 거리를 보존하면서 차원을 축소

Isomap
-각 샘플을 가장 가까운 이웃과 연결하는 식으로 그래프를 만듦, 샘플간의 지오데식 거리를 유지하면서 차원을 축소.

t-SNE
-비슷한 샘플은 가까이, 비슷하지 않은 샘플은 멀리 떨어지도록 하면서 차원을 축소. 주로 시각화에 사용. 고차원 공간에 있는 샘플의 군집을 시각화 할때 사용. (MNIST 데이터셋(784 차원)을 2D로 시각화할 때)