PCA

PCA: Goal: To reduce the dimensionality of the data, while retaining as much as possible of the variation present.

Drawbacks:

→ physical interpretation (not clear)

→ Not robust against outliers

→ Assumes linear combinat^n of features

→ sensitive to Scale of features

Ex: If one is mm and other feature in cm

⇒ wrong in chosing PC1

* Multi-dimensional Scaling (MDS) → (Based on preserving pairwise dirns)

PCA looks at variation, at which dir^n it is most spread out.

MDS is in d-dim space (for ex)

→ PCA maps i/p features from d dim feature space to K-dim latent features, with K<<d.

→ If we have large dataset (20 features)

Compute difference pairwise

Scale 20 features ⟶ 2 features.

→ MDS helps visualize data in low dimensions by creating a mapping that will also preserve the relative distance b/w data.

① Construct nxn dissimilar matrix (100 samples 100x100 matrix)

② Compare with PCA (10 features, 10x10 features)

classical approach to MDS

→ An $n \times n$ dissimilarity matrix D is provided, with matrix elements d_{ij} showing the euclidean distance b/w points i & j.

→ A double centred matrix B is formed (s.t mean of all rows & columns are zero)

$B = \frac{-1}{2} C D^2 C$ with $C = \| \frac{-1}{n} \|$

→ outputs $n \times 2$ or $n \times 3$ matrix

→ The eigenpair approach exploits the dissimilarity matrix D to obtain spatial co-ordinates x

$$z = \sqrt{\sum_{i=2}^{n} \sum_{j=1}^{i-1} \left(d_{ij} - \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2} \right)^2}$$

→ The value z can be thought of as an objective fn value, used to simplify find a solution (x_{ij}) resulting in an aggregate minimized distance equal to the given distance matrix.

→ MDS decomposes an $n \times n$ dissimilarity matrix D

o/p s $(n \times 2)$ or $(n \times 3)$

PCA vs MDS

(similar)

→ In PCA, we compute $N \times N$ covariance matrix

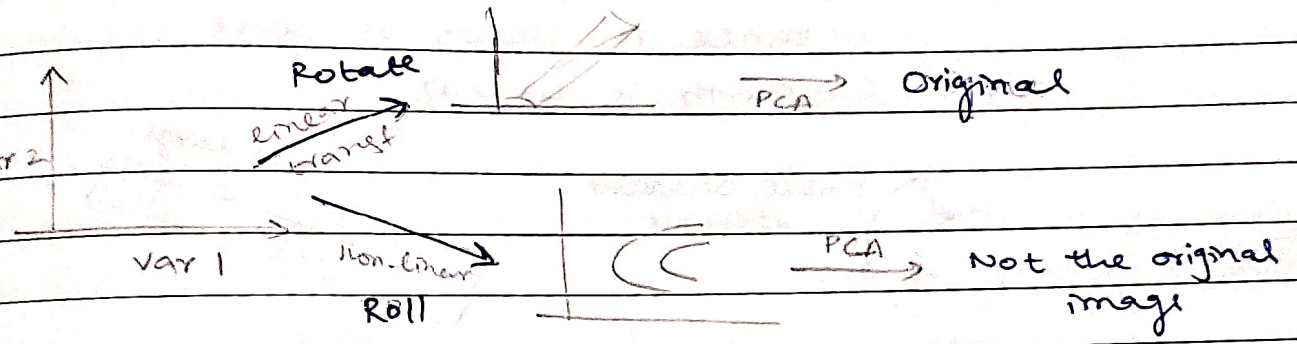
N → # of features of dataset.

→ In MDS, we compute the $D \times D$ dissimilarity matrix with D being the number of samples of the dataset

→ For euclidean dist d_{ij} in MDS
(it finds an embedding that preserves the interpoint distances equ to PCA).

* Non-linear data

Rotating data —apply→ PCA / MDS —reduce→ 2D (original data)



* Isomap Isometric feature mapping
Beautiful ⊛ Research Paper: Joshua B. Tenenbaum, Langford, Kin

→ MDS as we know seeks an embedding that preserves pairwise distances b/w data points.

→ Swiss Roll dataset
when data is curved (not straight)
Q. How to robustly measure distances.

* Manifold
→ A manifold is a space or space that locally resembles euclidean space near each point, but globally may be more complicated

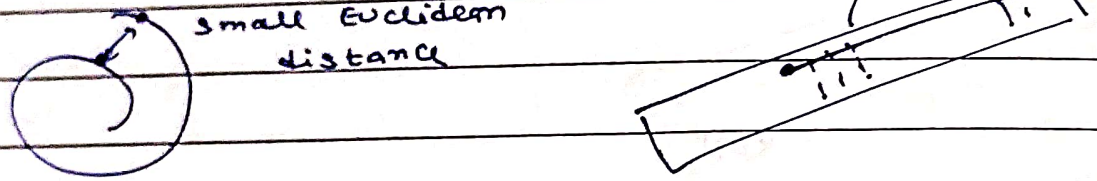
→ A manifold is called a 2-manifold when its surface appears to look relatively flat, but in reality is not. Earth is a 2-manifold.

* Geodesics (Geometry of surface → CRITERIA)

↳ Geodesics on manifolds define locally length minimizing curves, which minimizes the distance b/w the endpoints.

↳ Geodesics generalize the notion of a straight line in euclidean space to curved spaces.

like flights booking
distance is shown as curve
(∵ Earth is curved)



* Isometric Feature Mapping (Isomap)

Q: how to know structure / geometry of data?

↳ Geodesic distances measured on the manifold may be longer than the corresponding Euclidean st. line distance dij.

↳ The geodesic distances reflect the true low-dim geometry of the manifold, which the euclidean distance fails to capture.

↳ Isomap seeks to preserve the intrinsic geometry of data, as captured in the geodesic manifold.

ISOMAP ALGO

Q. How to compute geodesics rather than euclidean distances without knowing any manifold.

Step 1: Build the adjacency graph over high-dim points X.

→ Construct neighbourhood graph:
- Use KNN K → Hyperparameter
- Neighbours within a fixed radius (ε)

→ The graph G over all data points is defined by connecting points i & j (d_{ij}^X) provided they are closer than ε, or if i is one of the k nearest neighbours of j.

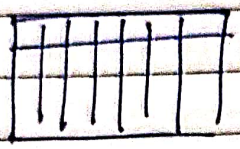
→ The edge length is d_{ij}^X

Step 2: Compute shortest paths using Floyd's Algo
→ If points i & j are linked by an edge; $d_{ij}^0 = d_{ij}^X$; else $d_{ij}^0 = \infty$
→ For each value of k = 1, 2, ... N in turn, replace all entries d_{ij}^0 by $\min\{d_{ij}^0, d_{ik}^0 + d_{kj}^0\}$
→ The matrix of final values $D^0 = \{d_{ij}^0\}$ will contain the shortest path distance b/w all pairs of points in G.

Step 3: Construct d-dimensional embedding by applying MDS to geodesic distances

Application:

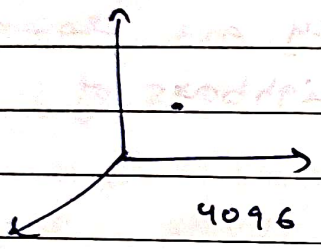
Ex: Simple images of face → 698 samples of 64x64 pixels images



each pixel has some value.

→ The images are thought of as points in a high dimensional (4096) vector space, with each input dimension corresponding to the brightness of one pixel in the image or the firing rate of one retinal ganglion cell.

→ Now increase the size



Each image will have some value

construct dataset

feature 1: 2 3 4 ... 4096

4096 → large vector space.

→ calculate geodesic distances

Isomap > PCA

Ex: Hand Rotation.

Shortcoming

computationally expensive