MDS

02 November 2024 08:44

Multi Dimensional Scaling — when? → 2D visualization; algo only works for euclidean data; identify struct in lower dimension.

↳ Assume we have a distance matrix $D \in R^{n \times n}$
↳ Contains distances $\delta_{ij} = |x_i - x_j|$ (lower dimensional space)

MDS tries to RECOVER the points $x_i \in R^d$ (original space)

↳ Another way of looking at it is:

Given abstract objects, $x_1, x_2 \dots x_n \in X$ (original space)
can we find an embedding $\phi : X \to R^d$
for some d such that
$|\phi(x_i) - \phi(x_j)| = d_{ij}$? (lower dimensional space)

Yes!

We can do it if distances d_{ij} come from R^d

"Matrix D is a euclidean distance matrix"

But

If the distance matrix doesn't belong to a space like that, you need some distortion of data. (for euclidean distance)

Types of MDS

Torgerson & Gower 1966 ① Classic MDS → is Matrix Euclidean?
- Y → embedding will be exact
- N → embedding won't be exact

$\delta_{ij} = d_{ij}$
Use eigendecomposition
Also: PCoA — principal coordinate analysis

special case of

Shepard & Kruskal 1964 ② Metric MDS → distances might be non-euclidean
$\delta_{ij} \approx d_{ij}$
Even if distance is euclidean, errors might still exist due to formulation
Uses iterative algo.
Also: Least square MDS

Shepard & Kruskal 1964 ③ Non-metric MDS → qualitative MDS
$d_{ij} \approx f(\delta_{ij})$
Ordinal ranking.
Also: Least square MDS.

* Original distance = δ_{ij}
Post MDS = d_{ij}

* Scaling in MDS: distances b/w all $i \neq j$ have to be simultaneously scaled.

Classic MDS + Gram Matrix

$$d_{ij}^2 \overset{\text{by definition}}{=} \|x_i - x_j\|^2 \overset{\text{inner prod}}{=} \langle x_i - x_j, x_i - x_j \rangle = \langle x_i, x_i \rangle + \langle x_j, x_j \rangle - 2\langle x_i, x_j \rangle$$

On rearranging:

$$\langle x_i, x_j \rangle = \frac{1}{2}\left(\underbrace{\langle x_i, x_i \rangle}_{d(0, x_i^2)} + \underbrace{\langle x_j, x_j \rangle}_{d(0, x_j^2)} - d_{ij}^2 \right)$$

$\langle x_i, x_j \rangle = \frac{1}{2}\left(\underbrace{\langle x_i, x_i \rangle}_{d(0, x_i^2)} + \underbrace{\langle x_j, x_j \rangle}_{d(0, x_j^2)} - d_{ij} \right)$

[norm of x_i] [norm of x_j]

Wait but what is a gram matrix?

for vectors $\bar{u}, \bar{v}$, $\langle \bar{u}, \bar{v} \rangle$ denotes their INNER PRODUCT SPACE.

so now, given a set of vectors $\{v_1, v_2 \dots v_n\}$, the gram matrix is defined as:

$$G = \begin{bmatrix} \langle v_1, v_1 \rangle & \langle v_1, v_2 \rangle & \dots & \langle v_1, v_n \rangle \\ \langle v_2, v_1 \rangle & \langle v_2 v_2 \rangle & \dots & \langle v_2, v_n \rangle \\ \vdots & & & \\ \langle v_n, v_1 \rangle & \langle v_n, v_2 \rangle & \dots & \langle v_n, v_n \rangle \end{bmatrix}_{n \times n} \rightarrow \text{symmetric matrix}$$

① G (encodes) represents the inner products of the vectors
- diagonals are squared norms of vectors
- off-diagonals are the inner product.

② Used to determine linear independence

All eigenvalues > 0

Linearly indep. if G is +ve definite

If some $\lambda = 0$ ⇒ vectors are linearly dependent.

③ Coordinate transformation

Vector → [Matrix] → Transformed vector.

④ Represents similarity between data points in a TRANSFORMED FEATURE SPACE.

⑤ Gram matrix of columns of $m \times n$ matrix $M = M^T M$ = $n \times n$ matrix

↓

We can choose the origin of data points.

↓

for simplicity we take 1st point as origin = Translation invariance.

$\langle x_i, x_j \rangle = \frac{1}{2}\left(d_{1i}^2 + d_{1j}^2 - d_{ij}^2\right)$

↳ gram matrix (s)

Express entries of gram matrix S w/ $S_{ij} = \langle x_i, x_j \rangle$ in terms of dist. values

S is +ve definite

↓

Decompose in terms of XX^T

↳ $S = XX^T$ when $X \in R^{n \times d}$ ↳ we've to find d.

Back to MML + diagonalizn

Say A is $n \times n$ matrix w/ n linearly independent eigenvectors.

$S = |\bar{e}_1, \bar{e}_2, \dots \bar{e}_n| \rightarrow$ eigenvectors ka set

$AS = A|\bar{e}_1, \bar{e}_2 \dots \bar{e}_n|$

$AS = [\lambda \bar{e}_1, \lambda \bar{e}_2 \dots \lambda \bar{e}_n]$

$AS = [\bar{e}_1\ \bar{e}_2 \dots \bar{e}_n] \times \begin{bmatrix} \lambda_1 & 0 & 0 \dots 0 \\ 0 & \lambda_2 \dots 0 \end{bmatrix} = S\Lambda$

$AS = [\bar{e}_1\ \bar{e}_2 \dots \bar{e}_n] \times \begin{bmatrix} \lambda_1 & 0 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & & 0 \\ 0 & 0 & \dots & & \lambda_n \end{bmatrix} = S\Lambda$

Λ = diagonal matrix → eigenvalues of A in diag.

SO

$A = S\Lambda S^{-1}$

for a symmetric matrix, eigenvectors are orthogonal. $\Rightarrow S^{-1} = S^T$

$A = S\Lambda S^{-1} = S\Lambda S^T$

Computing the eigendecomposition of S (gram matrix):

$S = V \Lambda V^T$ (V: eigenvector, Λ: eigenvalues)

$S = XX^T$ when $X \in R^{n \times d}$

} kinda like $X^2 = V^2\Lambda$, $X = V\sqrt{\Lambda}$

We can define $X = V\sqrt{\Lambda}$

If we want $d \leq n$, then

→ V_d = first d columns of V

→ Λ_d = first d eigenvalues of Λ

→ $X = V_d \sqrt{\Lambda_d}$

Row i of X gives the coordinate of x_i

So how is classical MDS so different?

distance b/w point & transformed point: euclidean distance

↓

If distance X but ranking assessment ✓

→ parametric transformation.

→ choice of d in V_d: look @ loss function

If distance is NOT euclidean, closed form solution may not be possible.

$Stress_D(x_1, x_2 \dots x_N) = \sum (d_{ij} - |x_i - x_j|^2)^{1/2}$ (new coordinates in lower dimension d')

$Stress_D$: MDS minimizes this (Gradient descent)

d_{ij}: original dist b/w ith & jth points

↓ generalize

$\sigma(x) = \sum w_{ij} (d_{ij}(x) - \delta_{ij})^2$

δ_{ij} → given distance b/w points i & j.

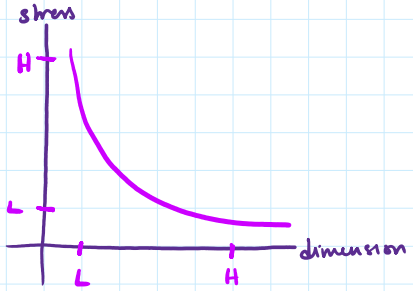
w_{ij}: weight b/w a pair of points

confidence in similarity b/w points i & j.

When do we get high stress?

Distorted representation in reduced dimension

→ means 'd' is too low.



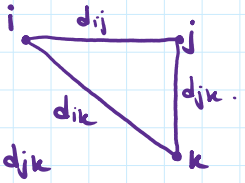
So Classical MDS: ① Compute gram matrix
(special case - metric MDS) ② Compute eigendecomposition
↳ m largest positive eigenvalue d.
③ Construct matrix containing final coordinates using projection.

Now, what does it take to be a 'metric distance function'?

① Positivity: $d_{ij} \geq 0 \quad i \neq j$
$d_{ii} = 0$

② Symmetry: $d_{ij} = d_{ji}$

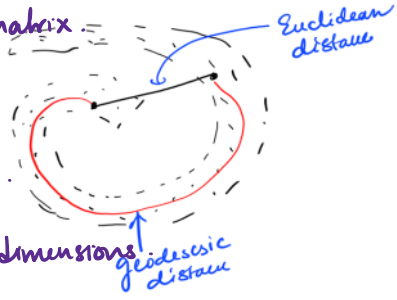
③ Triangle inequality: $d_{ik} \leq d_{ij} + d_{jk}$



Satisfied by: euclidean → only this gives closed form solution
- cosine
- hamming
- manhattan
- minkowski
- chebyshev
- Jaccard

} remaining needs to be optimized w/ gradient descent.

isometric feature map
ISOMAP — ultrasound + echocardiography images : interpretability
↳ extends MDS : utilizes geodesic distances introduced by neighboring graph.
↳ no diff. b/w MDS & ISOMAP once we get distance matrix.



Step 1: construct neighborhood graph of data X.
Step 2: Use dijkstra or floyd-warshall for distance b/w nodes.
Step 3: Apply MDS on distance matrix & extract coords w/ req. dimensions

Manifold learning ← isomap ✓
LLE X (local linear embedding)
Laplacian Eigenmap X

↓
find lower dimensional manifold within higher dimensional space.
↳ o/p = new coordinates in 2D space

Why manifold?
PCA + dimensionality reduction? Can.
but non-linear relationships? Cannot → sklearn.manifold

If highly clustered → t-SNE

Oh but
- noise affects manifold — PCA can tolerate
- manifold has no optimal no. of o/p dimensions — PCA works w/ explained variance