DATA COMPRESSION WITH DIMENSIONALITY REDUCTION - CHAPTER 5

- ALTERNATIVE APPROACH TO FEATURE SELECTION FOR DIMENSIONALITY REDUCTION → FEATURE EXTRACTION
- DATA COMPRESSION IMPORTANT IN MACHINE LEARNING → HELPS STORE + ANALYSE INCREASING AMOUNT OF DATA THAT ARE PRODUCED AND COLLECTED.

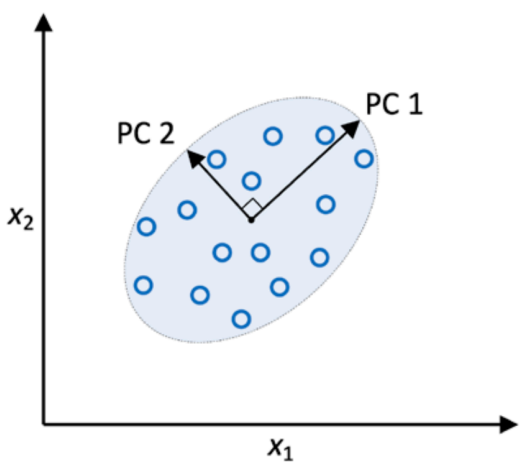
UNSUPERVISED DIMENSIONALITY REDUCTION VIA COMPONENT ANALYSIS

- FEATURE EXTRACTION → TRANSFORM OR PROJECT DATA ONTO NEW FEATURE SPACE.
- CONTEXT OF DIMENSIONALITY REDUCTION → FEATURE EXTRACTION → DATA COMPRESSION WITH GOAL TO MAINTAIN MOST OF THE RELEVANT INFORMATION.
  - ↳ NOT ONLY USED TO IMPROVE STORAGE SPACE OR COMPUTATIONAL EFFICIENCY
  - ↳ USED TO IMPROVE PREDICTIVE PERFORMANCE BY REDUCING CURSE OF DIMENSIONALITY

PRINCIPAL COMPONENT ANALYSIS (PCA)

- PCA → UNSUPERVISED LINEAR TRANSFORMATION TECHNIQUE → USED IN MANY DIFFERENT FIELDS → MAINLY FEATURE EXTRACTION + DIMENSIONALITY REDUCTION.
- SOME POPULAR APPLICATION OF PCA → EXPLORATORY DATA ANALYSIS, DENOISING SIGNALS IN STOCK MARKET TRADING, ANALYSIS OF GENOME DATA + GENE EXPRESSION LEVELS.
- PCA → HELPS US IDENTIFY PATTERNS BASED ON CORRELATION BETWEEN FEATURES.
  - ↳ LOOKS AT HIGH-DIMENSIONAL DATA, THEN ASKS: "IN WHICH DIRECTIONS DO THE POINTS SPREAD OUT THE MOST?" → DIRECTIONS WHERE DATA VARIES THE MOST.
    - ↳ THEN
      - → BUILDS NEW AXIS (NEW FEATURES) ALONG "MAXIMUM SPREAD" DIRECTIONS.
      - → KEEPS ONLY SOME OF THOSE AXES (OFTEN FIRST FEW) → END UP WITH FEWER DIMENSIONS THAN STARTED WITH.
      - → MAKES SURE NEW AXES ARE AT RIGHT ANGLES TO EACH OTHER (ORTHOGONAL) → NEW FEATURE CAPTURES A DIFFERENT, NON-OVERLAPPING ASPECT OF VARIATION.

SHORT VERSION: PCA ROTATES + COMPRESSES DATA INTO FEWER, UNCORRELATED DIMENSIONS THAT CAPTURE AS MUCH OF ORIGINAL VARIATION AS POSSIBLE.



- x_1, x_2 ⇒ ORIGINAL FEATURE AXES → PC 1, PC 2 → PRINCIPAL COMPONENTS
- PCA FOR DIMENSIONALITY REDUCTION → $d \times k$ - DIMENSIONAL TRANSFORMATION MATRIX, W
  - ↳ ALLOWS US TO MAP VECTOR OF FEATURE EXAMPLES, x TO NEW k-DIMENSIONAL FEATURE SUBSPACE → FEWER DIMENSIONS THAN ORIGINAL d-DIMENSIONAL FEATURE SPACE.

$$x = [x_1, x_2, \dots x_d], \quad x \in \mathbb{R}^d$$

TRANSFORMED BY TRANSFORMATION MATRIX, $W \in \mathbb{R}^{d \times k}$

$$xW = z$$

OUTPUT VECTOR:

$$z = [z_1, z_2, \dots, z_k], \quad z \in \mathbb{R}^k$$

- RESULT OF TRANSFORMING → FIRST PRINCIPAL COMPONENT WILL HAVE LARGEST VARIANCE.
  - ↳ PC1 CAPTURES THE MOST INFORMATION/VARIATION IN DATA → IF YOU LOOK AT ONE AXIS ONLY → THE SPREAD OF POINTS IS AS LARGE AS POSSIBLE.
- AFTER PC1, PCA LOOKS FOR SECOND DIRECTION (PC2) → BUT, PC2 MUST BE RIGHT ANGLE (ORTHOGONAL) TO PC1 → SO PC1 AND PC2 DON'T OVERLAP.
- THIS CAN KEEP GOING → PC3 → HAS TO BE ORTHOGONAL TO PC1 + PC2.
- PCA DIRECTIONS → SENSITIVE TO SCALING. ⇒ STANDARDISE FIRST.

APPROACH SUMMARISED:

1. STANDARDISE d-DIMENSIONAL DATASET
2. CONSTRUCT COVARIANCE MATRIX
3. DECOMPOSE COVARIANCE MATRIX INTO EIGENVECTORS + EIGENVALUES.
4. SORT EIGENVALUES BY DECREASING ORDER → TO RANK CORRESPONDING EIGENVECTORS.
5. SORT EIGENVECTORS → CORRESPONDS WITH k LARGEST EIGENVALUE → k IS DIMENSIONALITY OF NEW FEATURE SUBSPACE ($k \leq d$)
6. CONSTRUCT PROJECTION MATRIX, W FROM "TOP" k EIGENVECTORS.
7. TRANSFORM d-DIMENSIONAL INPUT DATASET, X, USING PROJECTION MATRIX, W → OBTAIN NEW k-DIMENSIONAL FEATURE SUBSPACE.

EIGENDECOMPOSITION

- FACTORISATION OF SQUARE MATRIX INTO EIGENVALUES + EIGENVECTORS → CORE OF PCA PROCEDURE
- COVARIANCE MATRIX - $A = A^T$
- DECOMPOSED SYMMETRIC VECTOR → EIGENVALUES ARE REAL NUMBERS + EIGENVECTORS ARE ORTHOGONAL TO EACH OTHER.
- DECOMPOSE COVARIANCE MATRIX → EIGENVECTORS ASSOCIATED WITH HIGHEST EIGENVALUE CORRESPONDS TO DIRECTION OF MAXIMUM VARIANCE.

FOUR STEPS OF PCA:

1. STANDARDISE THE DATA
2. CONSTRUCTING THE COVARIANCE
3. OBTAINING THE EIGENVALUES + EIGENVECTORS OF COVARIANCE MATRIX
4. SORTING THE EIGENVALUES BY DECREASING ORDER TO RANK THE EIGENVECTORS.

STEP 2

- $d \times d$ - DIMENSIONAL COVARIANCE MATRIX, d = NUMBER OF DIMENSIONS IN THE DATASET.
- COVARIANCE BETWEEN TWO FEATURES, x_j AND x_k → ON POPULATION LEVEL:

$$\sigma_{jk} = \frac{1}{n-1} \sum_{i=1}^{n} \left(x_j^{(i)} - \mu_j\right)\left(x_k^{(i)} - \mu_k\right)$$

μ_j, μ_k = SAMPLE MEANS OF FEATURE j, k SAMPLE MEAN = ZERO → IF DATABASE IS STANDARDISED

- POSITIVE COVARIANCE BETWEEN TWO FEATURES = FEATURES INCREASE + DECREASE TOGETHER
- NEGATIVE COVARIANCE → FEATURES VARY IN OPPOSITE DIRECTIONS.
- COVARIANCE MATRIX OF THREE FEATURES:

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_2^2 & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 \end{bmatrix}$$

⇒ COVARIANCE MATRIX REPRESENTS PRINCIPAL COMPONENT. (DIRECTION OF MAXIMUM VARIANCE)
↳ EIGENVALUES WILL DEFINE THEIR MAGNITUDE

STEP 3

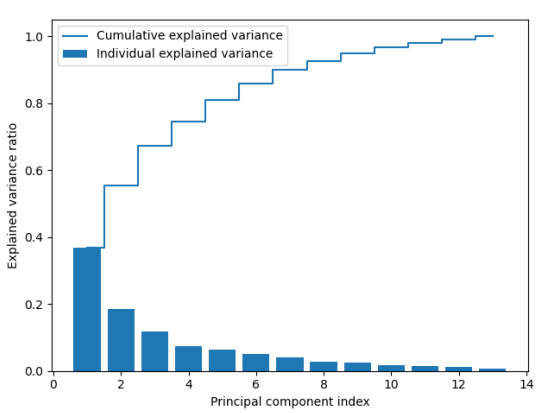
- OBTAIN EIGENPAIRS FOR COVARIANCE MATRIX
- EIGENVECTOR v:

$$\Sigma v = \lambda v \qquad \lambda = \text{SCALAR} = \text{EIGENVALUE}$$

TOTAL AND EXPLAINED VARIANCE

- ONLY SELECT SUBSET OF EIGENVECTORS (PRINCIPAL COMPONENT) THAT CONTAINS MOST OF THE INFORMATION (VARIANCE)
- EIGEN VALUES DEFINE MAGNITUDE → SORT BY DECREASING MAGNITUDE → INTERESTED IN TOP k EIGENVECTORS
- VARIANCE EXPLAINED RATIO

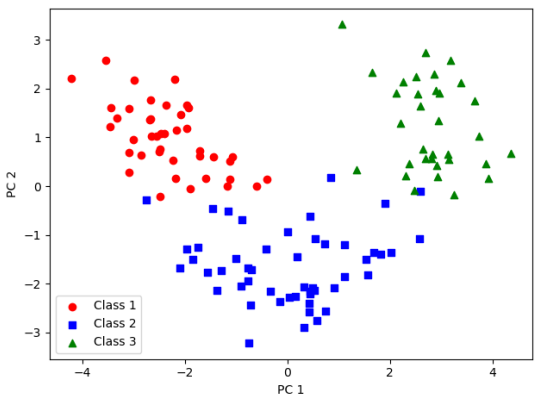
$$\frac{\lambda_j}{\sum_{j=1}^{d} \lambda_j}$$



- FIRST PRINCIPAL → ACCOUNTS TO APPROX. 40% OF VARIANCE.
- FIRST TWO PRINCIPAL COMPONENTS → COMBINED ALMOST 60% OF VARIANCE

FEATURE TRANSFORMATION

- PCA STEPS 5, 6, 7 (ABOVE)
  - ↳ SORT EIGENPAIRS BY DESCENDING ORDER OF EIGENVALUES
  - ↳ CONSTRUCT PROJECTION MATRIX FROM SELECTED EIGENVECTORS
  - ↳ PROJECTION MATRIX TO TRANSFORM DATA ONTO LOWER-DIMENSIONAL SUBSPACE.



- DATA MORE SPREAD ALONG PC1. → CONSISTENT WITH VARIANCE RATIO PLOT (ABOVE)

PCA UNSUPERVISED TECHNIQUE THAT DOESN'T USE ANY CLASS LABEL INFORMATION

ASSESSING FEATURE CONTRIBUTIONS

- PCA → CREATE PRINCIPAL COMPONENTS → REPRESENT LINEAR COMBINATIONS OF FEATURES.
- LOADING → HOW MUCH EACH ORIGINAL FEATURE CONTRIBUTES TO A GIVEN PRINCIPAL COMPONENT.
  - ↳ FACTOR LOADINGS COMPUTED → SCALING EIGENVECTORS BY SQUARE ROOT OF EIGEN VALUES.
    - ↳ RESULT: CORRELATION BETWEEN ORIGINAL FEATURES + PRINCIPAL COMPONENTS

SUPERVISED DATA COMPRESSION WITH LINEAR DISCRIMINANT ANALYSIS (LDA)

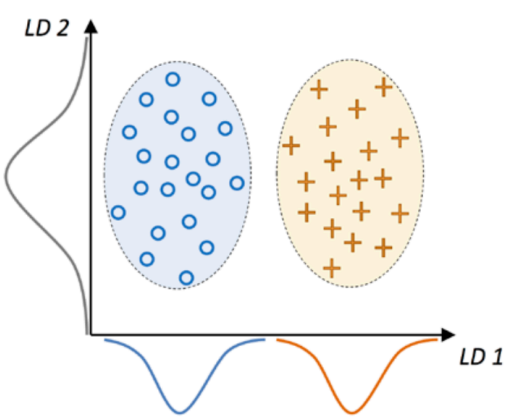
- LINEAR DISCRIMINANT ANALYSIS (LDA) → LINEAR TRANSFORMATION TECHNIQUE THAT TAKES CLASS LABEL INFORMATION INTO ACCOUNT.
- GENERAL CONCEPT → SIMILAR TO PCA → LDA FINDS THE FEATURE SUBSPACE THAT OPTIMISES CLASS SEPARABILITY.

PCA VS. LDA

- PCA + LDA BOTH LINEAR TRANSFORMATION TECHNIQUES → REDUCE NUMBER OF DIMENSIONS IN DATASET
- PCA TENDS TO RESULT IN BETTER CLASSIFICATION TASKS COMPARED TO LDA → WEIRD BECAUSE WE WOULD THINK THAT SCREENINGED WOULD BE BETER.
  - ↳ A.M. MARTINEZ

FISHER LDA

↳ LDA SOMETIMES CALLED FISHER LDA → RONALD A. FISHER (1936)
  - ↳ GENERALISED FOR MULTICLASS PROBLEMS BY C. RADHAKRISHNA RAO → FOR EQUAL CLASS COVARIANCE + NORMALLY DISTRIBUTED CLASSES.



- LDA FOR TWO-CLASS PROBLEM.
  - ↳ CLASS 1 CIRCLES
  - ↳ CLASS 2 CROSSES
- LINEAR DISCRIMINANT → x-AXIS (LD 1) → SEPARATE TWO NORMAL DISTRIBUTED CLASSES
- y-AXIS (LD 2) → VARIANCE IN DATASET → LOT OF VARIANCE IN DATASET
  - ↳ WOULD FAIL AT GOOD LINEAR DISCRIMINANT → DOESN'T CAPTURE ANY OF CLASS-DISCRIMINATORY INFORMATION.
- ASSUMPTION OF LDA → DATA IS NORMALLY DISTRIBUTED
  - ↳ ALSO ASSUMED → CLASSES HAVE IDENTICAL COVARIANCE MATRICES
  - ↳ TRAINING EXAMPLES ARE STATISTICALLY INDEPENDANT.
- IF ASSUMPTIONS ARE SLIGHTLY VIOLATED → LDA FOR DIMENSIONALITY REDUCTION CAN STILL WORK REASONABLY WELL.

LDA WORKINGS:

1. STANDARDISE d-DIMENSIONAL DATASET (d = NUMBER OF FEATURE)
2. EACH CLASS → COMPUTE d-DIMENSIONAL MEAN VECTOR → LDA TAKES LABELLED INFORMATION INTO ACCOUNT.
3. CONSTRUCT BETWEEN-CLASS SCATTER MATRIX S_B + WITHIN-CLASS SCATTER MATRIX S_W
4. COMPUTE EIGENVECTORS + CORRESPONDING EIGENVALUE OF MATRIX $S_W^{-1} S_B$
5. SORT EIGENVALUES BY DECREASING ORDER → RANK CORRESPONDING EIGENVECTORS
6. CHOOSE k EIGENVECTORS → k LARGEST EIGENVALUE → CONSTRUCT $d \times k$-DIMENSIONAL TRANSFORMATION MATRIX, W → EIGENVECTOR ARE COLUMN OF MATRIX
7. PROJECT EXAMPLES ONTO NEW FEATURE SUBSPACE USING TRANSFORMATION MATRIX, W

- LDA SIMILAR TO PCA → DECOMPOSING MATRICIES INTO EIGENVALUES + EIGENVECTORS → FOR NEW LOWER-DIMENSION FEATURE SPACE

COMPUTING SCATTER MATRICES

- ALWAYS STANDARDISE THE FEATURES
- CALCULATION OF MEAN VECTORS → CONSTRUCT WITHIN-CLASS SCATTER + BETWEEN-CLASS SCATTER MATRIX

$$m_i = \frac{1}{n_i} \sum_{x \in D_i} x_m$$

m_i = MEAN VECTOR
μ_m = MEAN FEATURE VALUE

- WINE DATASET (THREE MEAN VECTORS):

$$m_i = \begin{bmatrix} \mu_i \text{ alcohol} \\ \mu_i \text{ malic acid} \\ \vdots \\ \mu_i \text{ proline} \end{bmatrix}^T \quad i \in \{1, 2, 3\}$$

- MEAN VECTORS → COMPUTE WITHIN-CLASS SCATTER MATRIX, S_W:

$$S_W = \sum_{i=1}^{c} S_i$$

- CALCULATED BY SUMMING UP INDIVIDUAL SCATTER MATRICES

$$S_i = \sum_{x \in D_i} (x - m_i)(x - m_i)^T$$

ASSUMPTION: CLASS LABELS IN TRAINING DATASET ARE UNIFORMLY DISTRIBUTED.
↳ VIOLATED → CHECK WITH CLASS LABEL DISTRIBUTION

SCALE INDIVIDUAL SCATTER MATRICES, S_i → BEFORE WE SUM THEM UP.
↳ DIVIDE SCATTER MATRICES BY NUMBER OF CLASS-EXAMPLES, n_i
↳ SCATTER MATRIX → SAME AS COMPUTING COVARIANCE MATRIX Σ_i
↳ NORMALISED VERSION OF SCATTER MATRIX

$$\Sigma_i = \frac{1}{n_i} S_i = \frac{1}{n_i} \sum_{x \in D_i} (x - m_i)(x - m_i)^T$$

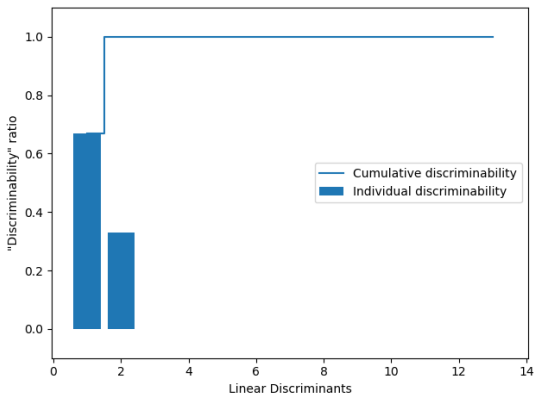
- NEXT STEP → COMPUTE BETWEEN-CLASS SCATTER MATRIX S_B:

$$S_B = \sum_{i=1}^{c} n_i (m_i - m)(m_i - m)^T$$

LINEAR DISCRIMINANTS FOR NEW FEATURE SUBSPACE

- INSTEAD OF EIGENDECOMPOSITION → SOLVE GENERALISED EIGENVALUE PROBLEM OF MATRIX $S_W^{-1} S_B$
- LDA ⇒ NUMBER OF LINEAR DISCRIMINANTS AT MOST $c-1$, c = NUMBER OF CLASS LABELS → S_B = SUM OF c MATRICES WITH RANK ONE OR LESS

COLLINEARITY ⇒ PERFECT CASE (ALL ALIGNED EXAMPLE POINTS FALL ON STRAIGHT LINE)
↳ COVARIANCE MATRIX = RANK ONE → EIGENVECTOR WITH NONZERO EIGENVALUE



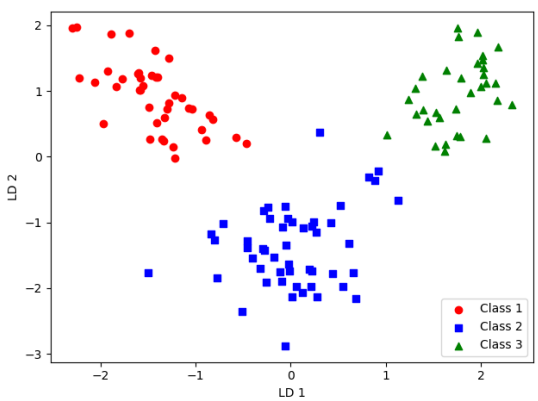
- MEASURE HOW MUCH CLASS-DISCRIMINATORY INFORMATION CAPTURED BY LINEAR DISCRIMINANTS (EIGENVECTORS)
- FIRST TWO LINEAR DISCRIMINANTS = CAPTURE 100% OF USEFUL INFORMATION

```
Matrix W:
 [[-0.1481 -0.4092]
 [ 0.0908 -0.1577]
 [-0.0168 -0.3537]
 [ 0.1484  0.3223]
 [-0.0163 -0.0817]
 [ 0.1913  0.0842]
 [-0.7338  0.2823]
 [-0.075  -0.0102]
 [ 0.0018  0.0907]
 [ 0.294  -0.2152]
 [-0.0328  0.2747]
 [-0.3547 -0.0124]
 [-0.3915 -0.5958]]
```

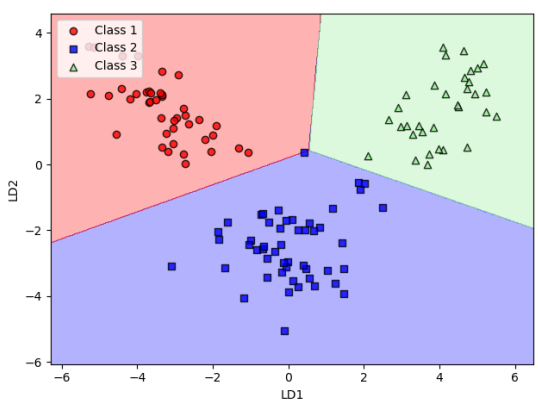
- TWO DISCRIMINATIVE EIGENVECTOR COLUMNS STACKED
  ↳ CREATE TRANSFORMATION MATRIX W

PROJECTING EXAMPLES ONTO NEW FEATURE SUBSPACE

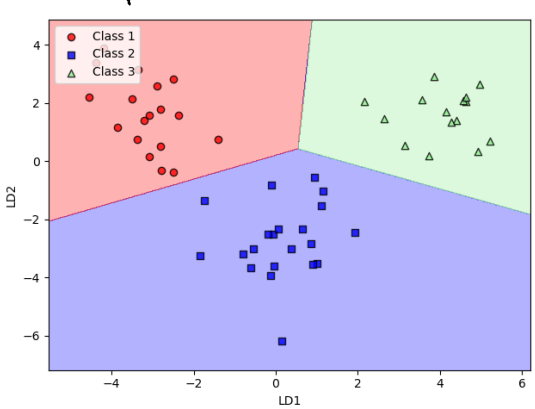
- TRANSFORMATION MATRIX W → TRANSFORM TRAINING DATASET → MULTIPLY MATRICES ⇒ $X' = XW$



- WHEN CLASSES → PERFECTLY LINEARLY SEPARABLE IN NEW FEATURE SUBSPACE



- LOWER-DIMENSION DATASET → LOGISTIC REGRESSION
  ↳ ONE EXAMPLE ONLY MISCLASSIFIED
  ↳ LOWER REGULARISATION STRENGTH → SHIFT DECISION BOUNDARIES

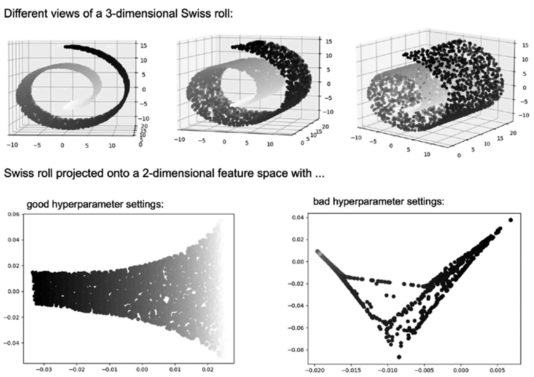


NONLINEAR DIMENSIONALITY REDUCTION VIA VISUALISATION

- T-DISTRIBUTED NEIGHBOR EMBEDDING (t-SNE) → FREQUENTLY USED IN LITERATURE → VISUALISE HIGH-DIMENSIONAL DATA SETS IN TWO OR THREE DIMENSIONS.
- t-SNE → PLOT IMAGE OF HANDWRITTEN IMAGES IN 2-DIMENSIONAL SPACE.

WHY CONSIDER NONLINEAR DIMENSIONALITY REDUCTION?

- DEVELOPMENT + APPLICATION OF NONLINEAR DIMENSIONALITY REDUCTION TECHNIQUE → MANIFOLD LEARNING → SCIKIT-LEARN LIBRARY COVERS OTHER COMPLEX ONES TOO.
- MANIFOLD LEARNING → LOWER DIMENSIONAL TOPOLOGICAL SPACE EMBEDDED IN HIGH-DIMENSIONAL SPACE
  - ↳ ALGORITHM HAS TO CAPTURE COMPLICATED STRUCTURE OF DATA → TO PROJECT IT ONTO LOWER-DIMENSIONAL SPACE. → RELATIONSHIP BETWEEN DATA POINTS IS PRESERVED.

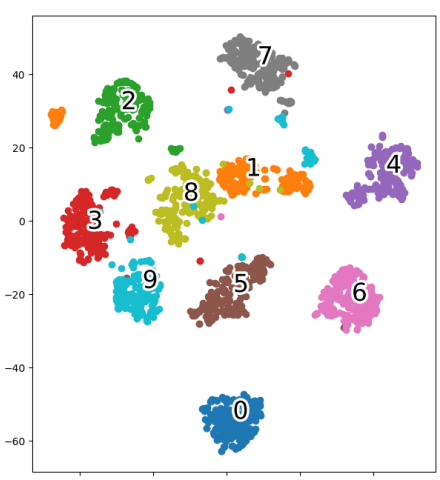


- VERY POWERFUL BUT ALSO VERY HARD TO USE.
  - ↳ DIFFICULT → WORKING WITH HIGH-DIMENSIONAL DATASETS THAT WE CANNOT VISUALISE + STRUCTURE IS NOT OBVIOUS.
    - ↳ COULD TRY PROJECTING DATASET TO 2 OR 3 DIMENSIONS] HARD OR IMPOSSIBLE TO ASSESS QUALITY
      - ↳ NOT ENOUGH FOR CAPTURING COMPLICATED RELATIONSHIPS
      - ↳ HENCE PEOPLE RELY ON PCA + LDA FOR DIMENSIONALITY REDUCTION.

VISUALISING DATA VIA t-DISTRIBUTED STOCHASTIC NEIGHBOR EMBEDDING

- IN A NUTSHELL → t-SNE MODELLING DATA POINTS BASED ON PAIR-WISE DISTANCES IN HIGH-DIMENSIONAL (ORIGINAL) FEATURE SPACE.
  - ↳ THEN → t-SNE LEARNS TO EMBED DATA POINTS INTO A LOWER-DIMENSIONAL SPACE → PAIR-WISE DISTANCES IN ORIGINAL SPACE IS PRESERVED.
- t-SNE → INTENDED FOR VISUALISATION PURPOSE → REQUIRES WHOLE DATASET FOR PROJECTION
  - ↳ PROJECTS POINTS DIRECTLY → CANNOT APPLY t-SNE TO NEW DATA POINTS

- DIGITS → SCIKIT-LEARN
  - ↳ 64-DIMENSIONAL DATASET → 2-DIMENSIONAL SPACE.
  - ↳ FOR t-SNE EMBEDDING → USING PCA IS RECOMMENDED



- t-SNE ABLE TO SEPARATE DIFFERENT DIGITS (CLASSES) NICELY. (NOT PERFECT)

UNIFORM MANIFOLD APPROXIMATION AND PROJECTION

- ANOTHER POPULAR VISUALIZATION TECHNIQUE → (UMAP)
- FASTER THAN t-SNE + PRODUCES SIMILARLY GOOD RESULTS AS t-SNE + USED TO PROJECT NEW DATA.