Dimensionality Reduction

Week 4

Basic Concepts

Eigenvalues and Eigenvectors

Singular value decomposition (SVD)

Eigenvalues and Eigenvectors

- Eigenvalues and eigenvectors are prominently used in the analysis of linear transformations
- For a given square matrix A, if a number λ and a vector u satisfy the condition:
    - $Au = \lambda u$
- then λ is called an eigenvalue and u is the corresponding eigenvector of A
- Is it always possible, at least for certain λ and u, to have matrix multiplication be the same as just multiplying the vector by a constant?

Eigenvalues and Eigenvectors...

- For a matrix A of size $d \times d$, there are d eigenvectors and eigenvalue pairs.
  - It is only possible to have only k $(k \leq d)$ nonzero eigenvalues for A
  - The number of nonzero eigenvalues are equal to the rank of the matrix
- If $U = [u_1, u_2, \ldots, u_d]$ are the d eigenvectors of matrix A and $\lambda_1 \ldots \lambda_d$ are the corresponding eigenvalues, then we have

$$Au_1 = \lambda_1 u_1, \quad Au_2 = \lambda_2 u_2, \quad \ldots, \quad Au_d = \lambda_d u_d$$

Eigenvalues and Eigenvectors...

- Collectively,

$$AU = U \begin{bmatrix} \lambda_1 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & \dots & \lambda_d \end{bmatrix} = UD$$

- The matrix U is always orthogonal which means

$$u_i^T u_j = 0 \; if \; i \neq j \; and \; 1 \; otherwise$$

- It is obvious that $U^T = U^{-1}$ therefore we have: $A = UDU^T$
  - This is called Eigenvalue Decomposition of matrix A.
- The matrix U is an orthogonal matrix, called a full eigenvector matrix.
- The matrix U is always an orthogonal matrix, that rotates the coordinates in a way to de-correlate the data dimensions.

Eigenvalues and Eigenvectors...

- Finding Eigenvalues and Eigenvectors
  - Eigenvalues of a matrix A can be found by solving the characteristic polynomial in λ

$$Au = \lambda u$$
$$Au - \lambda u = 0$$
$$(A - \lambda I)u = 0$$

  - Since we have already know that $\vec{u} \neq 0$

$$det(A - \lambda I) = 0$$

  - The roots of the polynomial are the eigenvalues of the matrix A.

Eigenvalues and Eigenvectors...

- Finding Eigenvalues and Eigenvectors
  - Once all the eigenvalues are obtained, a eigenvector corresponding to a particular eigenvalue can be obtained by solving $Au_1 = \lambda_1 u_1$
  - Consider, $A = \begin{bmatrix} 2 & 2 \\ 5 & -1 \end{bmatrix}$
  - The first step is to find $det(A - \lambda I)$

$$det(A - \lambda I) = det\left(\begin{bmatrix} 2 & 2 \\ 5 & -1 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$$
$$= det\left(\begin{bmatrix} 2 & 2 \\ 5 & -1 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}\right) = \begin{vmatrix} 2-\lambda & 2 \\ 5 & -1-\lambda \end{vmatrix}$$
$$= (2-\lambda)(-1-\lambda) - 10 = \lambda^2 - \lambda - 12$$

The eigenvalues of A are the solutions of the quadratic equation

$\lambda^2 - \lambda - 12 = 0$, namely λ_1 = –3 and λ_2 = 4.

Eigenvalues and Eigenvectors...

- The next step , finding eigenvectors with $\lambda = -3, Au = -3u$
- Assume $u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ $Au = \begin{bmatrix} 2 & 2 \\ 5 & -1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 2u_1 + 2u_2 \\ 5u_1 - u_2 \end{bmatrix}$ $-3u = \begin{bmatrix} -3u_1 \\ -3u_2 \end{bmatrix}$
- Set them equal $\begin{bmatrix} 2u_1 + 2u_2 \\ 5u_1 - u_2 \end{bmatrix} = \begin{bmatrix} -3u_1 \\ -3u_2 \end{bmatrix}$ $\begin{cases} 5u_1 = -2u_2 \\ u_1 = -\frac{2}{5}u_2 \end{cases}$,
- This means that, while there are infinitely many nonzero solutions (solution vectors) of the equation $Au = -3u$
  - They all satisfy the condition that the first entry u_1 is −2/5 times the second entry u_2

Eigenvalues and Eigenvectors...

- Thus all solutions of this equation can be characterized by: $\begin{bmatrix} 2t \\ -5t \end{bmatrix} = t \begin{bmatrix} 2 \\ -5 \end{bmatrix}$, where t is any real number.

- The nonzero vectors u that satisfy $Au = -3u$ are called eigenvectors associated with the eigenvalue $\lambda = -3$.
- One such eigenvector is $u_1 = \begin{bmatrix} 2 \\ -5 \end{bmatrix}$
- Similarly, we can find eigenvectors associated with the eigenvalue $\lambda = 4$ by solving $Au = 4u$: $\begin{bmatrix} 2u_1 + 2u_2 \\ 5u_1 - u_2 \end{bmatrix} = \begin{bmatrix} 4u_1 \\ 4u_2 \end{bmatrix}$

Singular value decomposition (SVD)

- SVD is a method of decomposing a matrix into three other matrices: $X = USV^T$
  - Where, X is a $n \times d$ matrix
  - U is a $n \times d$ orthogonal matrix (same as U in previous section)
  - S is a $d \times d$ diagonal matrix with elements $S(i,i) = \sigma_i$
  - V is a $d \times d$ orthogonal matrix
- In linear algebra, the SVD is a factorization of a real or complex matrix
- The SVD represents an expansion of the original data in a coordinate system where the covariance matrix is diagonal.

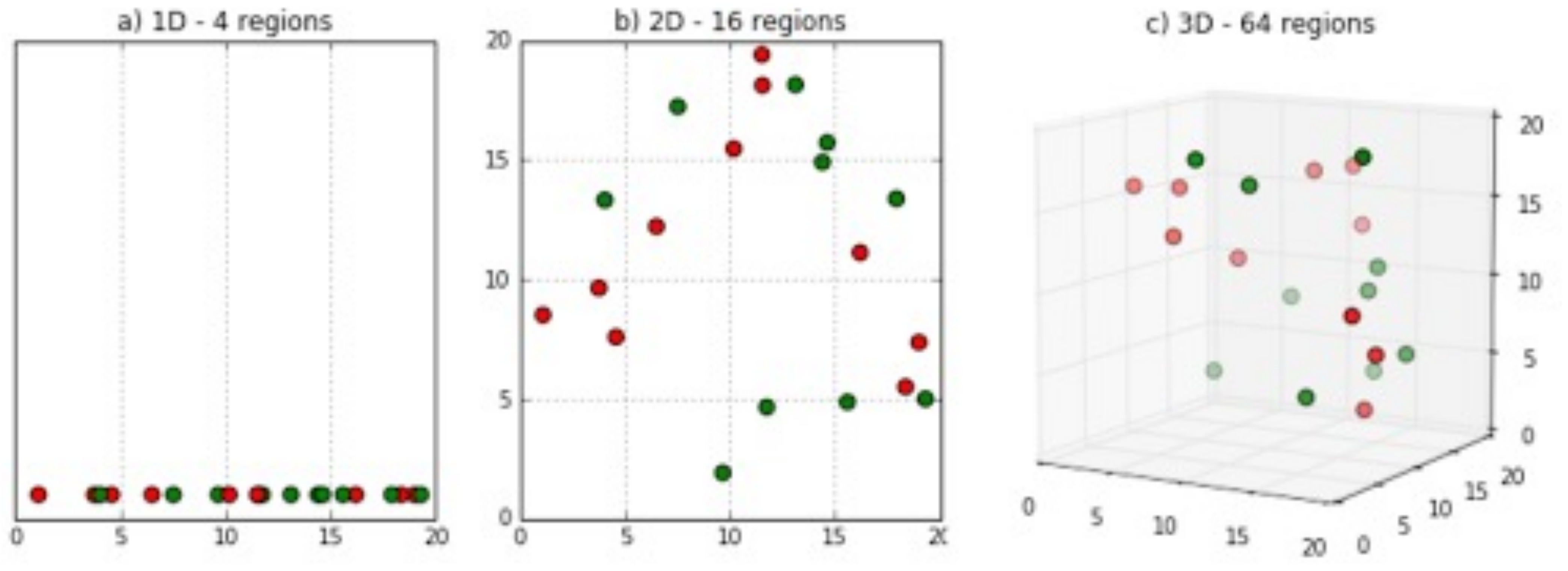
Singular value decomposition (SVD)

- Calculating the SVD consists of finding:
  - The eigenvalues and eigenvectors of XX^T *and* X^TX
  - The eigenvectors of X^TX make up the columns of V
  - The eigenvectors of XX^T make up the columns of U
  - The singular values in S are square roots of eigenvalues from XX^T *or* X^TX

- The singular values are the diagonal entries of the S matrix and are arranged in descending order.
- The singular values are always real numbers.
- If the matrix XX^T *or* X^TX is a real matrix, then U and V are also real.

Typical Dimensionality in Data

- Typical dimensions of data:
  - **[Text data]**
    - If you crawl a news website to have one week's news, DIM>10000
    - It is dictionary size you have built based on the words (We need to represent each document based on the words in a dictionary)
  - **[Image data]**:
    - 64×64 image would have 4096 dimensions!
  - **[Genomic data]**:
    - Parkinson case-control data has 408,803 Single-nucleotide polymorphisms (SNPs), & Alzheimer has 380,157 SNPs.



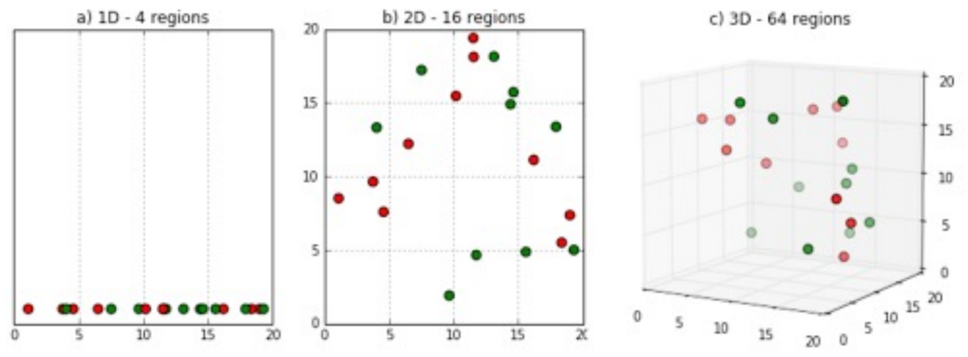
Curse of Dimensionality

- What happens with increasing dimensionality?
  - Volume of the space increases
  - Data become sparse

Curse of Dimensionality...

- In the figure, first we had observed some points in 1D data in 4 regions (20 divided by 5).
- Subsequently these points are transferred into 2D space, into a 16 regions.
- In the next step the points are in a 3D space with 64 regions.

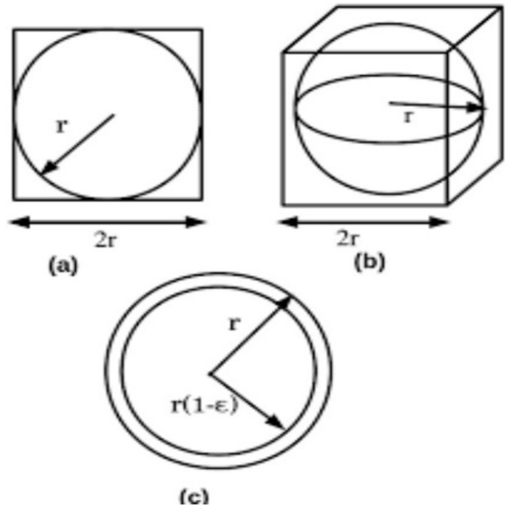
- What would happen when we get to 100 dimensions?
  - The curse of dimensionality dictates that as the number of dimensions increases, the number of regions grows exponentially.
  - That makes our data sparse and somehow not useful anymore
  - Some of our intuitions from low dimensional spaces fail badly in high dimensions.



Curse of Dimensionality...

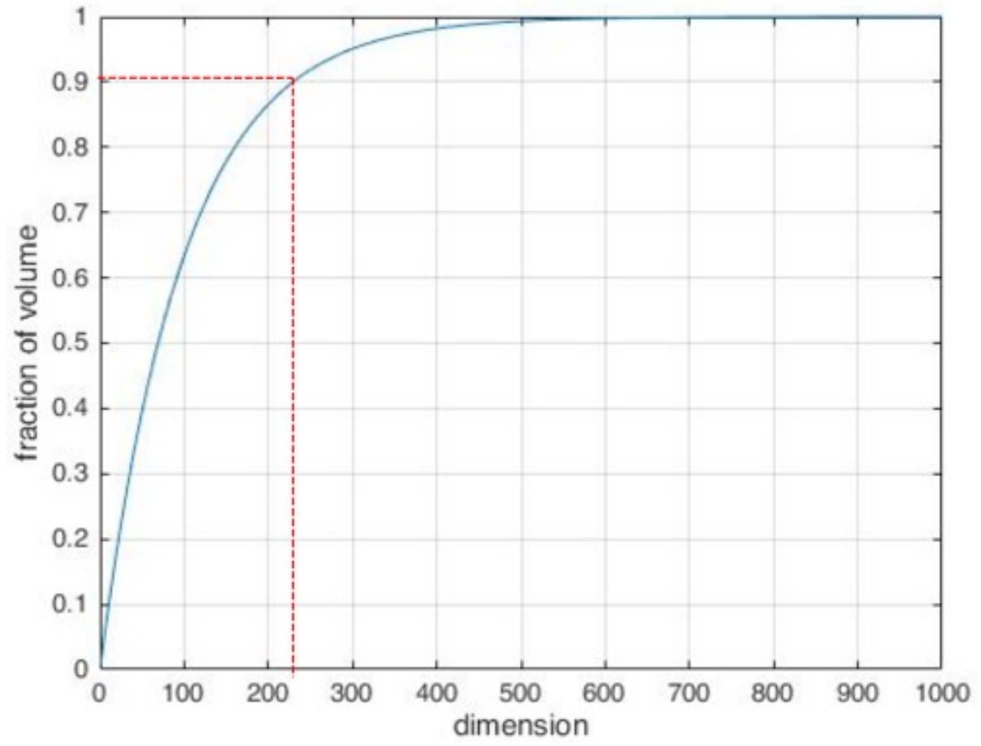
- Think of N data points spread uniformly in a sphere of radius r in d-dimension.
- The volume of the hypersphere can be calculated as $v_d(r) = k_d r^d$
- Now, if we think a data point at $\mathrm{r} = 1 - \epsilon$ then the volume of that hypersphere can be defined by $v_d(1-\varepsilon) = k_d(1-\epsilon)^d$
- The fraction of volume that is between $\mathrm{r} = 1$ and $\mathrm{r} = 1 - \epsilon$ can be computed as:

$$\frac{v_d(1) - v_d(1-\epsilon)}{v_d(1)} = \frac{k_d(1)^d - k_d(1-\epsilon)^d}{k_d(1)^d} = \frac{k_d - k_d(1-\epsilon)^d}{k_d}$$
$$= \frac{k_d(1-(1-\epsilon)^d)}{k_d} = 1-(1-\epsilon)^d$$



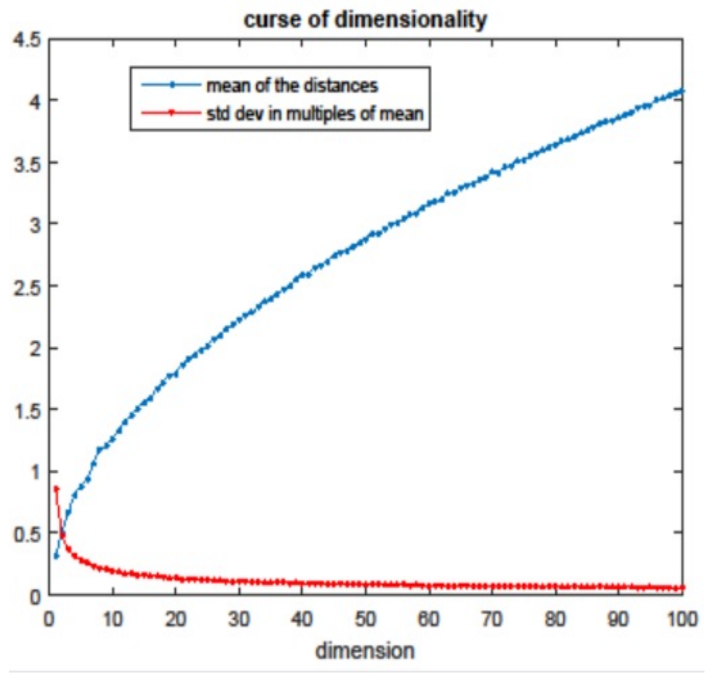
Curse of Dimensionality...

- Now for any fixed value of ϵ, which is fairly small constant, if we keep increasing the dimension d then the fraction volume $1 - (1-\epsilon)^d$ will have an exponential curve like the figure;
- This shows that the volume of the hypersphere with radius $\mathrm{r} = 1 - \epsilon$ tends to zero
- The fraction of volume tends to grow to 100% of the volume as the dimensionality tends to infinity, whereas the volume of the surrounding hypercube i.e., $\mathrm{r} = 1$ remains constant. Counterintuitive!!!!



Curse of Dimensionality...

- In high dimensional spaces:
  - most of the training data resides on the surfaces of the hypersphere or corner of the hypercube.
- The distance from origin to all different points become similar
  - the curse of dimensionality result in less distinctive distances in high dimensions.
- So given a point in high dimensions
  - the relative distance between points far from it and close from it, becomes negligible.
- As we can see in the figure, with increasing dimensionality variances among mean of distances decreases.



Concentration Effect

- Let's a point vector denoted by $||X_d||$ and the mean point vector $\mathrm{E}[||X_d||]$
- Now assume that variance of ratio of them converges to zero with increasing data dimensionality i.e.,

$$lim_{d \to \infty} var\left(\frac{||X_d||}{E[||X_d||]}\right) = 0$$

- Then we can say that the proportional difference between the farthest-point distance D_{max} and the closest-point distance D_{min} (the relative contrast) vanishes i.e., $\frac{D_{max} - D_{min}}{D_{min}} \to 0$
- Reduces the utility of the measure to discriminate between near and far neighbours.

Curse of Dimensionality...

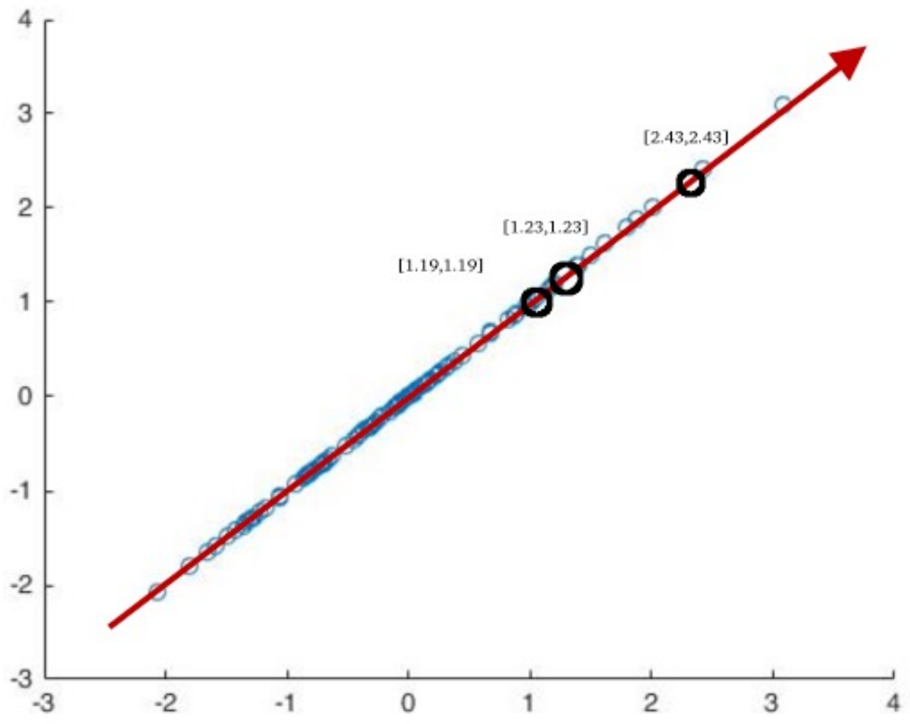
- Concentration effect
  - Relative contrast between near and far neighbours diminishes as the dimensionality increases.
- This problem can imply that
  - Clustering or KNN algorithms may be meaningless in high dimensions.
  - However, there might still be patterns in high dimension. We just need better distance metrics. Research is on!
  - Until we develop better distance metrics, we should aim to reduce the dimensionality where possible

Solving Curse of Dimensionality

- In some problems, there are too many variables.
  - Are all variables important?
    - If not then
      - some of them are irrelevant
      - can be removed
- If all variables are numeric, what if they are correlated?
  - This means redundancy!
  - Can we club them together?

Solving Curse of Dimensionality

- Dimensionality reduction refers to the process of converting a dataset of dimension Q into dimension R where $R < Q$ ensuring similar information contents.



- We can take a subset of data from this graph, which looks like this:

$$X = \begin{bmatrix} 1.19 & 1.19 \\ 1.23 & 1.23 \\ 2.43 & 2.43 \end{bmatrix}$$

- is the first and second features of these data points are the same?
  - why not just to use only one of these features?

Solving Curse of Dimensionality

$$X = \begin{bmatrix} 1.19 & 1.19 \\ 1.23 & 1.23 \\ 2.43 & 2.43 \end{bmatrix}$$

So, we can transform this data points as the only dimension by using a projection vector:

$$\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

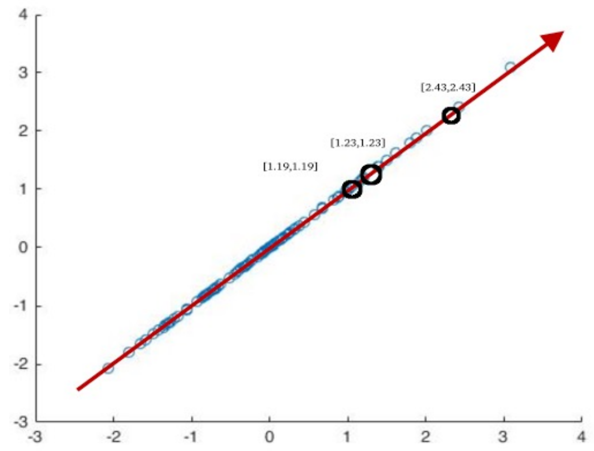
- By multiplying data points and projection vector we will have:

$$\begin{bmatrix} 1.19, 1.19 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} = 1.19$$

$$\begin{bmatrix} 1.23, 1.23 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} = 1.23$$

$$\begin{bmatrix} 2.43, 2.43 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} = 2.43$$

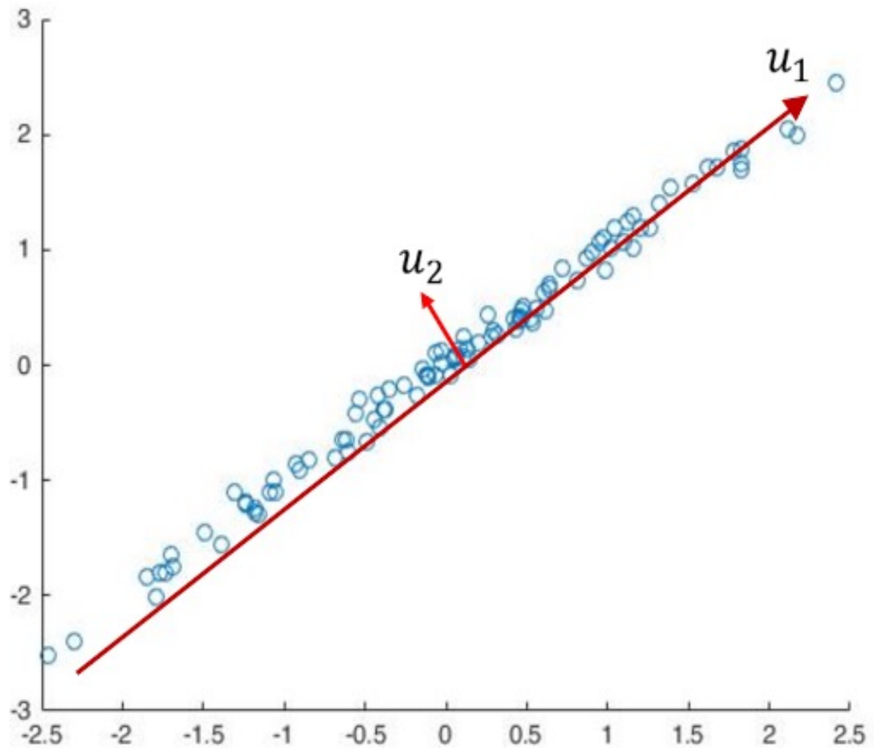
$$\Rightarrow X' = \begin{bmatrix} 1.19 \\ 1.23 \\ 2.43 \end{bmatrix}$$



- X' is the projected data into a single dimension (the red arrow in the figure).
- We have just reduced one dimension of this 2D data.
- If you notice the formation of the data, you can see that is also the direction of maximum variance in data!

Solving Curse of Dimensionality

- What if the data is not exactly on the red arrow? Consider this example.
- As you can see in the figure:
  - The u_1 dimension vector, points towards the direction of the highest variance
  - The u_2 dimension vector, points towards the lowest variance in the subspace, orthogonal to the u_1 vector
- Thus, projecting onto maximum variance direction (u_1) means capturing more variance and results in capturing more information to analyse.



Principal Component Analysis (PCA)

Preliminaries

Formulation

Implementation

Preliminaries

- The goal of PCA is to take n data points in d dimensions, which may be correlated, and summarizes them by a new set of uncorrelated axes.
  - The uncorrelated axes are called principal components or principal axes.
  - These axes are linear combinations of the original d dimensions.
  - Principal components are sorted in descending order based on captured variance along each axis.

Formulation of PCA

- Let us say that we first project the data on a new axis, whose direction is specified by a $d-$dimensional vector u_1
- Since we are only interested in direction of maximum variance, we assume u1 to be a unit length vector, i.e. $\|u_1\| = 1,\ or\ u_1^T u_1 = 1$
- Now each data point x_i can be projected on the vector u_1 to create a new co-ordinate as $y_{i1} = u_1^T x_i$
- So the variance of the data projected on u_1 is:

$$\left(\frac{1}{n-1}\right)\sum_{i=1}^{n}(u_1^T x_i - u_1^T \bar{x})^2$$

Formulation of PCA

- The mean of the new data is

$$\bar{y} = u_1^T \bar{x}$$

- and the variance is:

$$\left(\frac{1}{n-1}\right)\sum_{i=1}^{n}(u_1^T x_i - u_1^T \bar{x})^2$$
$$= u_1^T \left[\left(\frac{1}{n-1}\right)\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T\right] u_1 = u_1^T C u_1$$

- Now as we know we would like to find out the direction so that the variance $u_1^T C u_1$ is maximized.

Formulation of PCA

- Recall that we also assume $u_1^T u_1 = 1$
- By putting it together we want to find: $\begin{cases} \max_x \; u_1^T C u_1 \\ s.t. \quad u_1^T u_1 = 1 \end{cases}$
- For solving this problem we introduce Lagrange multiplier and change the problem into an unconstrained maximization problem: $\max\limits_x \; u_1^T C u_1 + \lambda_1 (1 - u_1^T u_1)$
- If you want to find maximums or minimums a good way to get started is to find out where the slope of the function (derivative) is equal to zero.
- Taking derivative w.r.t. u_1 and setting it to zero we obtain:

$$C u_1 = \lambda_1 u_1$$

Formulation of PCA

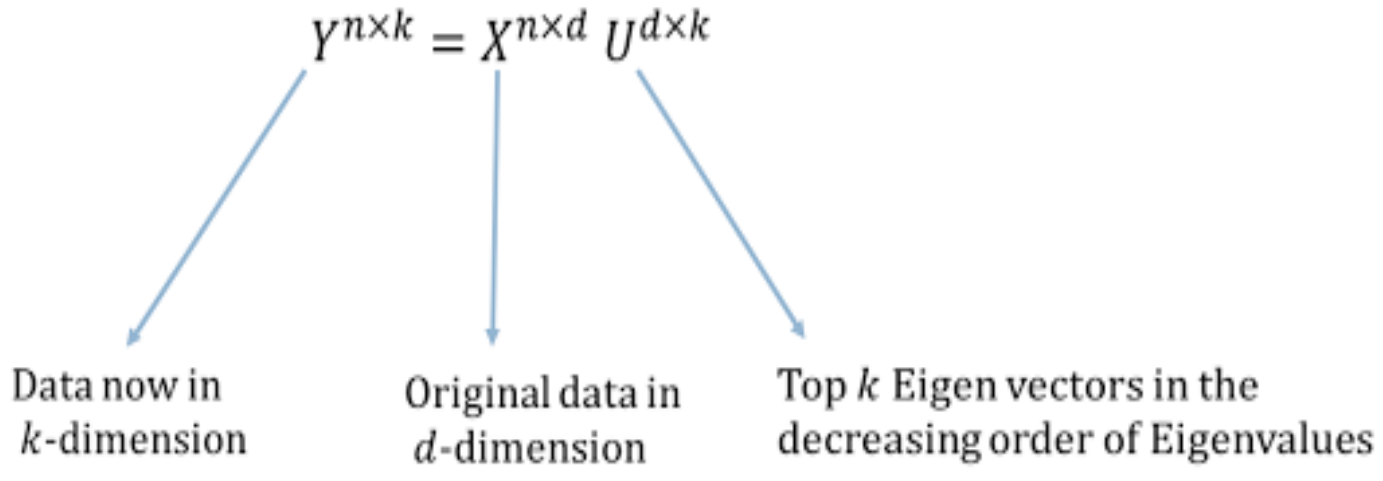
- This is an eigenvalue problem, where λ_1 is the largest eigenvalue of C and u_1 is the corresponding eigenvector.
- u_1 is known as the first principal component.
- Now what about $u_2, \dots, u_d$?
- Next set of axes $u_2, \dots, u_d$ can be found incrementally by finding a direction that maximizes the variance and is orthogonal to all the principal axes found so far.
- The directions have to be orthogonal since we want them to be uncorrelated
- Therefore, the principal axes can be collectively written using the Eigenvector matrix $U = [u_1, u_2, \dots, u_d]$ in the order of decreasing eigenvalues of the covariance matrix C

Formulation of PCA

- The question which arises here is what if we project the data using all d principal components?
  - Well in this case we just doing de-correlation but no dimensionality reduction.
  - However, if we project data on only top k principal components such that $k \leq d$
    - we achieve dimensionality reduction
    - each new dimension is also uncorrelated of other dimensions.

Formulation of PCA

- PCA via Eigen Value Decomposition:
  - Compute data covariance matrix C
  - Perform Eigen value decomposition (EVD) as $C = UDU^T$
  - Reduced dimension data is given by:



$$Y^{n \times k} = X^{n \times d}\, U^{d \times k}$$

Data now in k-dimension
Original data in d-dimension
Top k Eigen vectors in the decreasing order of Eigenvalues

Formulation of PCA

- PCA: Minimum Error Formulation
  - This is an alternative formulation of PCA based on projection error minimisation
  - Suppose we project our data on k dimensions from d dimensions
  - Obviously losses incurred due to losing some features in data ($k < d$).
  - But the error we have while using PCA's best k dimensions in terms of least square error, is the minimum possible error that we can have.

Formulation of PCA

- PCA: Minimum Error Formulation
  - Let us consider a set of new axes $u_1,\ldots, u_d$ in such a way that they are mutually orthogonal, i.e.,

$$u_i^T u_j = 1 \; if \; i = j \; otherwise \; 0$$

  - Project a point such as x_i on $u_1,\ldots, u_d$ to get new coordinates as $y_{ij} = x_i^T u_j$
  - So for all d dimensions we can write this as: $x_i = \sum_{j=1}^{d} y_{ij} u_j$

$$x_i = \sum_{j=1}^{k} y_{ij} u_j + \sum_{j=k+1}^{d} y_{ij} u_j$$

  - If we would like to minimise the mean square error due to projection in new k dimension, we have:

$$\min_{u_1,\ldots,u_k} \frac{1}{n} \sum_{i=1}^{n} ||x_i - \sum_{j=1}^{k} y_{ij} u_j||^2$$

Implementation of PCA

- PCA for data where $n < d$
  - There are cases when the number of data points (n) is less that number of dimensions d
  - say we have 100 images in 64×64 dimensions, n=100 and d=64×64=4096
  - In this case, the number of nonzero eigenvalues of data covariance matrix is less than or equal to n
  - If we use Eigen Value Decomposition (EVD) on the covariance matrix of size $d \times d$, we need to perform computations of the order of $O(d^3)$
    - This may be too expensive!
  - In such cases, SVD can reduce the computations to $O(n^3)$ or less.

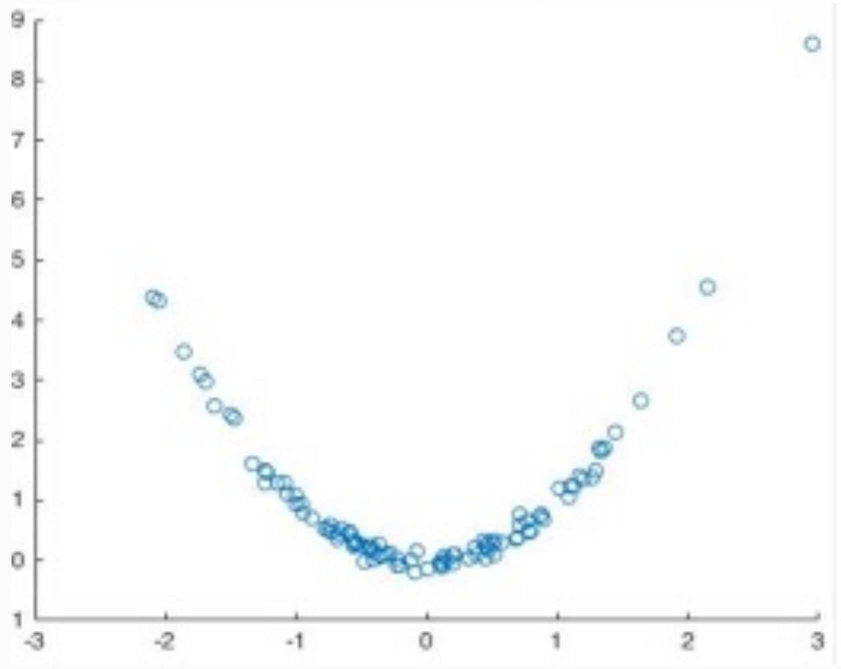
Implementation of PCA

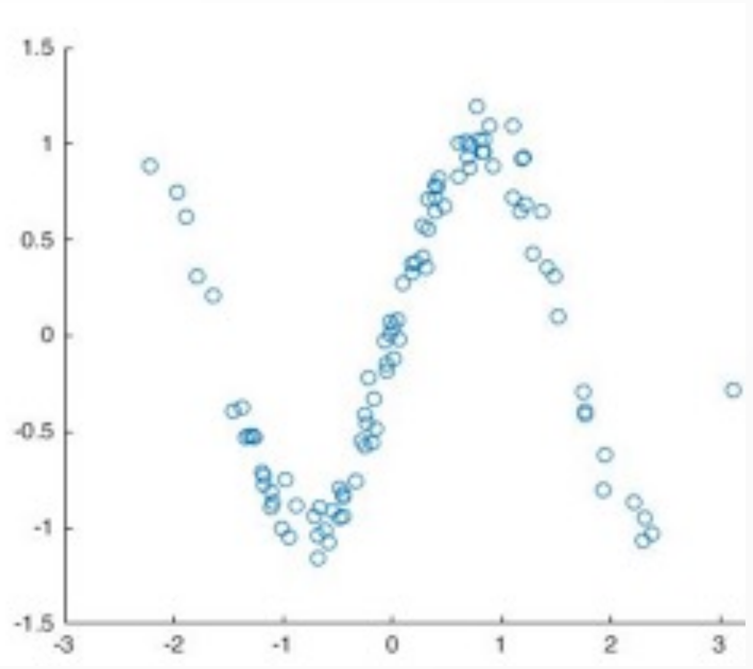
- Using SVD for PCA
  - given any $n{\times}d$ matrix Y, its Singular Value Decomposition (SVD) is given as $$X = USV^T$$
    - where, U is a $n{\times}d$ orthogonal matrix (same as U in previous section)
    - S is a $d{\times}d$ diagonal matrix with elements $S(i,i) = \sigma_i$
    - V is a $d{\times}d$ orthogonal matrix
  - Now if Y is mean-centred version of X then the covariance of Y is: $$(n-1)C = YY^T = US(V^TV)SU^T = USISU^T = US^2U^T$$
  - Remember that $V^TV = I$, therefore: $C = U(\frac{s^2}{n-1})U^T = UDU^T$

Implementation of PCA

- Using SVD for PCA
  - Also U of SVD is same as U of EVD.
  - Therefore, the singular vectors of SVD are the same as Eigenvectors of EVD and $D = \frac{s^2}{n-1}$
  - We have the relation $\lambda_d = \frac{\sigma_d^2}{n-1}$
  - So if you do not want to use EVD you can just use SVD and get the matrix S or get the singular values and then compute the eigenvalues $\lambda_d = \frac{\sigma_d^2}{n-1}$
  - This gives of the things we need to perform PCA.
    - Remember performing PCA is nothing but multiplying U matrix to data matrix.

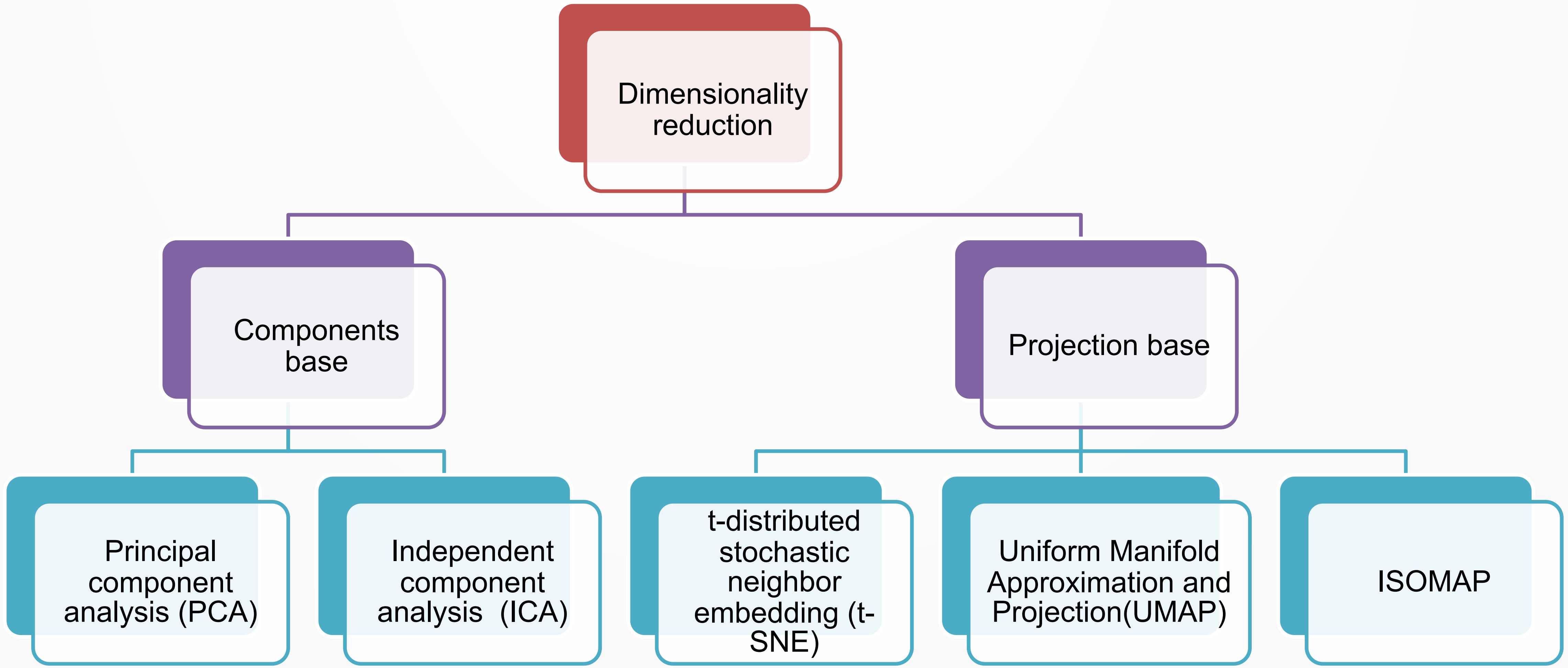
Examples of nonlinear relationship among variables





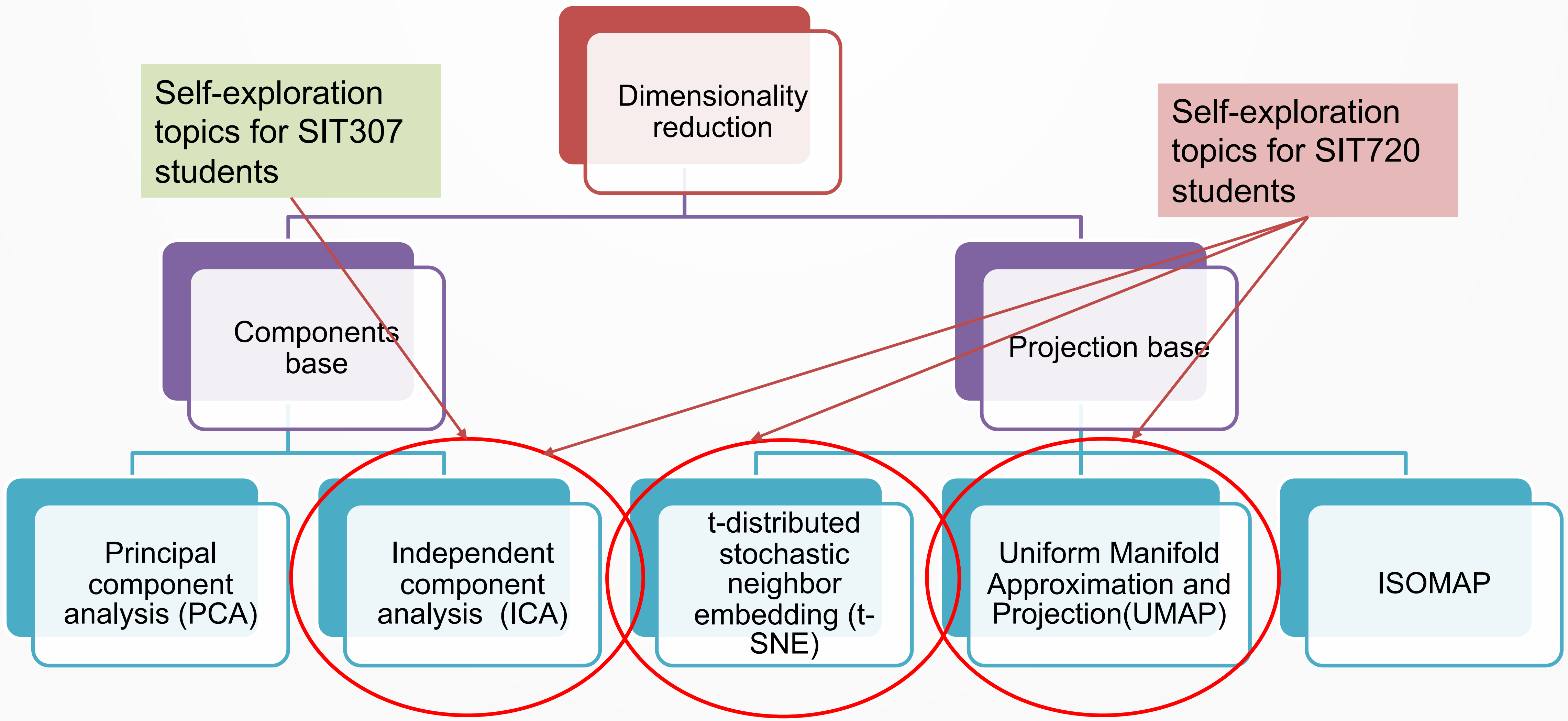
Is this alright to use PCA in this kind of scenarios?

Components vs projection-based Dimensionality reduction


Dimensionality reduction
Components base
Projection base
Principal component analysis (PCA)
Independent component analysis (ICA)
t-distributed stochastic neighbor embedding (t-SNE)
Uniform Manifold Approximation and Projection(UMAP)
ISOMAP

Components vs projection-based Dimensionality reduction

Please visit unit site for resources and references


Self-exploration topics for SIT307 students
Dimensionality reduction
Self-exploration topics for SIT720 students
Components base
Projection base
Principal component analysis (PCA)
Independent component analysis (ICA)
t-distributed stochastic neighbor embedding (t-SNE)
Uniform Manifold Approximation and Projection(UMAP)
ISOMAP

Thank You.