# Foundations of Machine Learning (ECE 5984)

## - Dimensionality Reduction -

Eunbyung Park

Assistant Professor

School of Electronic and Electrical Engineering

Eunbyung Park (silverbottlep.github.io)

# Eigenvalues and Eigenvectors

# Matrix Decomposition

- We can decompose an integer into its prime factors
  - 12 = 2 x 2 x 3.
- Similarly, matrices can be decomposed into products of other matrices
- Eigendecomposigion, SVD, LU decomposition, …

# Eigenvector

- An eigenvector of a square matrix $A \in \mathbb{R}^{n\times n}$ is a nonzero vector $v$ such that

$$Av = \lambda v$$

, where the scalar $\lambda$ is the eigenvalue

- If $v$ is an eigenvector of $A$ with an eigenvalue $\lambda$, then any rescaled $\alpha v$ is also an eigenvectors
- So, usually, we find the *'normalized eigenvectors'*

# Compute Eigenvalues

$$Av = \lambda v$$

$$Av - \lambda v = 0$$

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

- If nonzero solution for $v$ exists, then $(A - \lambda I)$ should be “non-invertible”.

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

- A.k.a, characteristic polynomial

# Exercise

- What are the eigenvalues and eigenvectors of $A$?

$$A = \begin{bmatrix} -3 & 5 \\ 4 & -2 \end{bmatrix}$$

# How Many Distinct Eigenvalues?

- An eigenvector of a square matrix $A \in \mathbb{R}^{n \times n}$ is a nonzero vector $v$ such that

$$Av = \lambda v$$

, where the scalar $\lambda$ is the eigenvalue

# How Many Distinct Eigenvalues?

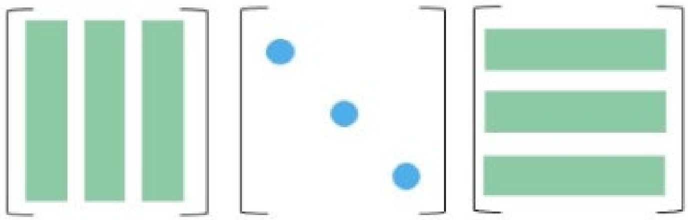
- An eigenvector of a square matrix $A \in \mathbb{R}^{n \times n}$ is a nonzero vector $v$ such that

$$Av = \lambda v$$

, where the scalar $\lambda$ is the eigenvalue

- There can be maximum distinct 'n' eigenvalues.
- The eigenvalues of an n by n matrix are the roots of a polynomial of degree n. So there are n eigenvalues, though some of them may be repeated.

$$AQ = Q\Lambda \qquad A = Q\Lambda Q^{-1}$$



# Symmetric Eigendecomposition

- If $A$ is a symmetric (also square) matrix, then
- All the eigenvalues are real
- The eigenvectors corresponding to different eigenvalues are orthogonal
- If we normalize all eigenvectors, then

$$QQ^{\top} = ?$$

$$AQ = Q\Lambda \qquad A = Q\Lambda Q^{-1}$$

# Symmetric Eigendecomposition

- If $A$ is a symmetric (also square) matrix, then
- All the eigenvalues are real
- The eigenvectors corresponding to different eigenvalues are orthogonal
- If we normalize all eigenvectors, then

$$QQ^{\top} = I$$

$$AQ = Q\Lambda \qquad A = Q\Lambda Q^{-1} \qquad A = Q\Lambda Q^{\top}$$

# Symmetric Decomposition

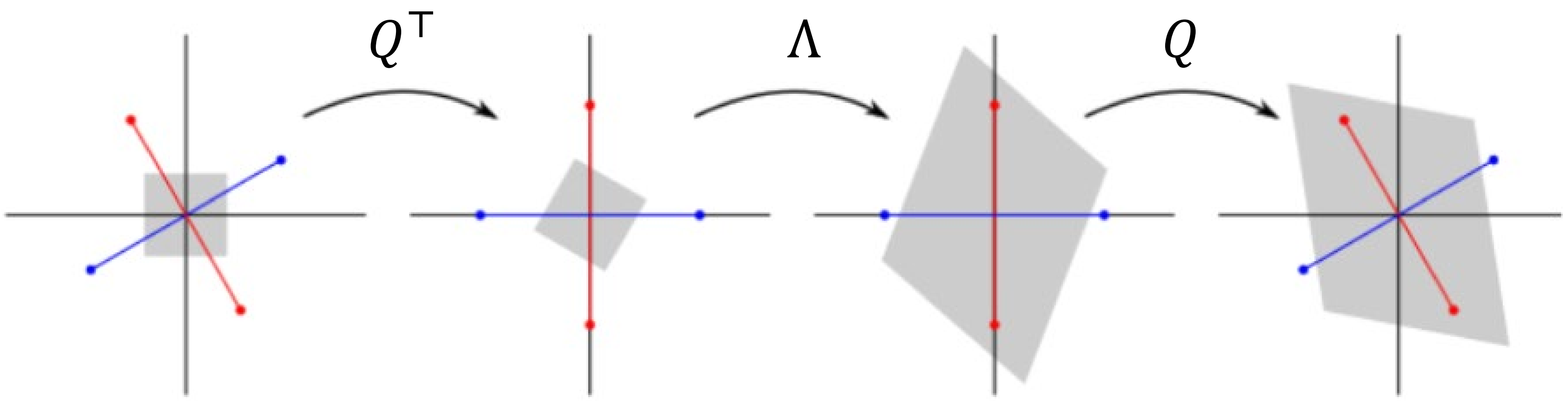
$$A = Q\Lambda Q^{-1} = Q\Lambda Q^{\top}$$

$$\underset{\mathrm{Q}}{\begin{array}{cccc} | & | & & | \\ u_1 & u_2 & \cdots & u_d \\ | & | & & | \end{array}} \qquad \begin{array}{cccc} \lambda_1 & & & \\ & \lambda_2 & & \mathbf{0} \\ & & \ddots & \\ \mathbf{0} & & & \lambda_d \end{array} \qquad \underset{\mathrm{Q}^{\top}}{\begin{array}{c} — \; u_1^{\top} \; — \\ — \; u_1^{\top} \; — \\ \vdots \\ — \; u_d^{\top} \; — \end{array}}$$

# Geometric Interpretation of Eigendecomposition

- Matrix is all about linear transformation!
- Orthogonal matrices ≈ Rotational matrices
- $Ax$ → scale and rotate the vector $x$

$$Ax = Q\Lambda Q^\top x$$

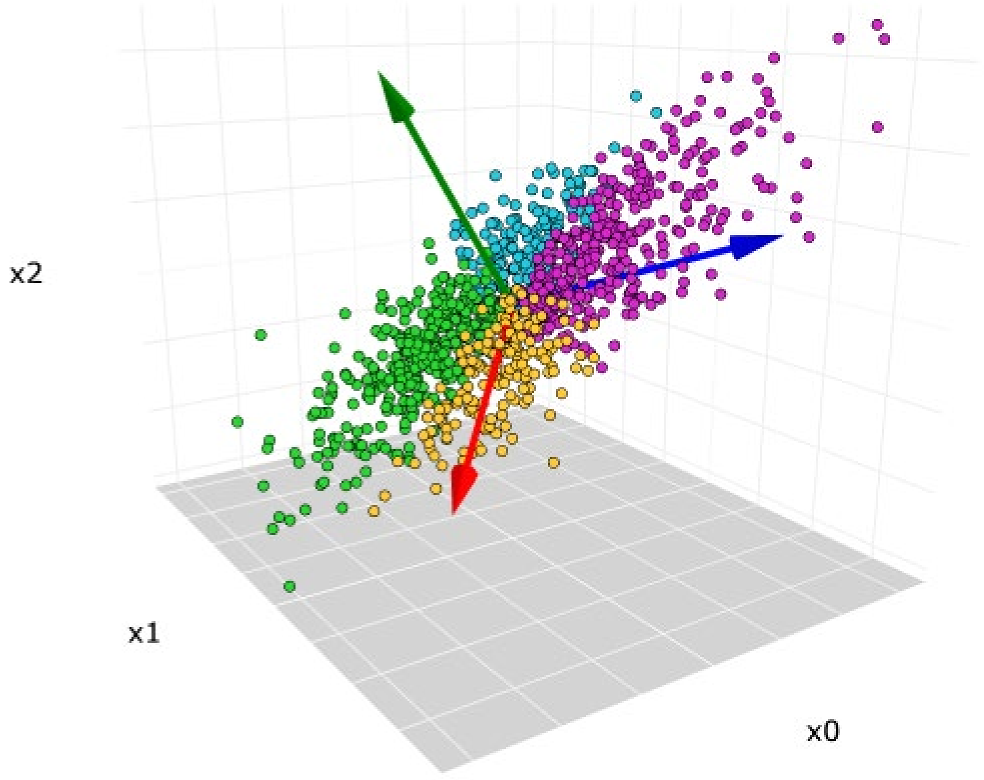



[linear algebra - Geometric Interpretation of Eigendecomposition - Mathematics Stack Exchange]

# Principle Component Analysis (PCA)

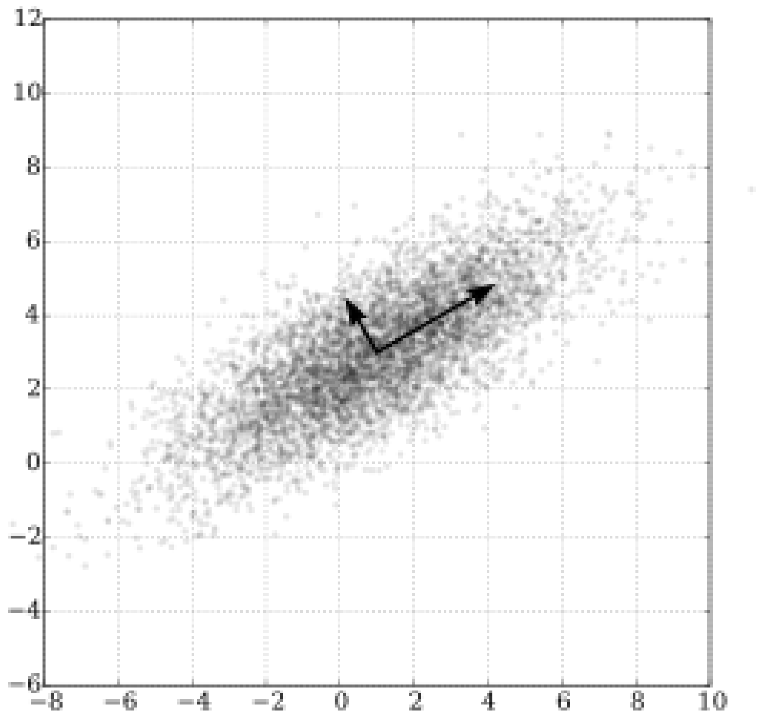
# Dimensionality Reduction

- Redundant features
  - E.g., mph, kph
- Correlation between features
  - E.g., enjoying study, grade, skill



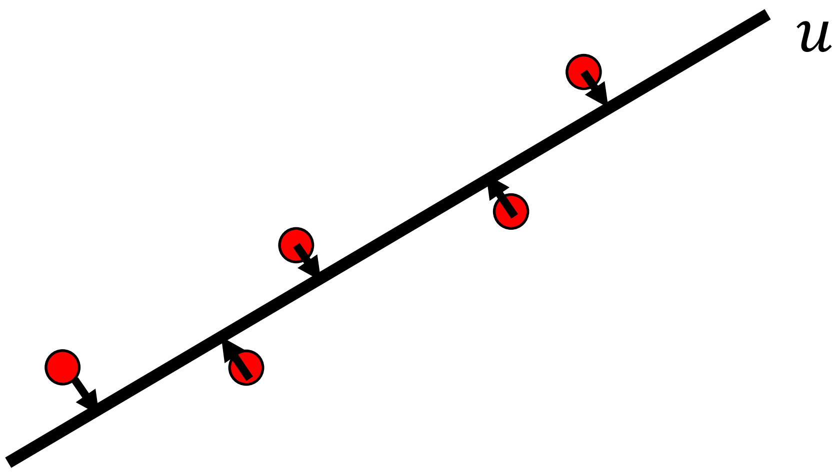
# Principal Component Analysis (PCA)

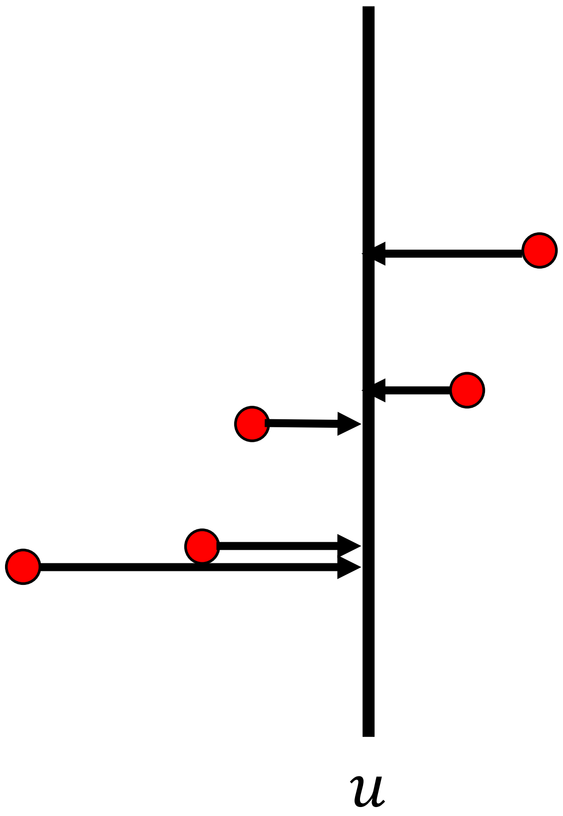
- Finding 'principal' component that explains the data



# Maximizing The Variance

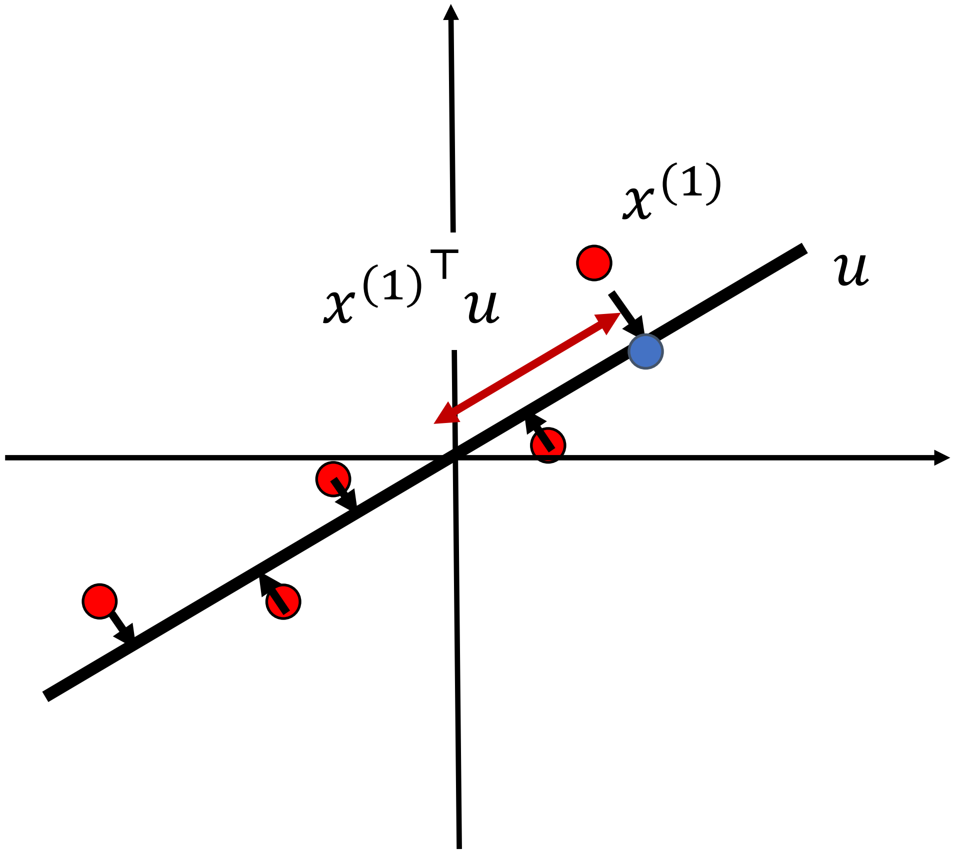
- Finding unit vector u, after data projection, the variance of the projected data is maximized



# Maximizing The Variance

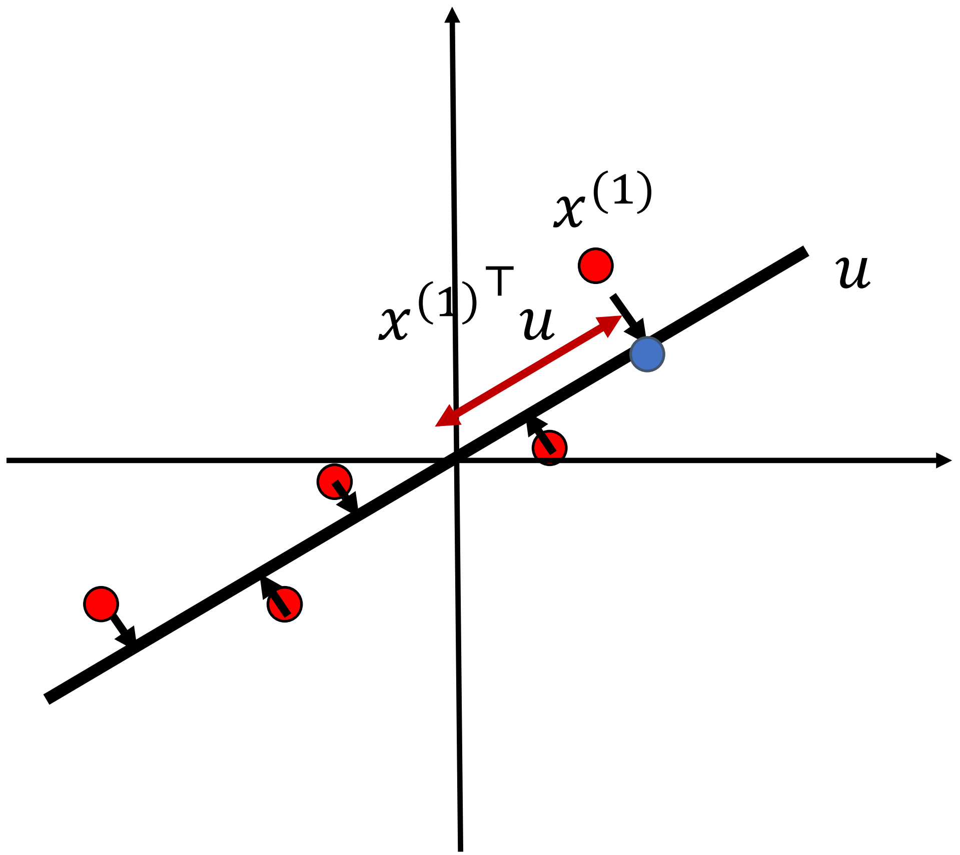
- Finding unit vector u, after data projection, the variance of the projected data is maximized




${x^{(i)}}^{\top} u$: The length of the projection of $x^{(i)}$ onto $u$

# Maximizing The Variance

- Finding unit vector u, after data projection, the variance of the projected data is maximized




$$\frac{1}{m}\sum_{i=1}^{m}\left({x^{(i)}}^\top u\right)^2 = \frac{1}{m}\sum_{i=1}^{m} u^\top x^{(i)} {x^{(i)}}^\top u$$

$$= u^\top \left(\frac{1}{m}\sum_{i=1}^{m} x^{(i)} {x^{(i)}}^\top\right) u \qquad s.t.\ \|u\|_2 = 1$$

Assuming the data is normalized, zero mean

# Optimization

- How to optimize it?

$$\max_u u^\top \Sigma u \qquad s.t. \|u\|_2 = 1$$

Constraint -> unconstraint

Lagrangian, take the derivative, set it to zero!

# Optimization

- How to optimize it?

$$\max_{u} u^{\top}\Sigma u \qquad\qquad s.t.\ \|u\|_2 = 1$$

Constraint -> unconstraint

Lagrangian, take the derivative, set it to zero!

$$L(u, \lambda) = u^{\top}\Sigma u - \lambda(u^{\top}u - 1)$$

# Optimization

- How to optimize it?

$$L(u, \lambda) = u^\top \Sigma u - \lambda(u^\top u - 1)$$

$$\frac{\partial L}{\partial u} = \Sigma u - \lambda u = 0$$

$$\Sigma u = \lambda u$$

# PCA and Eigenvector

Covariance matrix

Eigenvalue

$$\Sigma u = \lambda u$$

Eigenvector

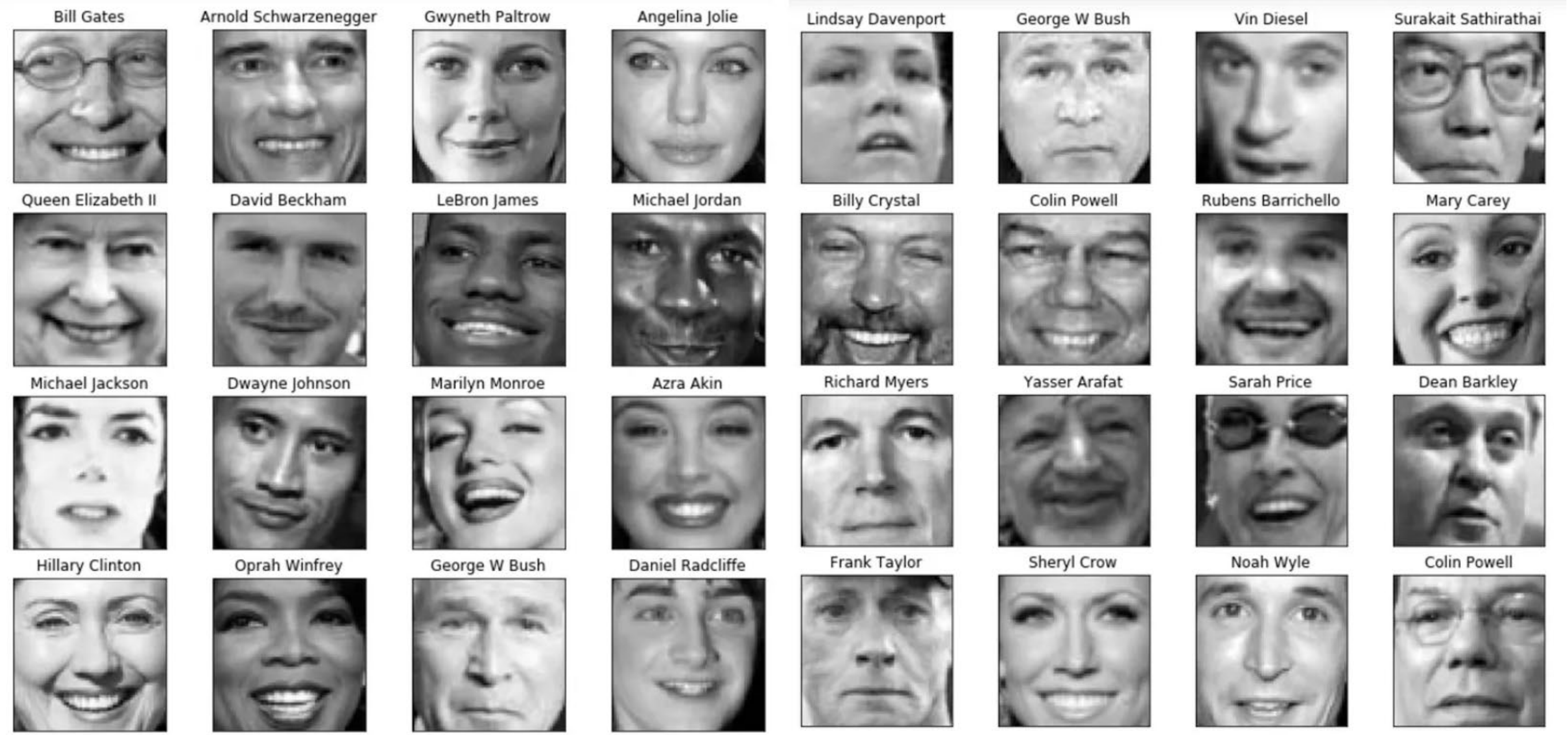
# Projected Coordinates

$$y^{(i)} = \begin{bmatrix} u_1^\top x^{(i)} \\ u_2^\top x^{(i)} \\ \dots \\ u_k^\top x^{(i)} \end{bmatrix} \in \mathbb{R}^k$$

The new coordinate, using top-k principal component

# Eigenface

- Dataset (1000 x 64 x 64 -> 1000 x 4096)




[Eigenfaces: Recovering Humans from Ghosts | by Nev Acar | Towards Data Science]

# Eigenface

- Dataset (1000 x 64 x 64 -> 1000 x 4096)

$$\Sigma u = \lambda u$$

# Eigenface

- Eigenvectors

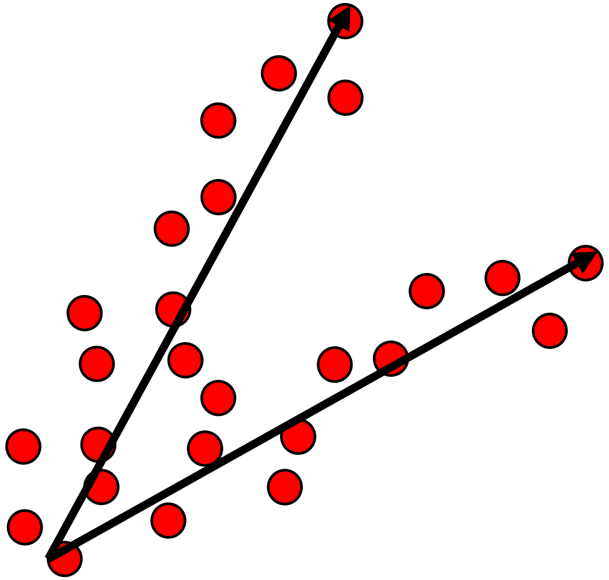

[Eigenfaces: Recovering Humans from Ghosts | by Nev Acar | Towards Data Science]

# Code Demo

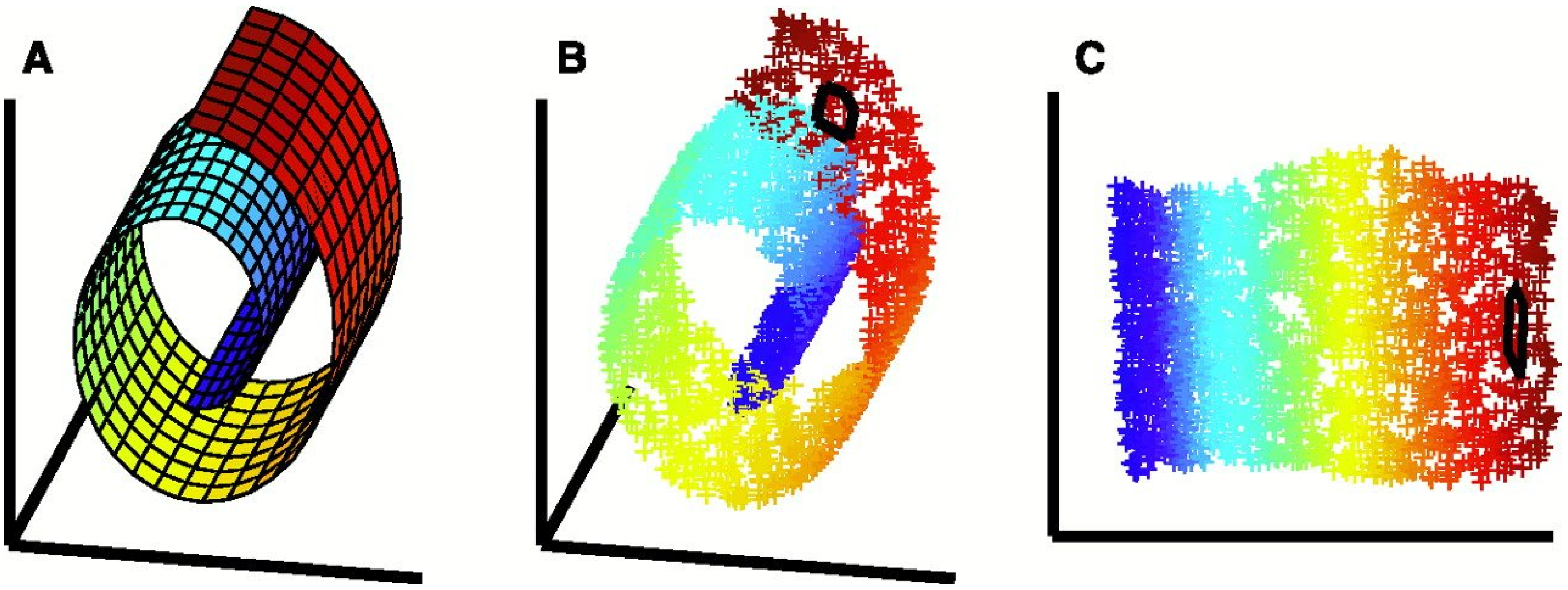
05.09-Principal-Component-Analysis.ipynb - Colaboratory (google.com)

# Nonlinear Dimensionality Reduction

# Orthogonal Assumption of PCA

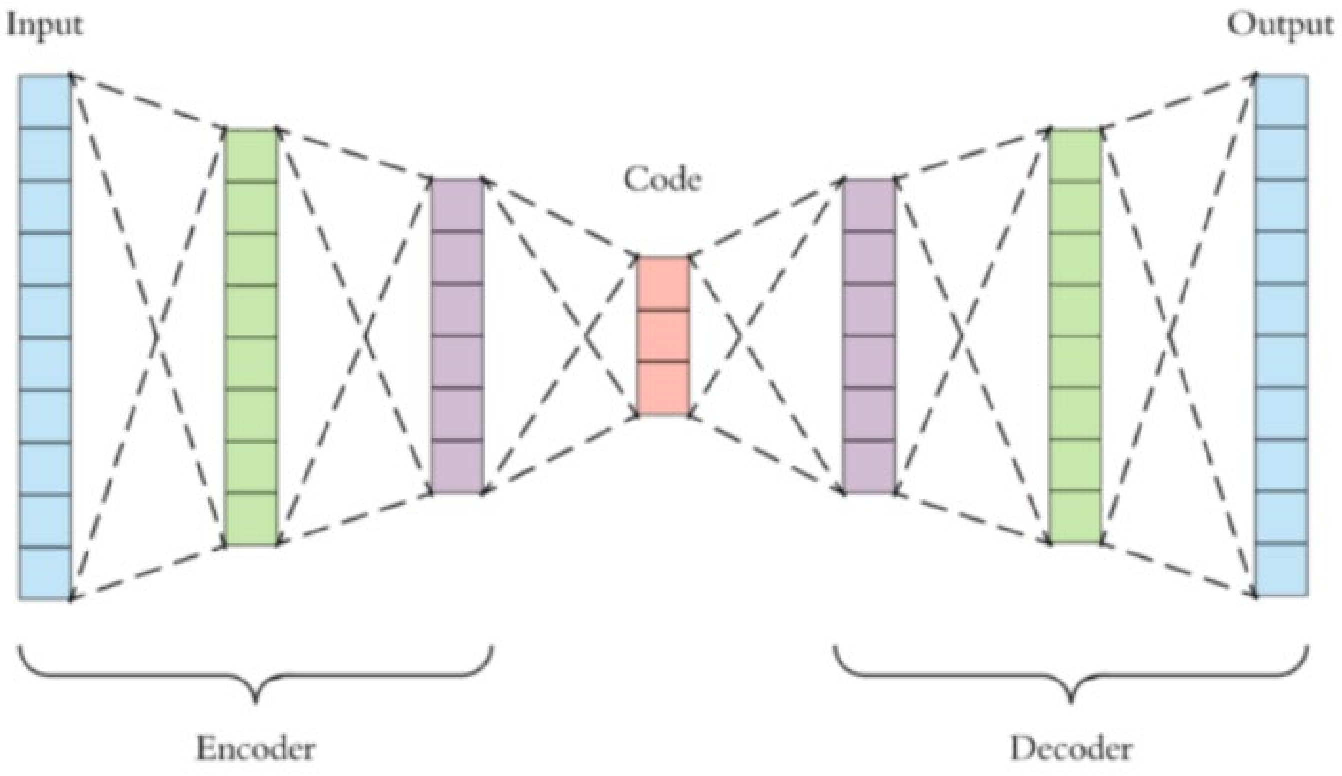


# Nonlinear Dimensionality Reduction

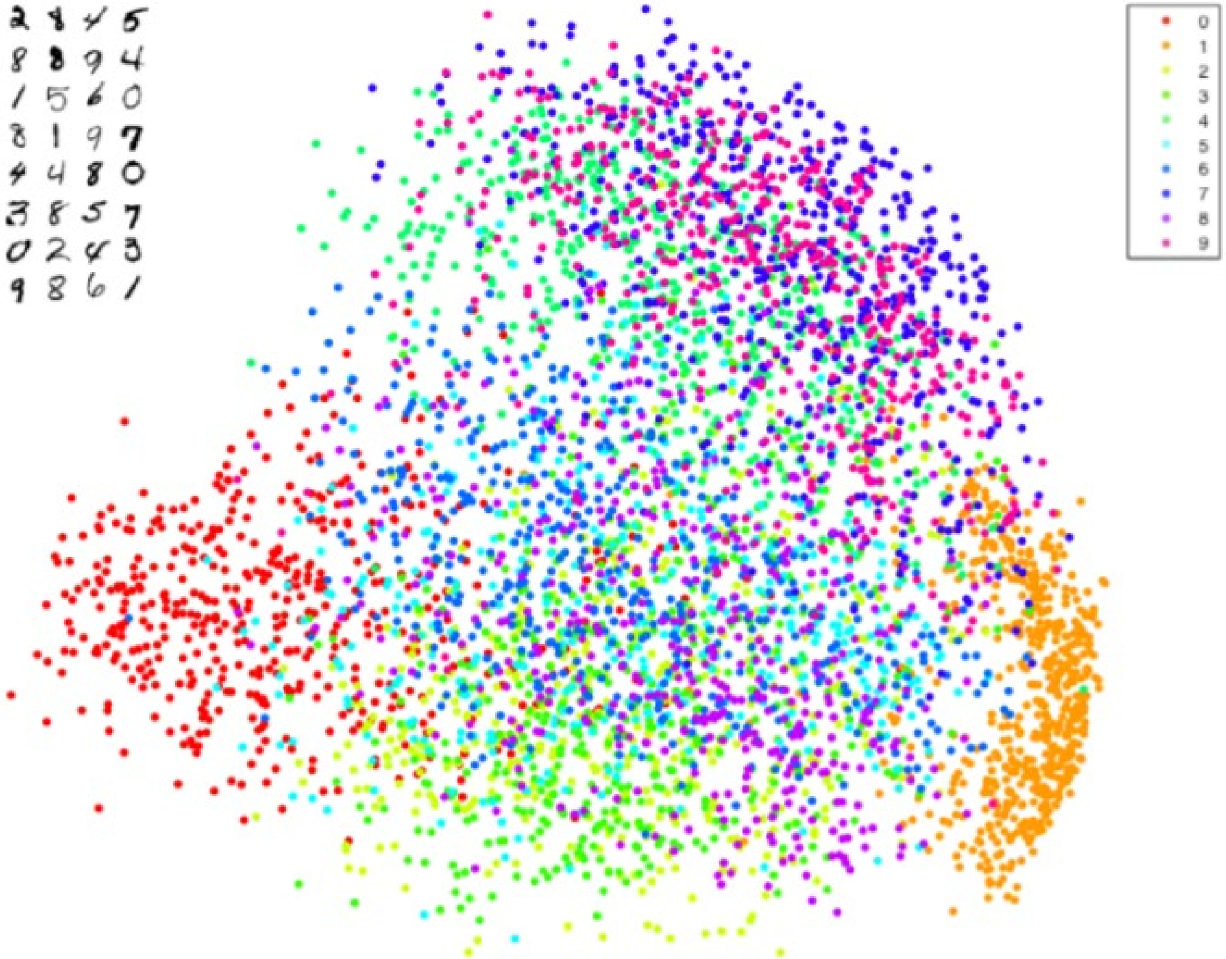


# Nonlinear Dimensionality Reduction

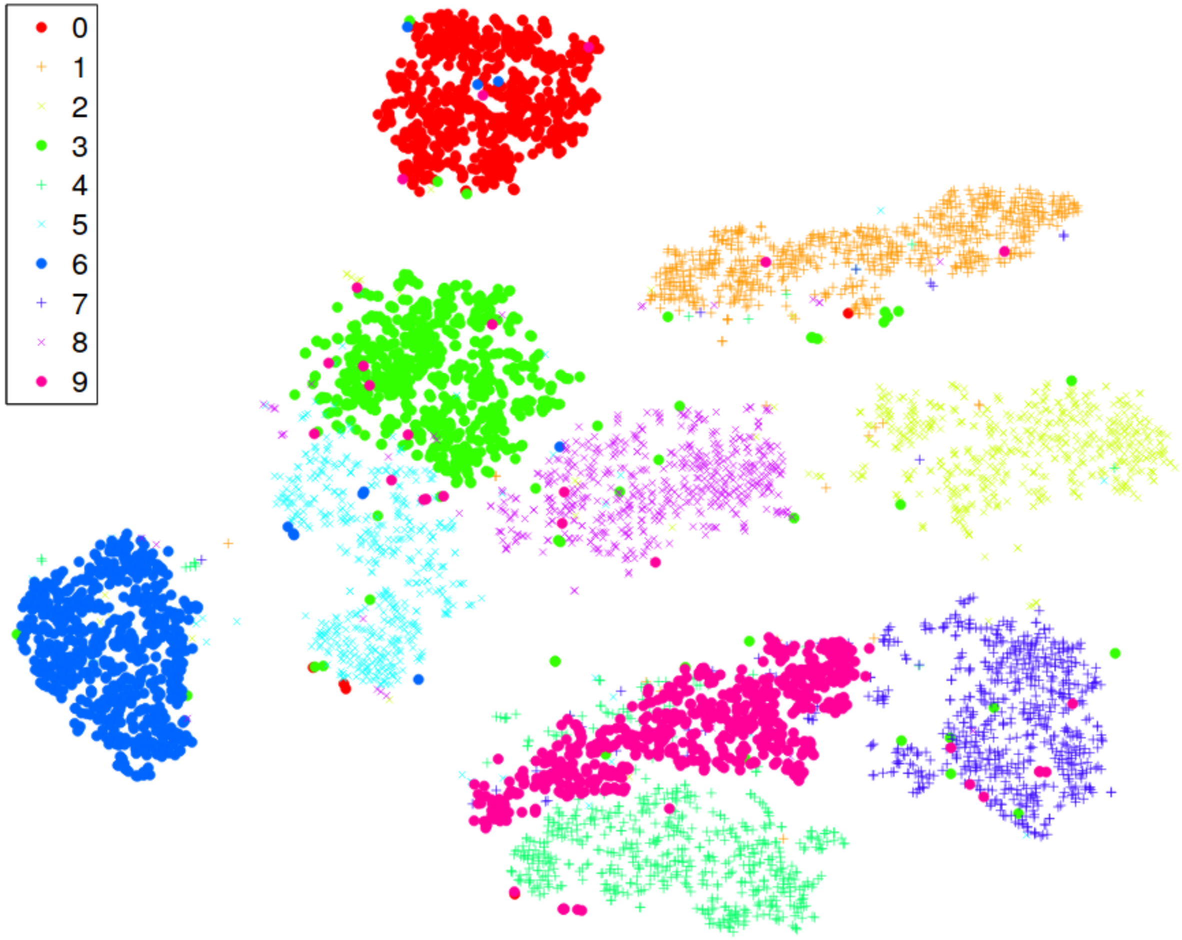
- Neural Networks based auto-encoder



# PCA 2D Embeddings for MNIST



# t-SNE 2D Embeddings for MNIST



# Stochastic Neighbor Embedding (SNE)

- High dimensional neighborhood information as a distribution
- Given $x^{(i)}$, $P_{j|i}$ is the probability that point $x^{(i)}$ chooses $x^{(j)}$ as its neighbor
- Final distribution over pairs is symmetrized

$$P_{j|i} = \frac{\exp\left(-\frac{\left\|x^{(i)} - x^{(j)}\right\|^2}{2\sigma_i^2}\right)}{\sum_{k\neq i} \exp\left(-\frac{\left\|x^{(i)} - x^{(k)}\right\|^2}{2\sigma_i^2}\right)}$$

$$P_{ij} = \frac{1}{2N}\left(P_{i|j} + P_{j|i}\right)$$

# Stochastic Neighbor Embedding (SNE)

- High dimensional neighborhood information as a distribution
- Given $x^{(i)}$, $P_{j|i}$ is the probability that point $x^{(i)}$ chooses $x^{(j)}$ as its neighbor
- Final distribution over pairs is symmetrized

$$P_{j|i} = \frac{\exp\left(-\frac{\left\|x^{(i)} - x^{(j)}\right\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\left\|x^{(i)} - x^{(k)}\right\|^2}{2\sigma_i^2}\right)}$$

$$P_{ij} = \frac{1}{2N}\left(P_{i|j} + P_{j|i}\right)$$

# SNE Objective

- Given data, $x^{(1)}, \dots, x^{(N)} \in \mathbb{R}^D$, we define the distribution $P_{ij}$
- Goal: Find $y^{(1)}, \dots, y^{(N)} \in \mathbb{R}^d$, for some $d \ll D$, minimizing

$$KL(P||Q) = \sum_{ij} P_{ij} \log\left(\frac{P_{ij}}{Q_{ij}}\right) \qquad Q_{ij} = \frac{\exp\left(-\left\|y^{(i)} - y^{(j)}\right\|^2\right)}{\sum_{l \neq k} \exp\left(-\left\|y^{(l)} - y^{(k)}\right\|^2\right)}$$

$$P_{j|i} = \frac{\exp\left(-\left\|x^{(i)} - x^{(j)}\right\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\left\|x^{(i)} - x^{(k)}\right\|^2 / 2\sigma_i^2\right)} \qquad P_{ij} = \frac{1}{2N}\left(P_{i|j} + P_{j|i}\right)$$

# KL Divergence

- Measures distance between two distributions, P and Q
- Not a metric function – not symmetric
- $KL(P||Q) \geq 0$
- $KL(P||Q) = 0$ only when $P == Q$

$$KL(P||Q) = \sum_{ij} P_{ij} \log\left(\frac{P_{ij}}{Q_{ij}}\right)$$

# Optimizing SNE

$$\min_{y^{(1)}\dots y^{(N)}} KL(P||Q) = \min_{y^{(1)}\dots y^{(N)}} \sum_{ij} P_{ij} \log\left(\frac{P_{ij}}{Q_{ij}}\right)$$

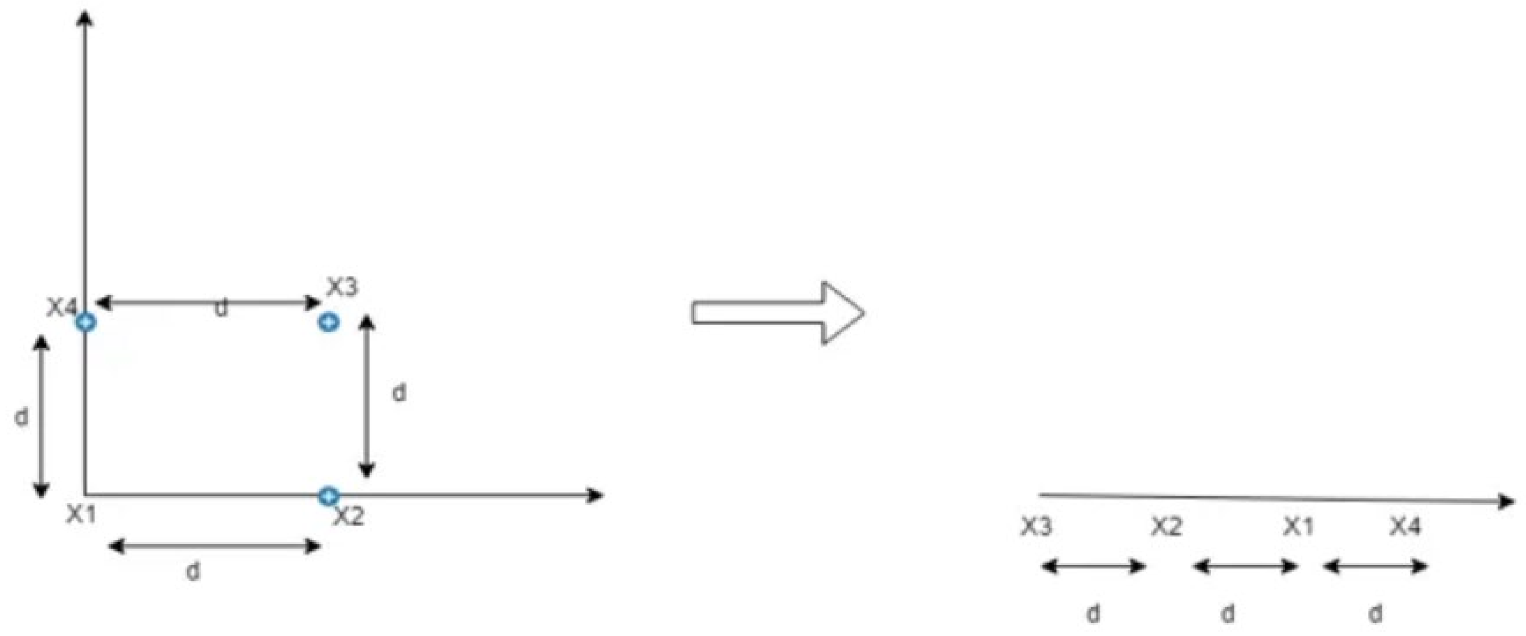
$$= \min_{y^{(1)}\dots y^{(N)}} -\sum_{ij} P_{ij} \log(Q_{ij}) + \text{const}$$

$$\frac{\partial}{\partial y^{(i)}} - \sum_{ij} P_{ij} \log(Q_{ij}) = \cdots = \sum_{j} (P_{ij} - Q_{ij})(y^{(i)} - y^{(j)})$$

- Gradient descent!
- Non-convex, multiple runs!
- Main issue – crowding problem

# Crowding Problem

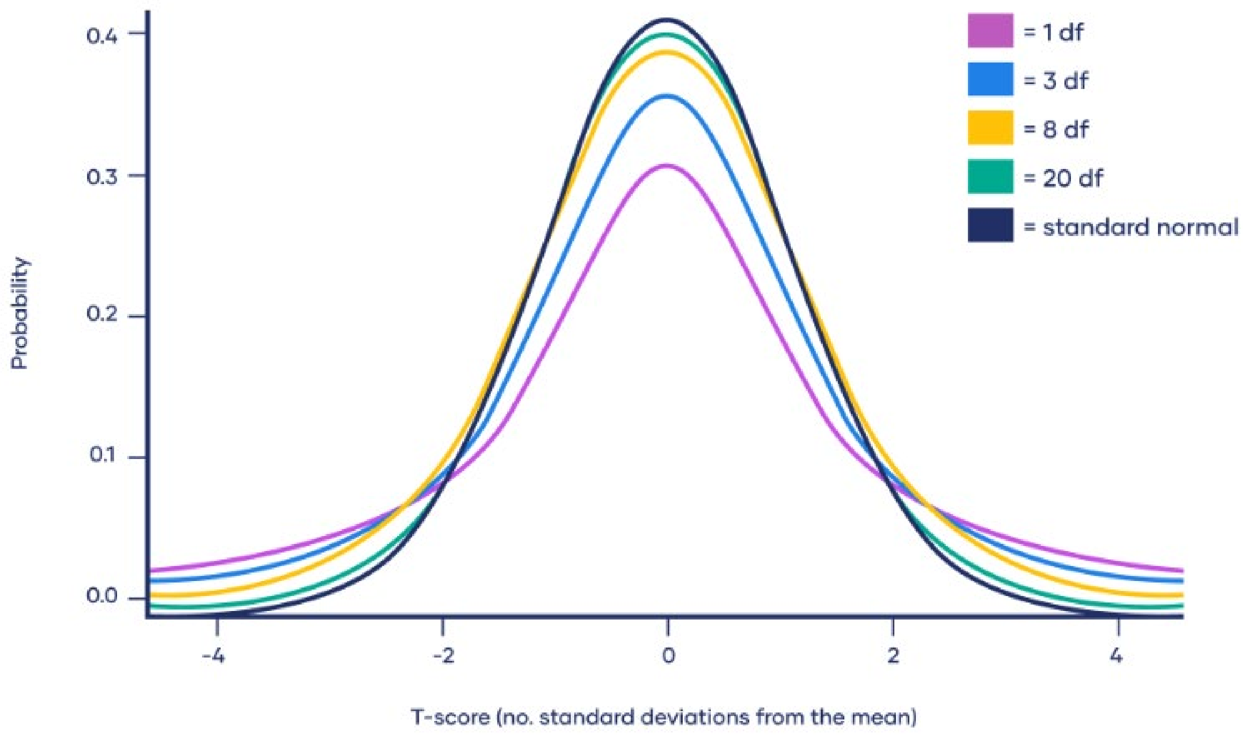
- In high dimension, we have more room
- In low dimension, we do not have enough room to accommodate all neighbors




[A Probabilistic Algorithm to Reduce Dimensions: t — Distributed Stochastic Neighbor Embedding (t-SNE) | by Rajvi Shah | Towards AI]

# t-SNE

- t-Ditributed Stochastic Neighbor Embedding
- Student’s t distribution
- Probability goes to zero much slower than a Gaussian



# t-SNE

- t-Ditributed Stochastic Neighbor Embedding
- We can now redefine $Q_{ij}$ as
- $P_{ij}$ is same as before

$$Q_{ij} = \frac{\left(1 + \left\|y^{(i)} - y^{(j)}\right\|^2\right)^{-1}}{\sum_{l \neq k} \left(1 + \left\|y^{(l)} - y^{(k)}\right\|^2\right)^{-1}}$$

# t-SNE Algorithms

---

**Algorithm 1**: Simple version of t-Distributed Stochastic Neighbor Embedding.

---

**Data**: data set $X = \{x_1, x_2, ..., x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations $T$, learning rate $\eta$, momentum $\alpha(t)$.
**Result**: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, ..., y_n\}$.
**begin**
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i}+p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  **for** $t$=*1* **to** $T$ **do**
    compute low-dimensional affinities $q_{ij}$ (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  **end**
**end**

---

# t-SNE Visualization

- Visualizing MNIST: An Exploration of Dimensionality Reduction - colah's blog