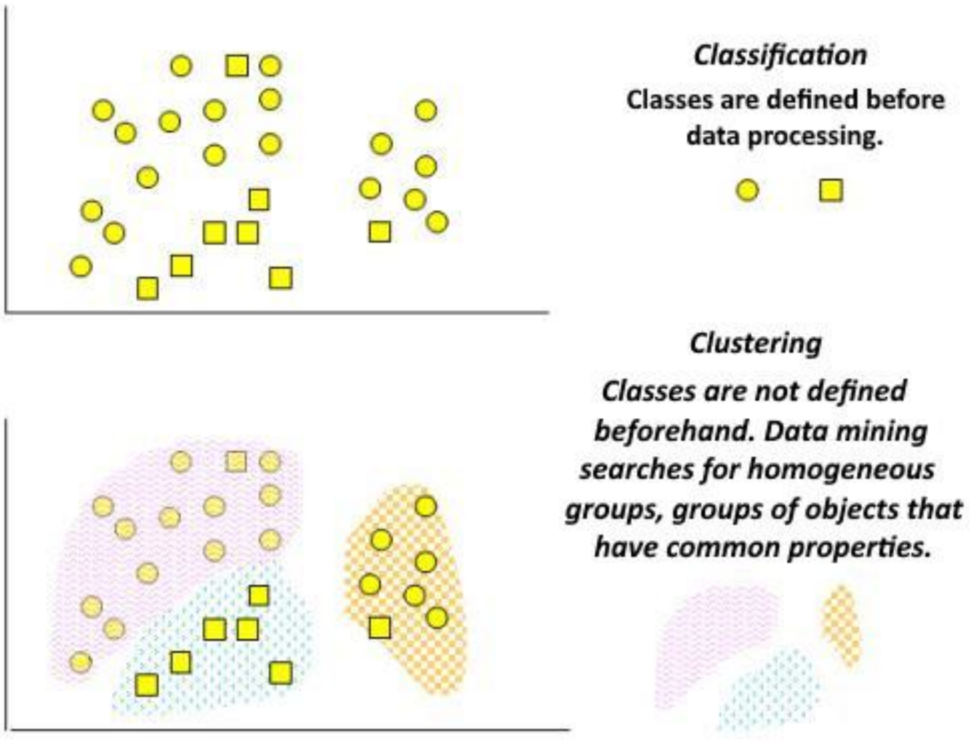
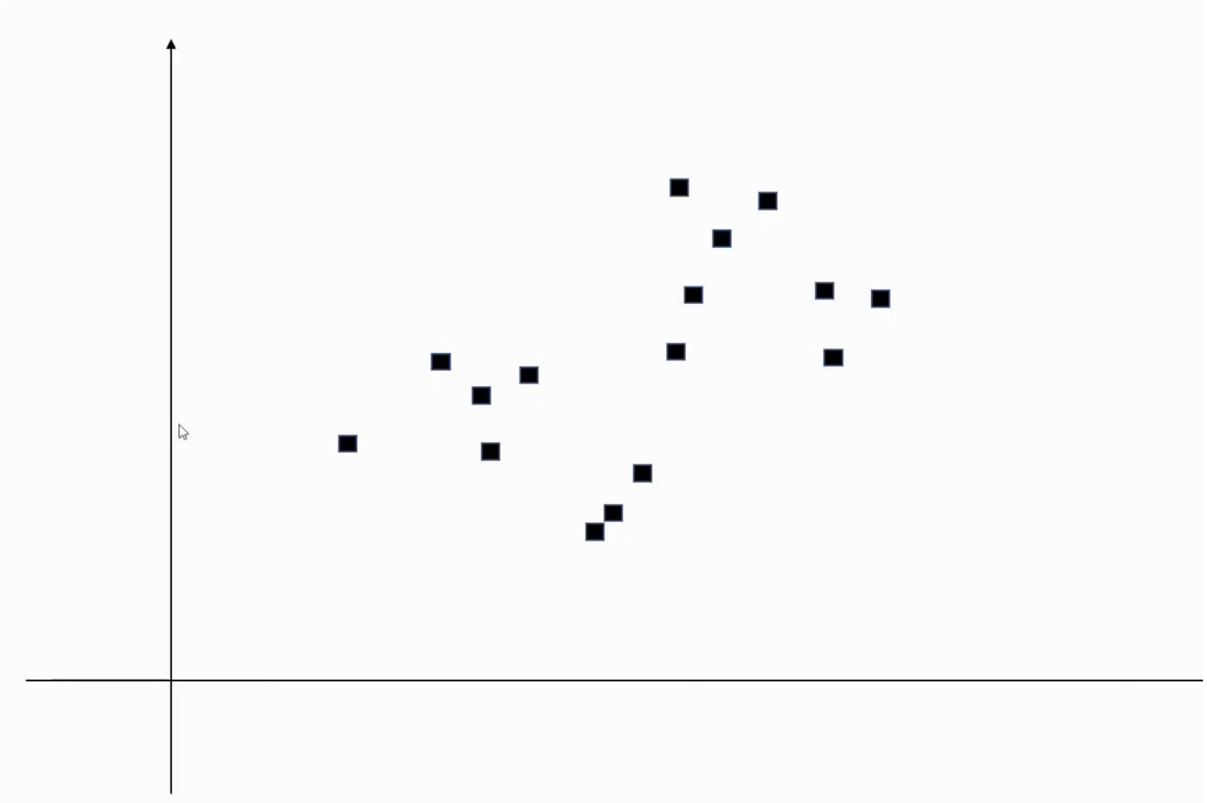
CYBER 207
Applied Machine Learning for Cybersecurity

Summer Week 8
Live Session Slides

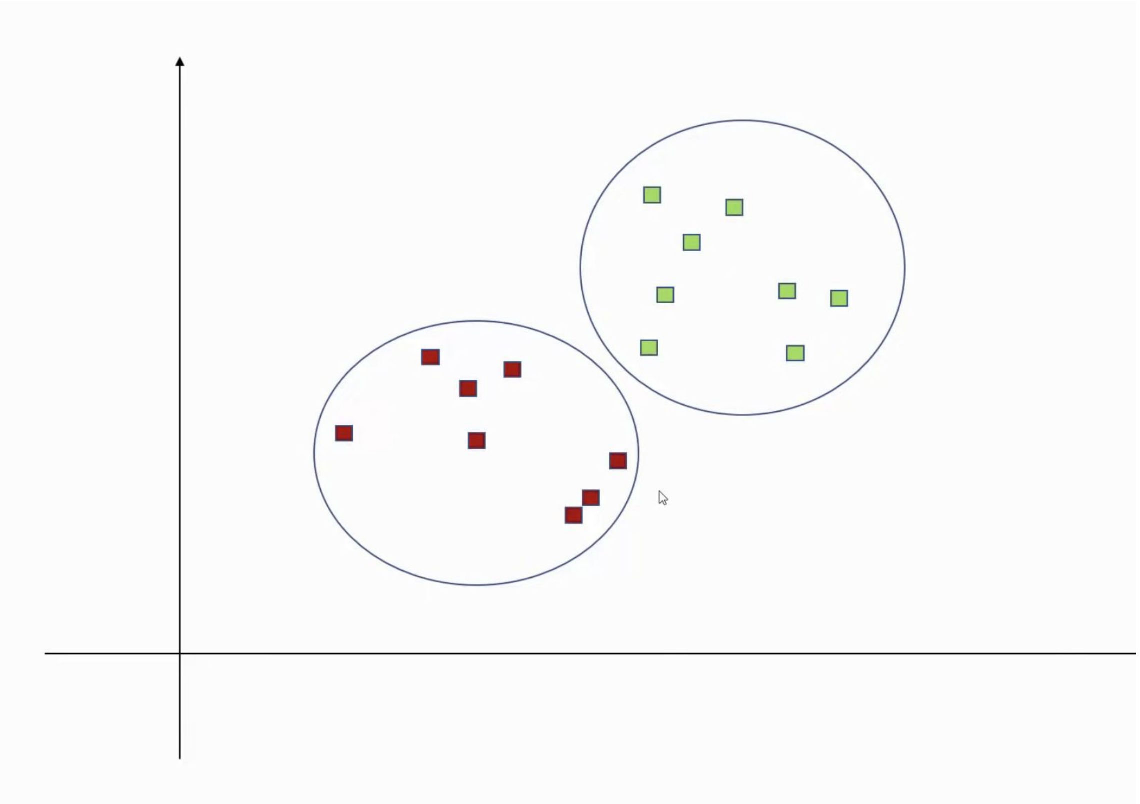
Clustering vs Classification



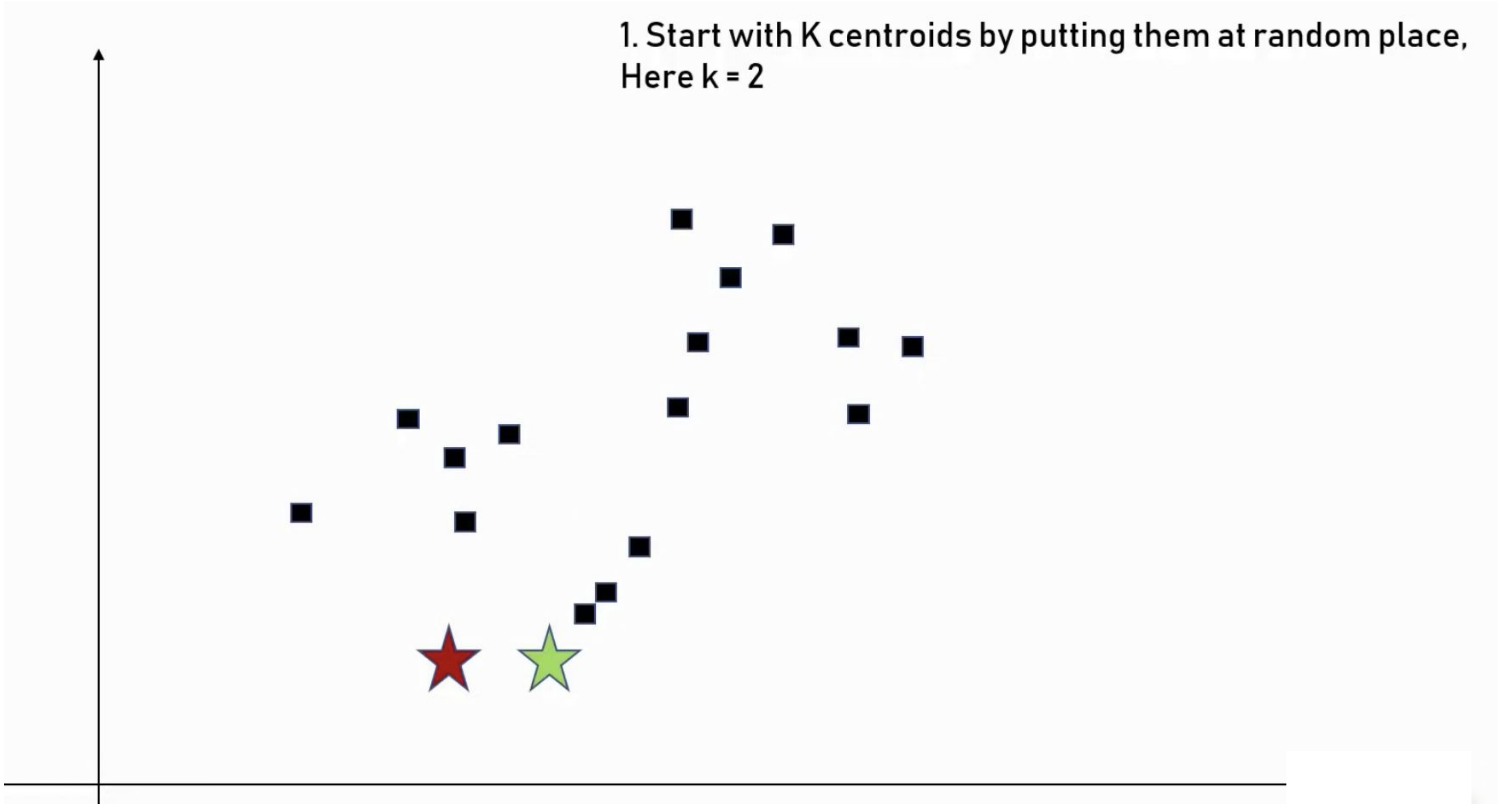
K-Means Clustering



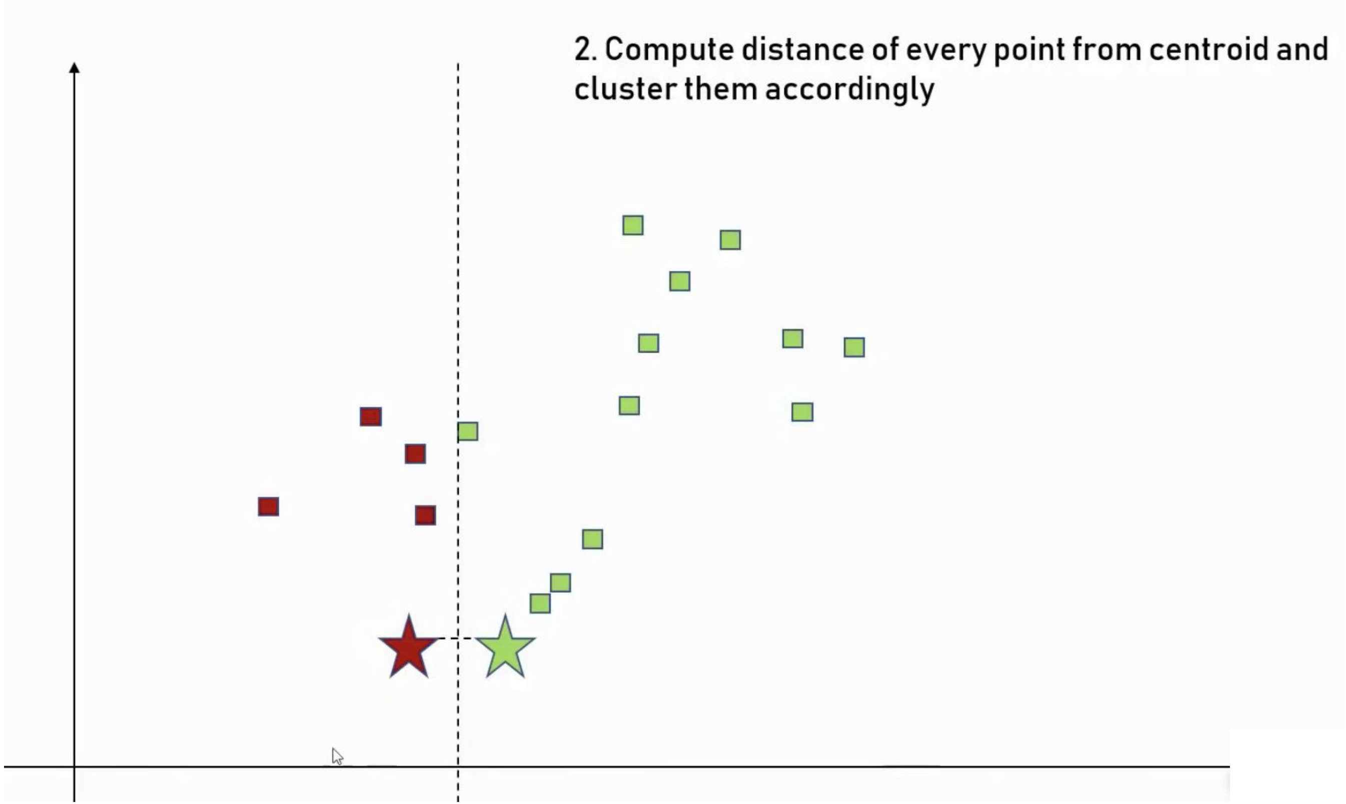
K-Means Clustering



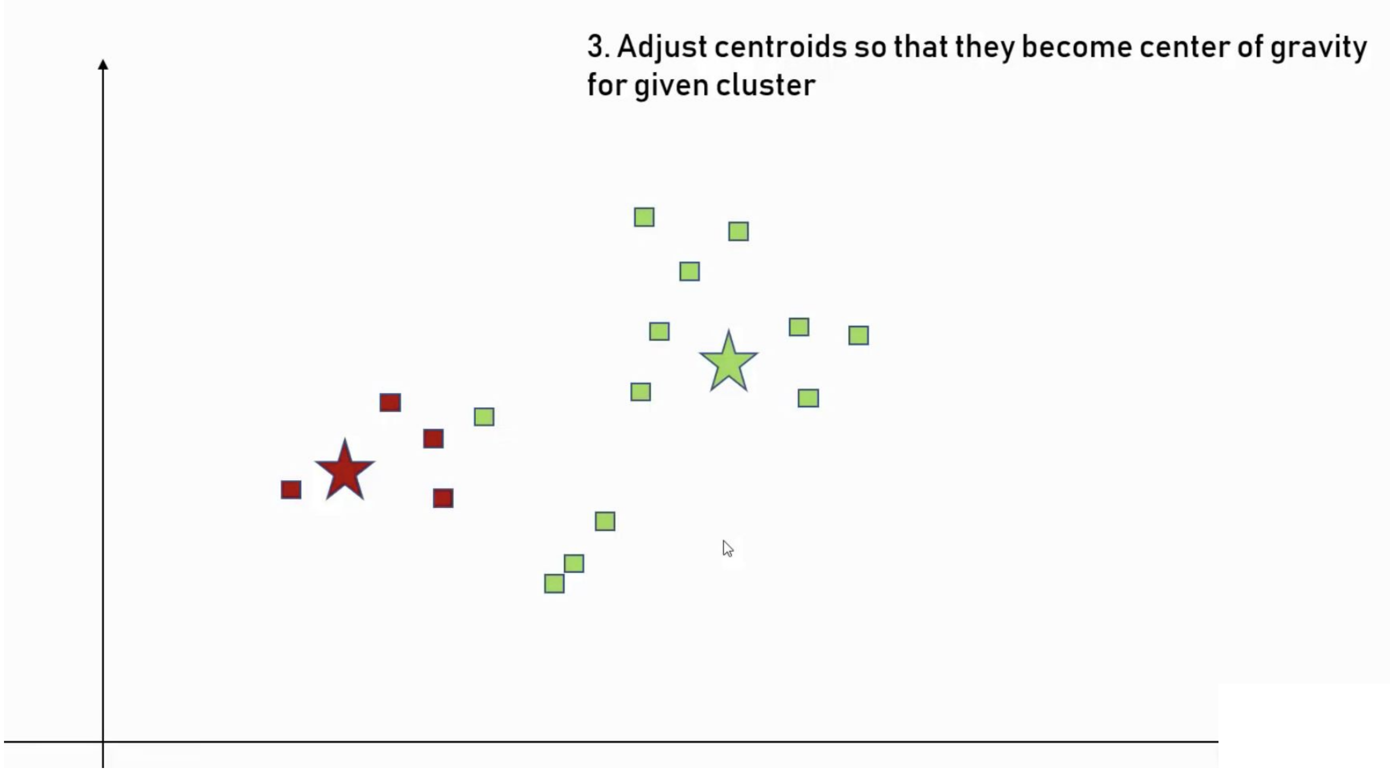
K-Means Clustering



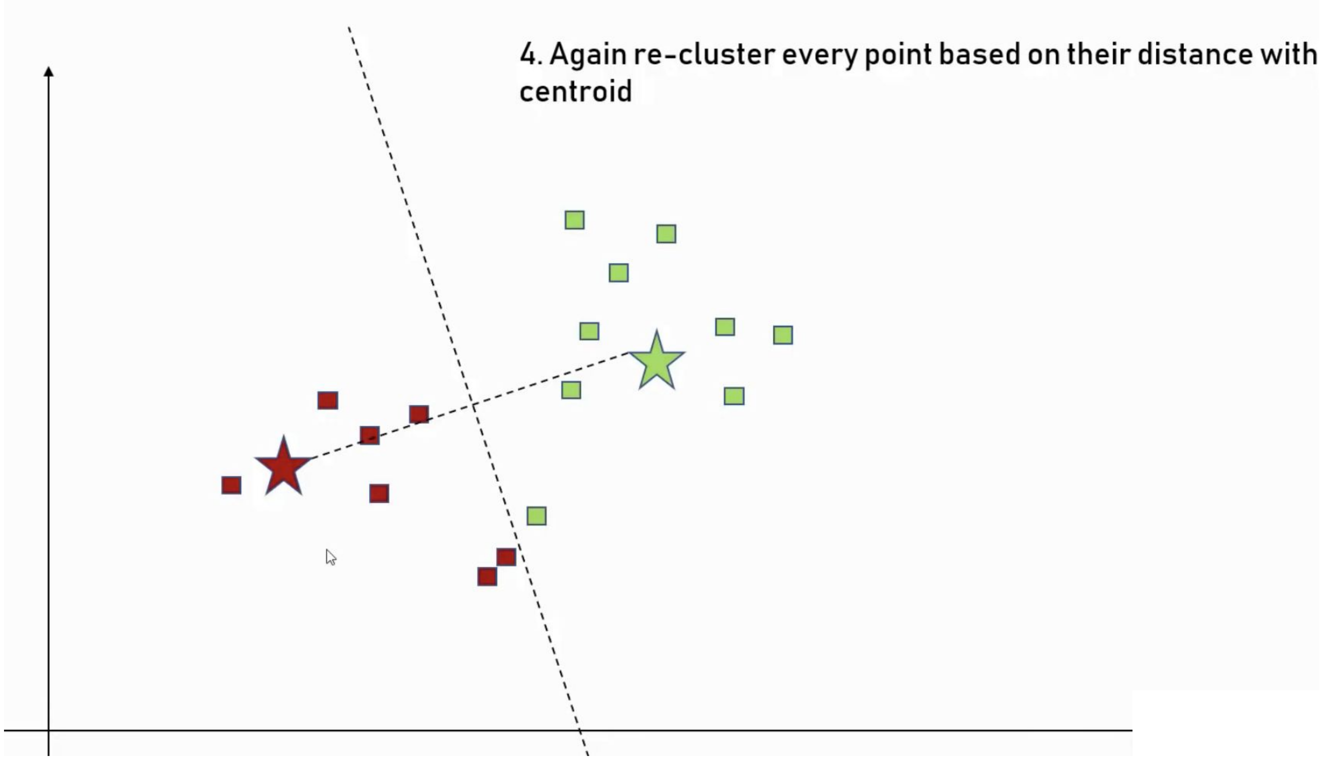
K-Means Clustering



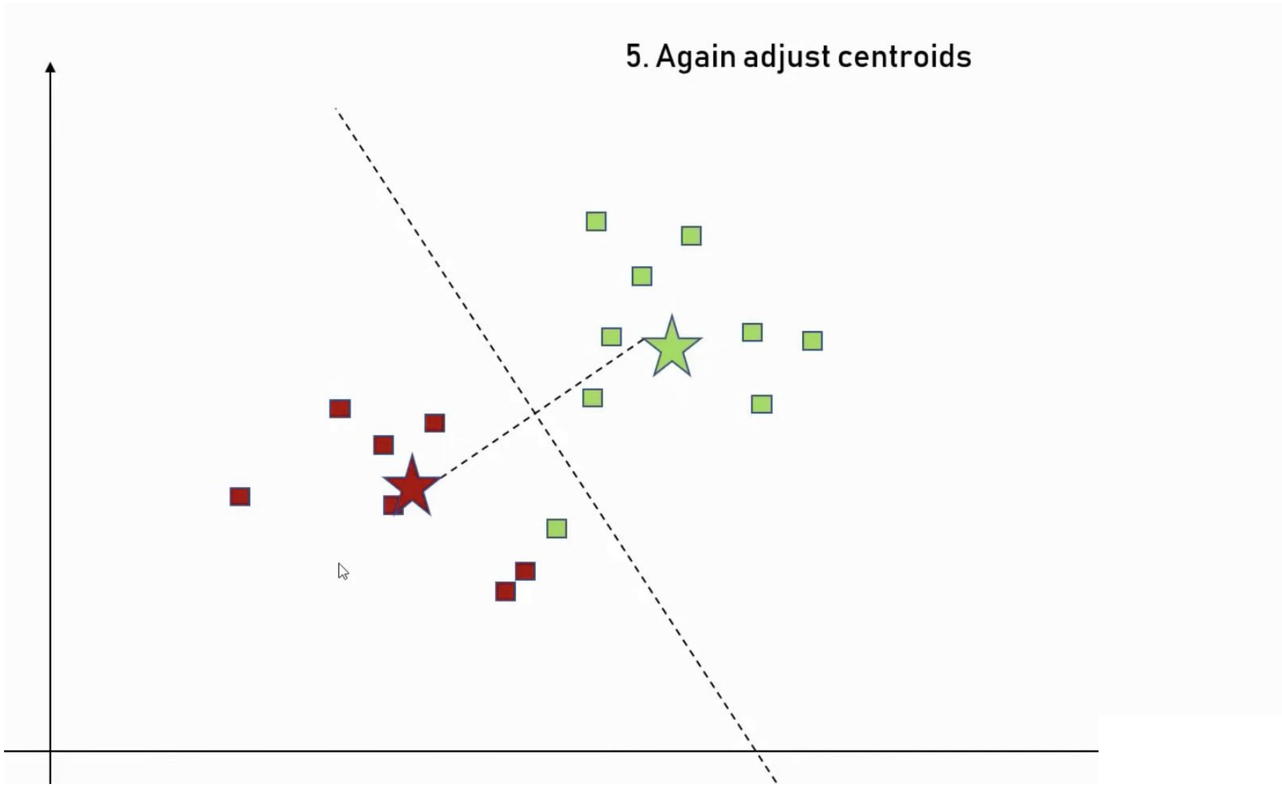
K-Means Clustering



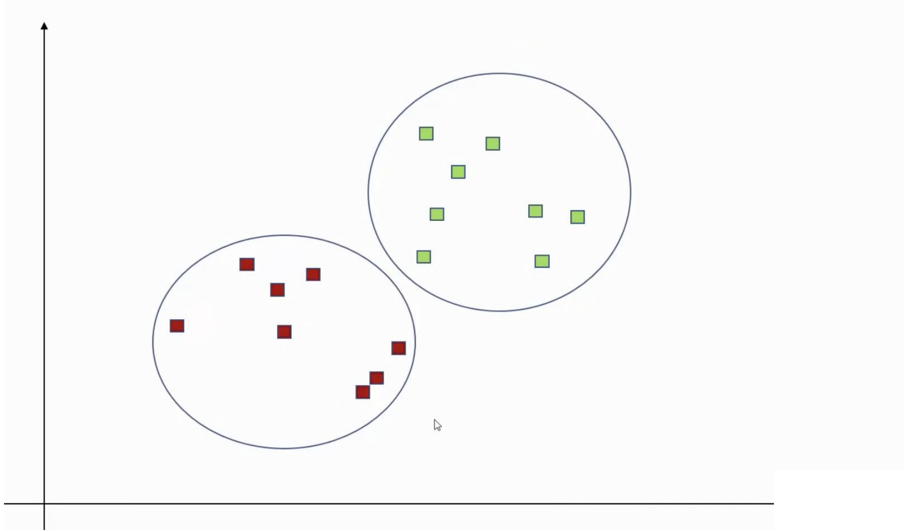
K-Means Clustering

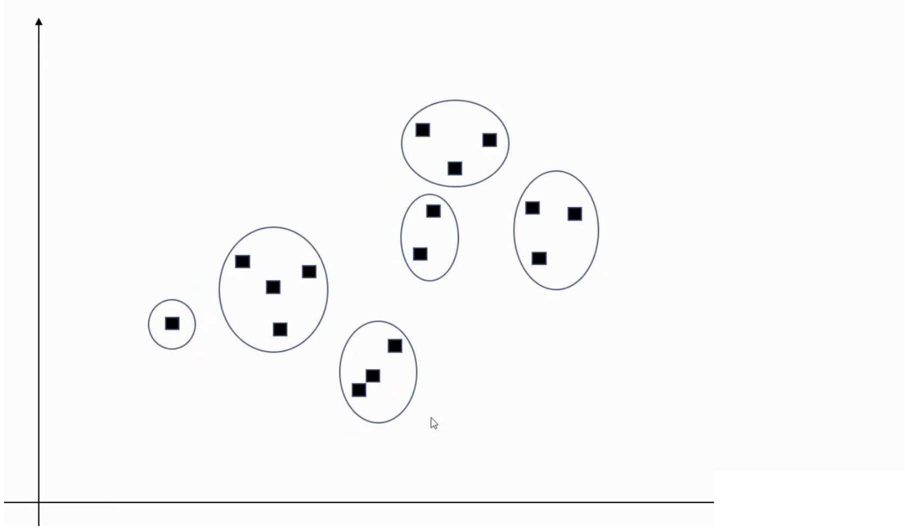


K-Means Clustering

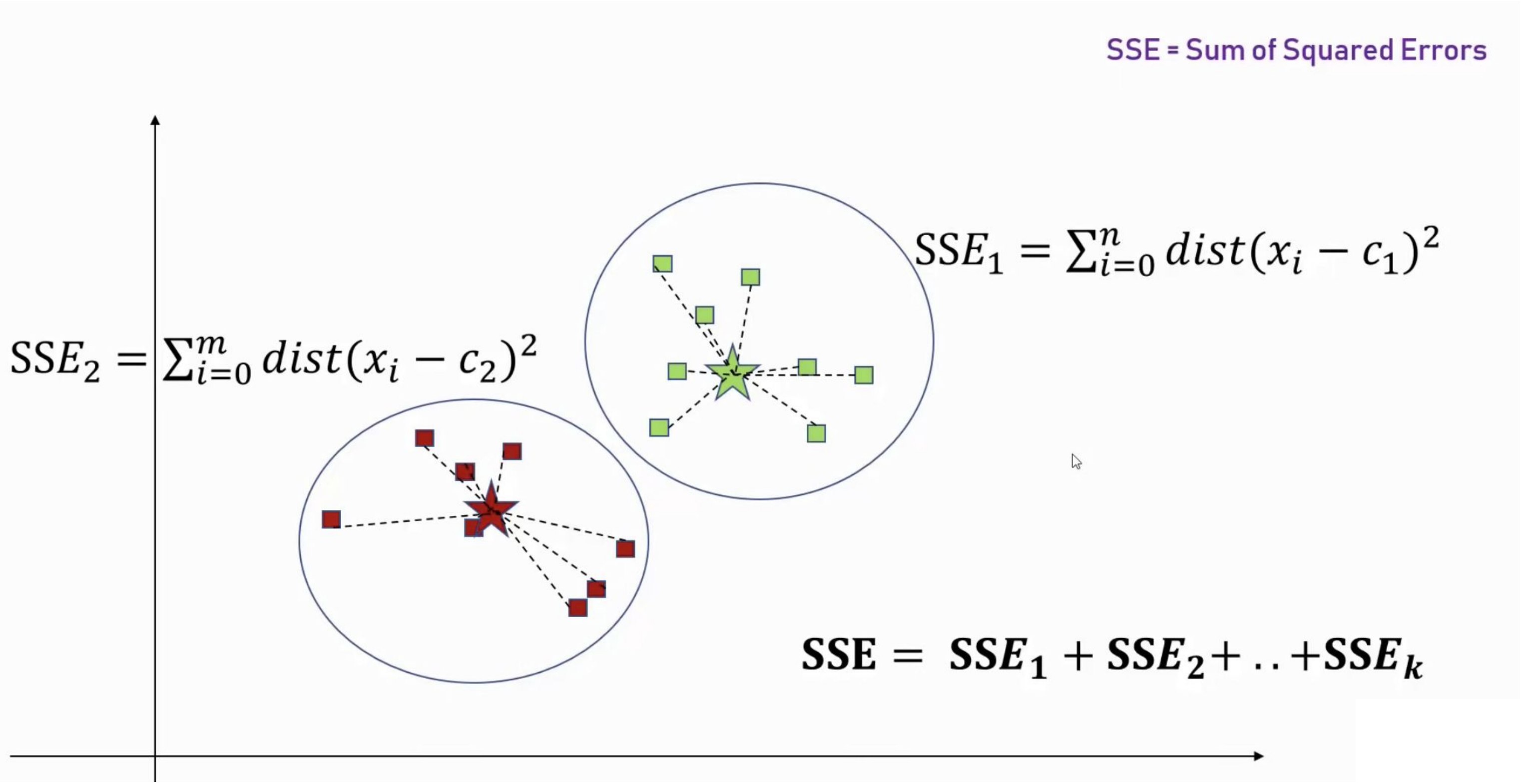


K-Means Clustering: Finding k

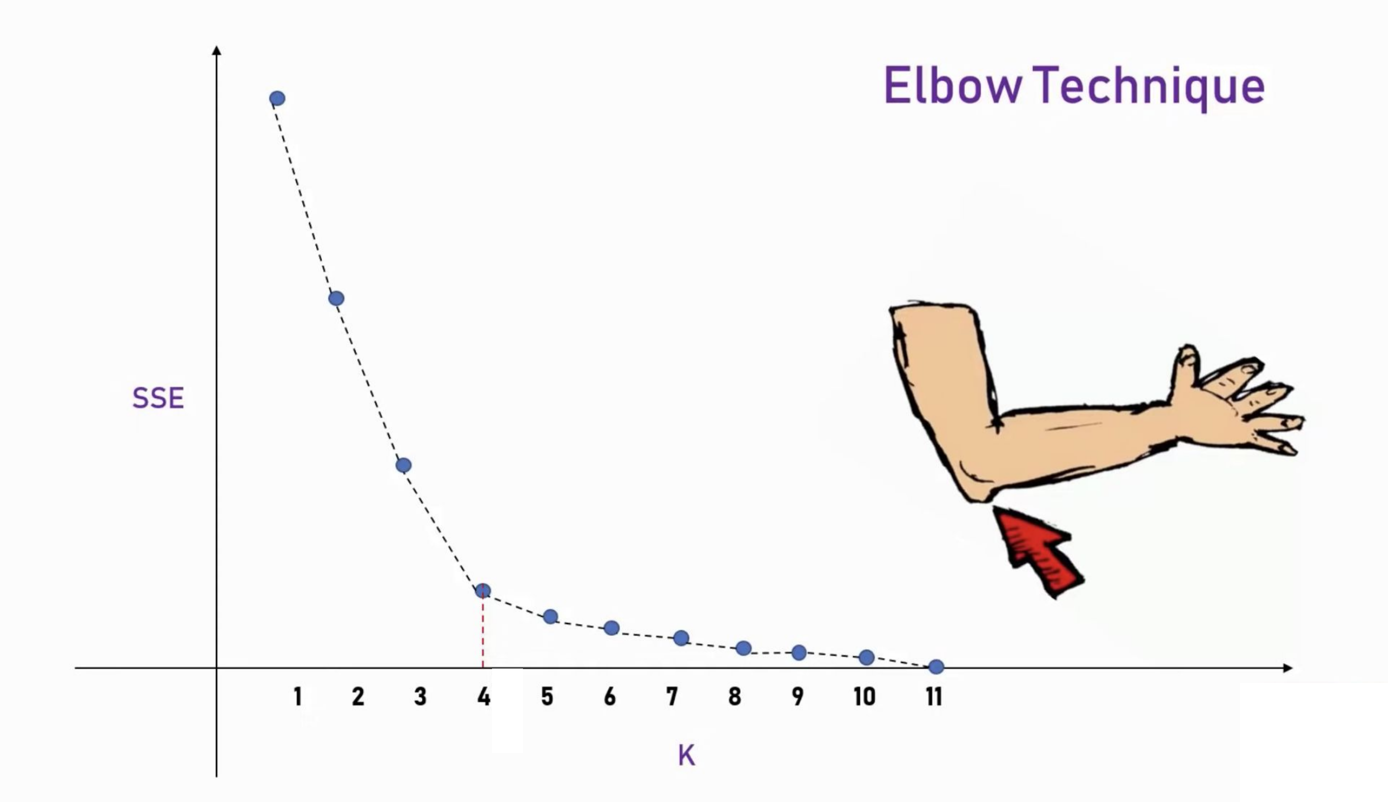




K-Means Clustering: Finding k



K-Means Clustering: Finding k

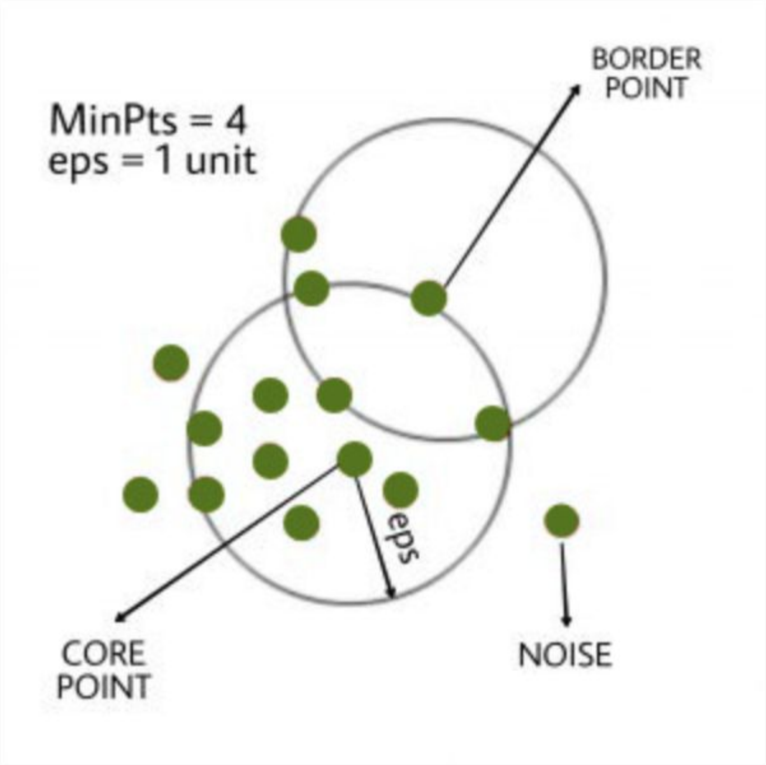


K-Means Algorithm

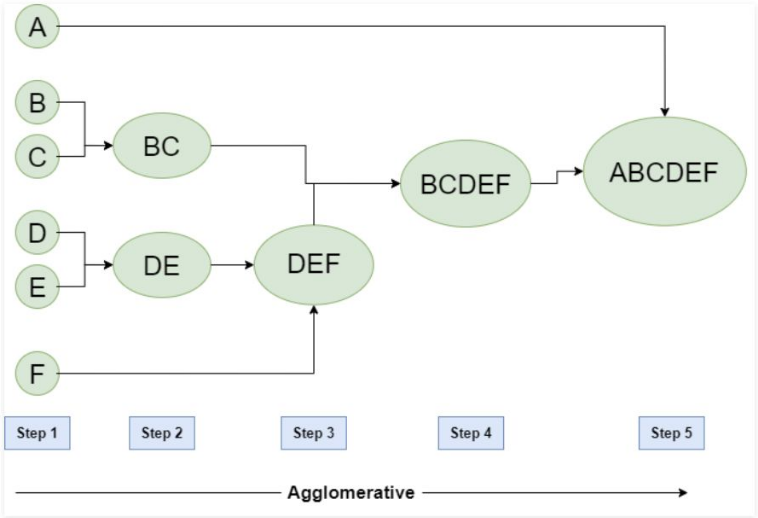
Algorithm 1 k-means algorithm

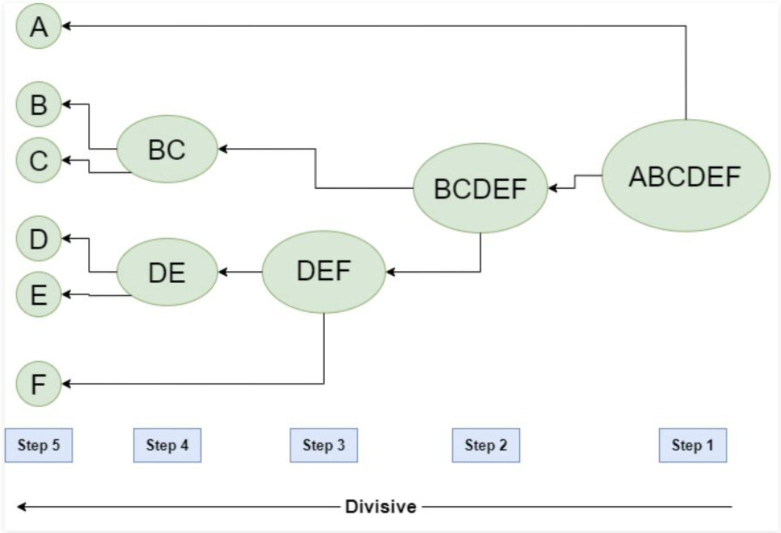
```
Specify the number k of clusters to assign.
Randomly initialize k centroids.
repeat
    expectation: Assign each point to its closest centroid.
    maximization: Compute the new centroid (mean) of each cluster.
until The centroid positions do not change.
```

K-Means Algorithm: Types: DBSCAN

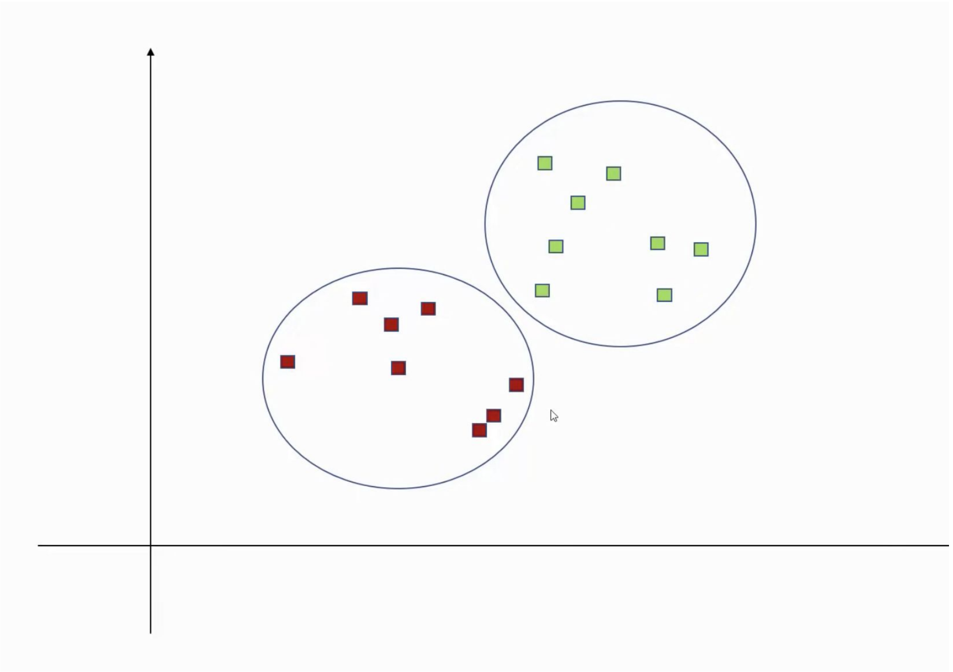


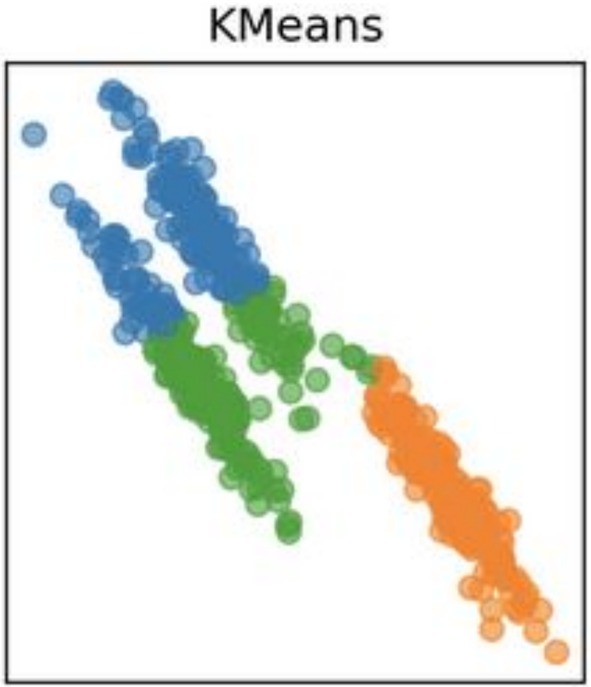
K-Means Algorithm: Types: Hierarchical



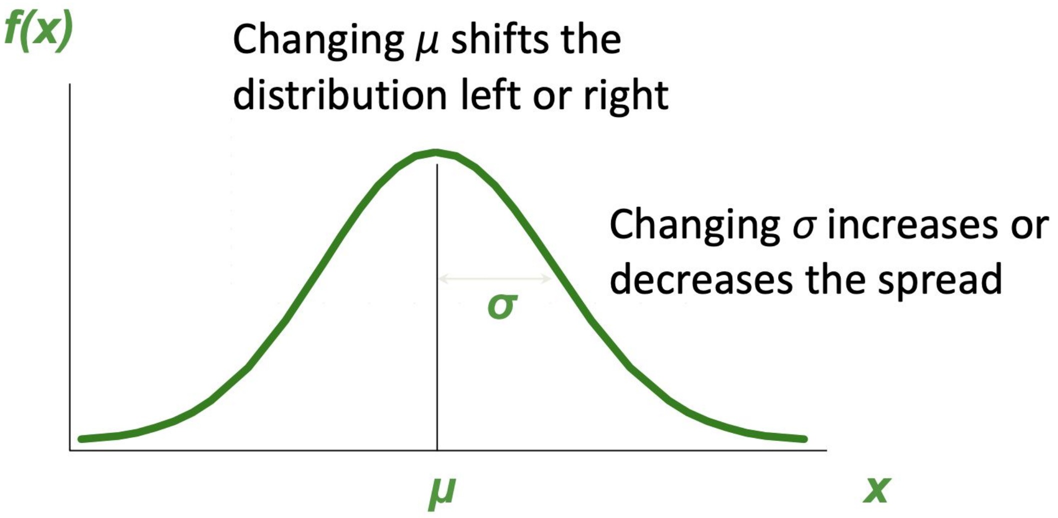


K-Means Limitations





Gaussian Distribution



Probability density function $f(x)$ is a function of x given μ and σ

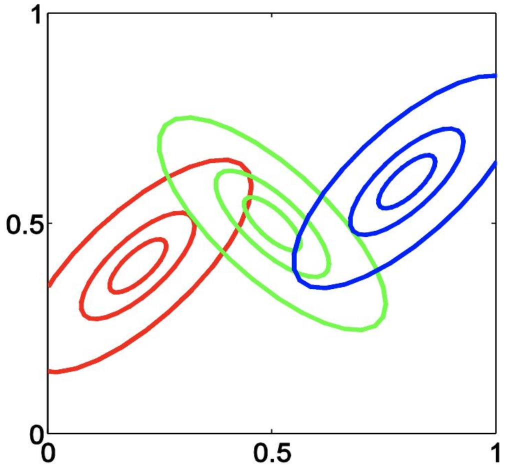
$$N(x \mid \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$$

Gaussian Mixture Models

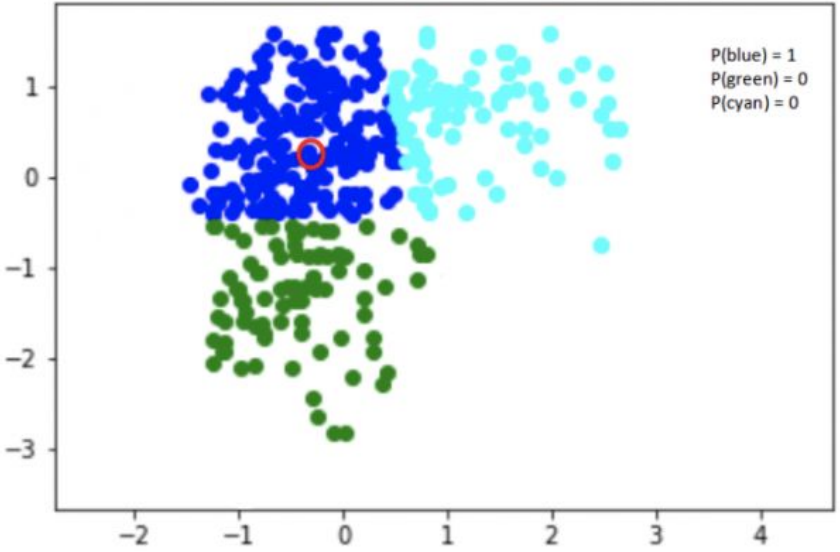
- Linear combination of Gaussians

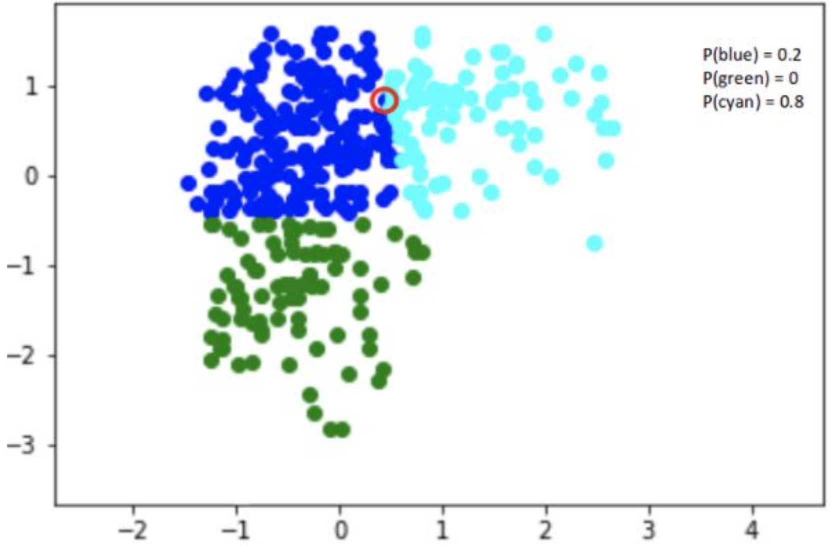
$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \quad \text{where} \quad \sum_{k=1}^{K} \pi_k = 1, \quad 0 \leqslant \pi_k \leqslant 1$$

parameters to be estimated

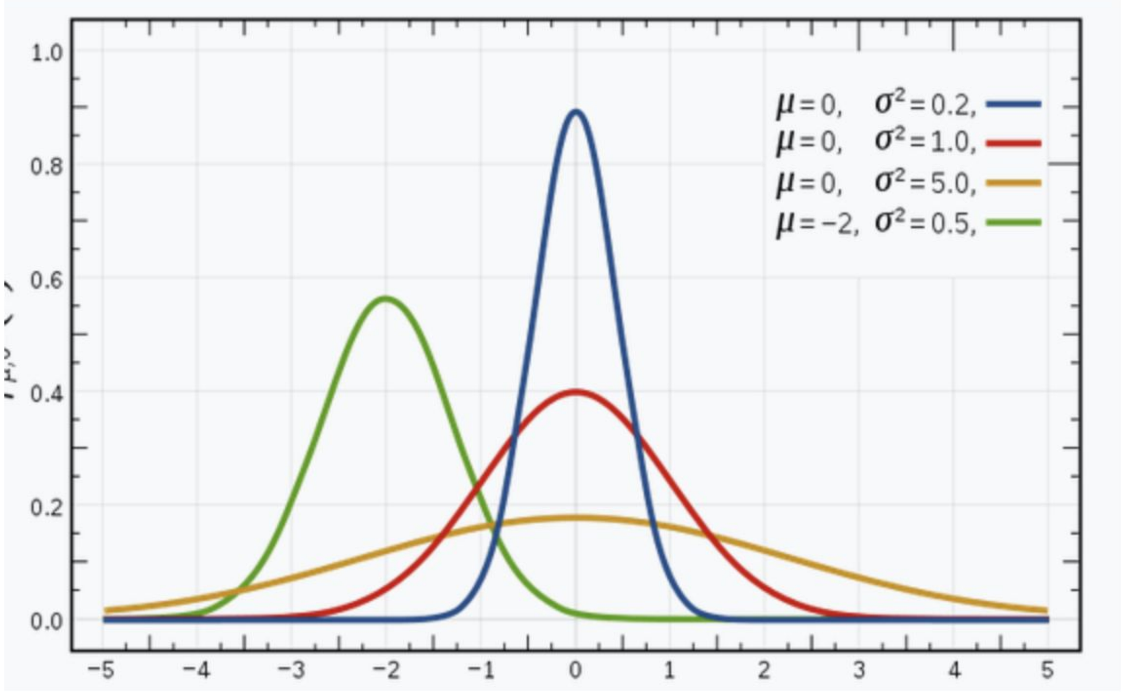


Gaussian Mixture Models



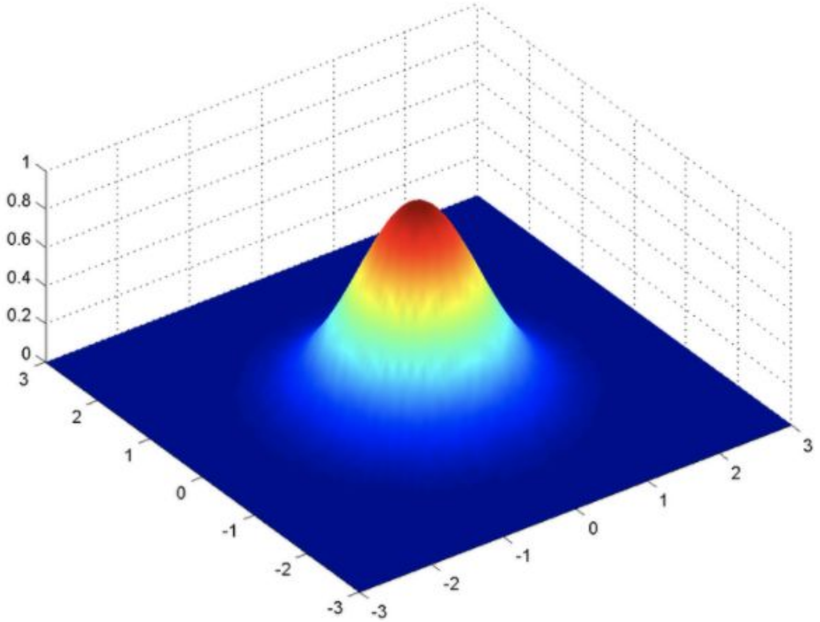


Gaussian Distribution Contd.



$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Gaussian Distribution Contd.



$$f(x \mid \mu, \Sigma) = \frac{1}{\sqrt{2\pi|\Sigma|}} \exp\left[-\tfrac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^t \Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \right]$$

Expectation Maximization

E-Step

$$r_{ic} = \frac{\text{Probability Xi belongs to c}}{\text{Sum of probability Xi belongs to } c_1, c_2, \ldots c_k} = \frac{\pi_c \mathcal{N}(x_i \; ; \; \mu_c, \Sigma_c)}{\sum_{c'} \pi_{c'} \mathcal{N}(x_i \; ; \; \mu_{c'}, \Sigma_{c'})}$$

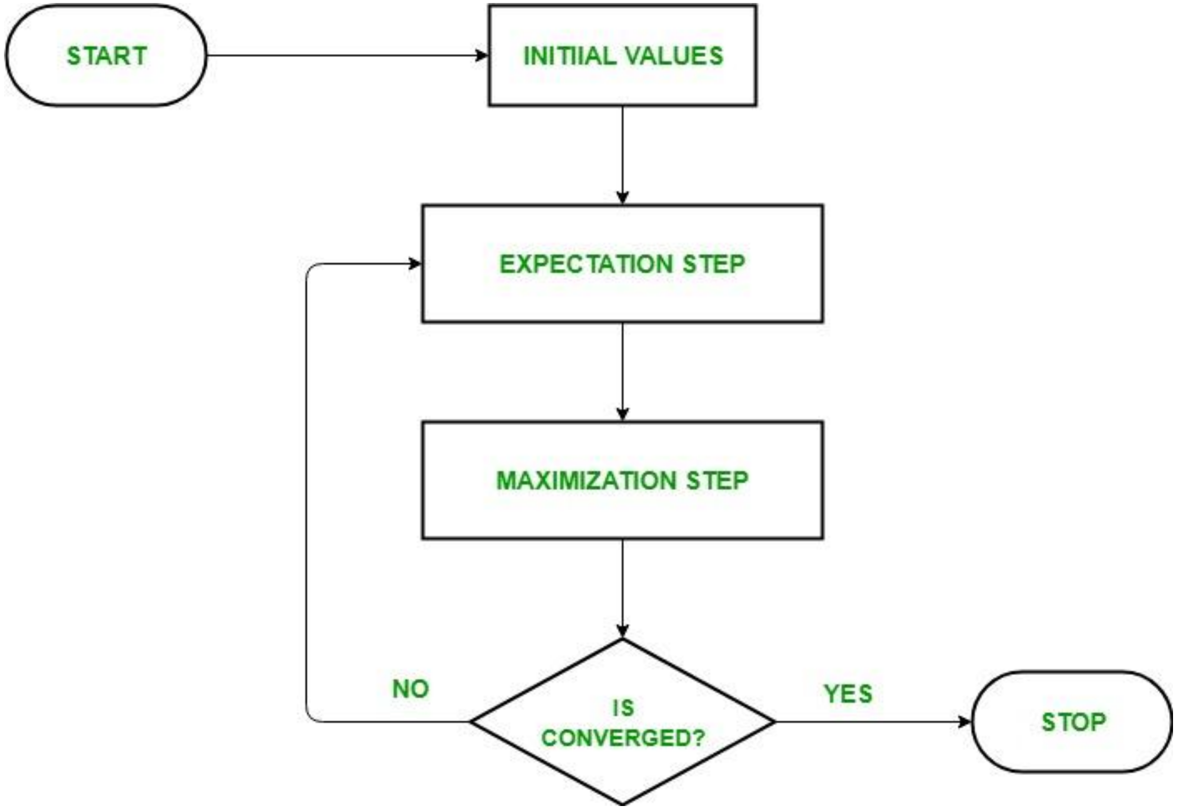
M-Step

$$\Pi = \frac{\text{Number of points assigned to cluster}}{\text{Total number of points}}$$

$$\mu = \frac{1}{\text{Number of points assigned to cluster}} \Sigma_i \; r_{ic} x_i$$

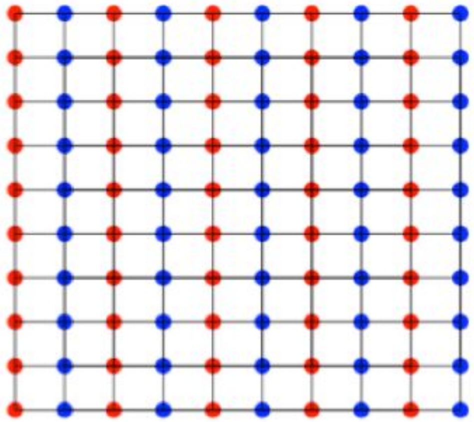
$$\Sigma_c = \frac{1}{\text{Number of points assigned to cluster}} \Sigma_i \; r_{ic} \, (x_i - \mu_c)^T \, (x_i - \mu_c)$$

Expectation Maximization

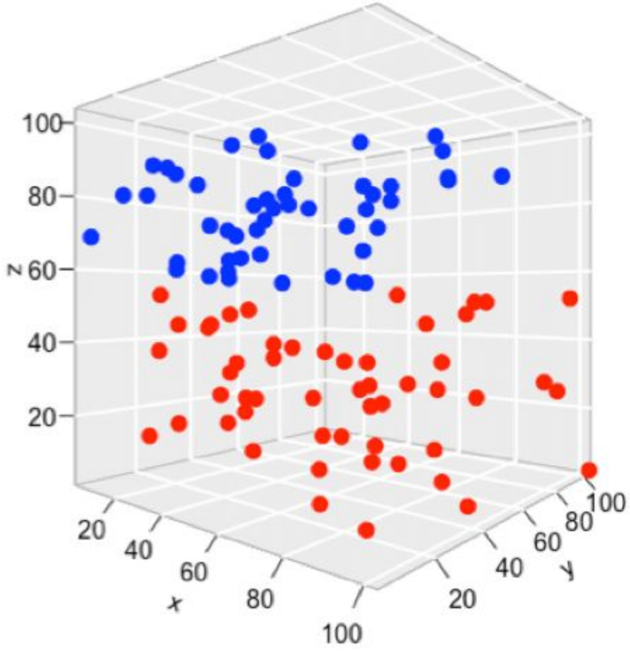


Curse of Dimensionality

(A) 1-D



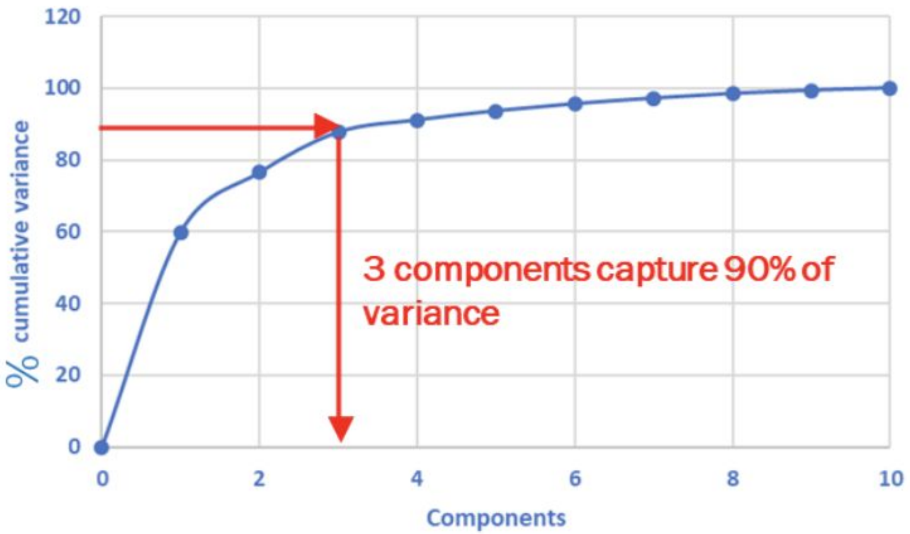
(B) 2-D



(C) 3-D

Principal Component Analysis

Component	Initial Eigenvalues		
	Total	% of Variance	Cumulative %
1	5.994	59.938	59.938
2	1.654	16.545	76.482
3	1.123	11.227	87.709
4	.339	3.389	91.098
5	.254	2.541	93.640
6	.199	1.994	95.633
7	.155	1.547	97.181
8	.130	1.299	98.480
9	.091	.905	99.385
10	.061	.615	100.000



Principal Component Analysis (Terms)

Dimensionality: It is the number of features or variables present in the given dataset. More easily, it is the number of columns present in the dataset.

Correlation: It signifies that how strongly two variables are related to each other.

Orthogonal: It defines that variables are not correlated to each other, and hence the correlation between the pair of variables is zero.

Eigenvectors: If there is a square matrix M, and a non-zero vector v is given. Then v will be eigenvector if Av is the scalar multiple of v.

Covariance Matrix: A matrix containing the covariance between the pair of variables is called the Covariance Matrix.

Principal Component Analysis Terms

1. Getting the dataset.
2. Representing data into a structure.
3. Standardize the data
4. Getting the covariance
5. Calculating Eigenvalues and Eigenvectors
6. Sorting the Eigenvectors
7. Calculating the new features or principal components
8. Remove less or unimportant features from the new dataset.

Code Review

Eigenfaces: Key Idea

Assume that most face images lies on a low dimensional subspace determined by the first ***k (k<<<d)*** directions of maximum variance

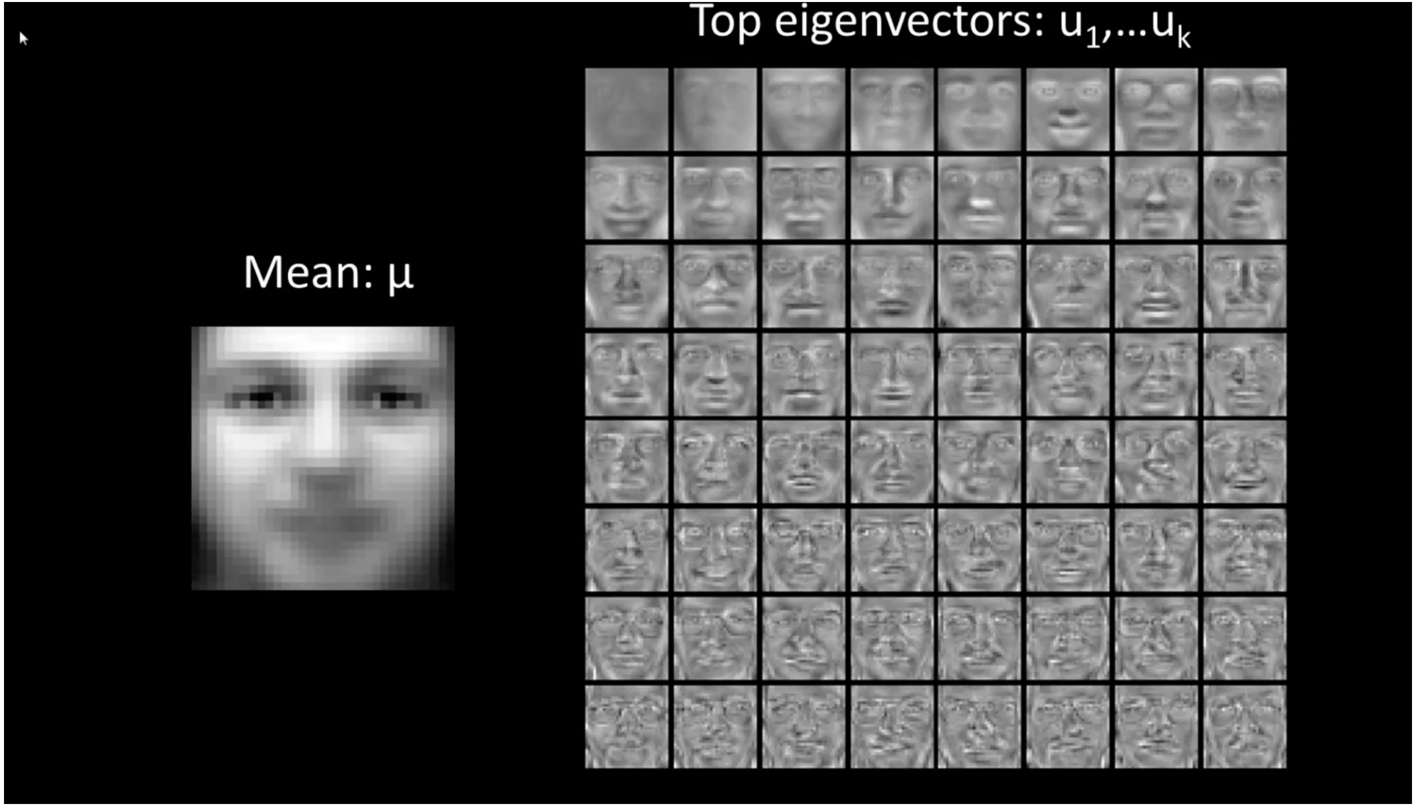
Use PCA to determine the vectors or Eigenfaces $\mathbf{u}_1, \mathbf{u}_2, \ldots\ldots \mathbf{u}_k$ that span the subspace

Represent all face images in the dataset as linear combinations of eigenfaces. Find the coefficients by dot product.

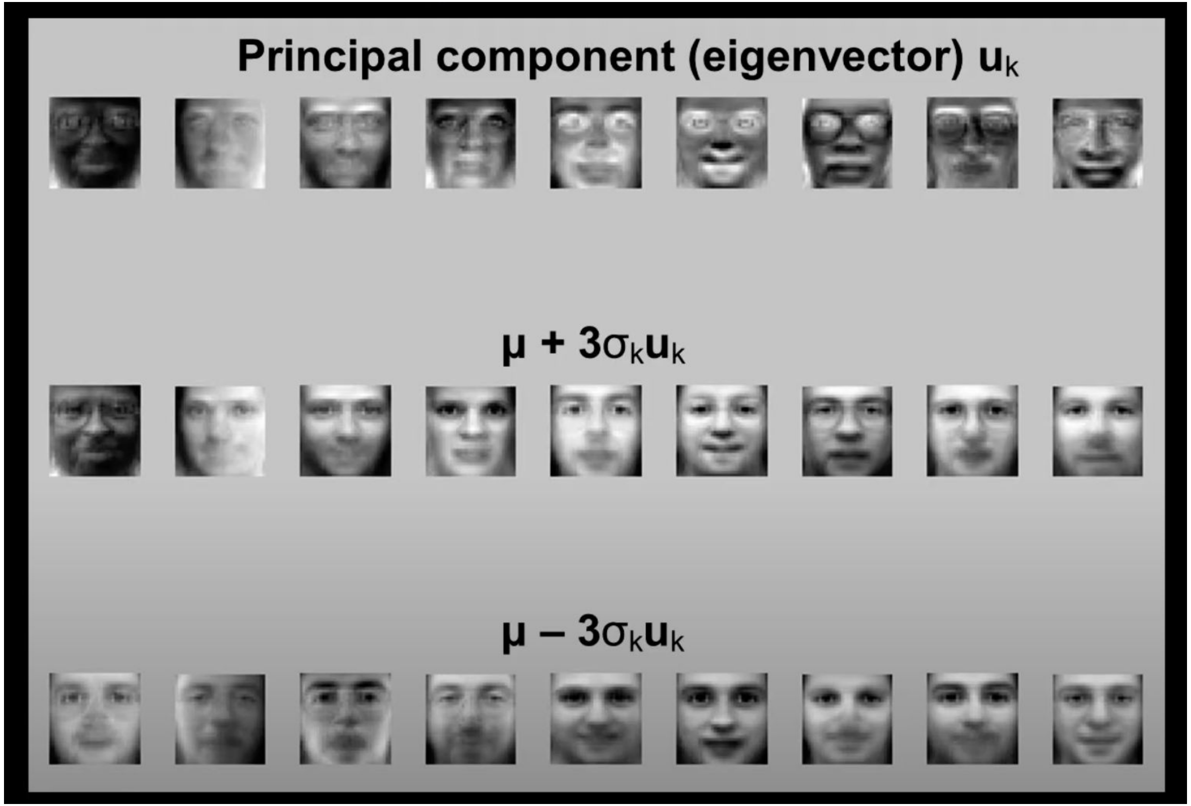
Eigenfaces Example

Training images
$x_1, \ldots, x_m$

Eigenfaces Example



Eigenfaces Example



Eigenfaces Example

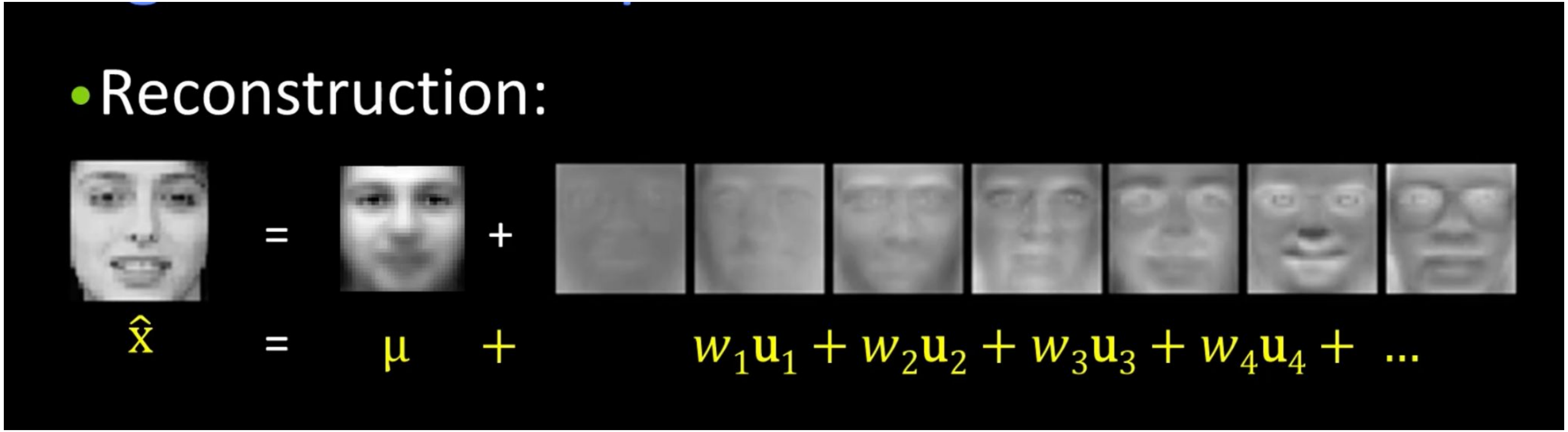
- Face **x** in "face space" coordinates (dot products) :

$$\mathbf{x} \rightarrow [\mathbf{u}_1^T(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^T(\mathbf{x}-\mu)]$$

$$= [w_1, \ldots w_k]$$

This vector is the representation of the face.

Eigenfaces Example



Recognition with Eigenfaces

Given novel image $\mathbf{x}$:

- Project onto subspace:

$$[w_1, \ldots, w_k] = [\mathrm{u}_1^T(\mathbf{x} - \mu), \ldots, \mathrm{u}_k^T(\mathbf{x} - \mu)]$$

- Classify as closest training face in k-dimensional subspace
- This is why it's a *generative* model.

Code Review