The Introduction To Artificial Intelligence

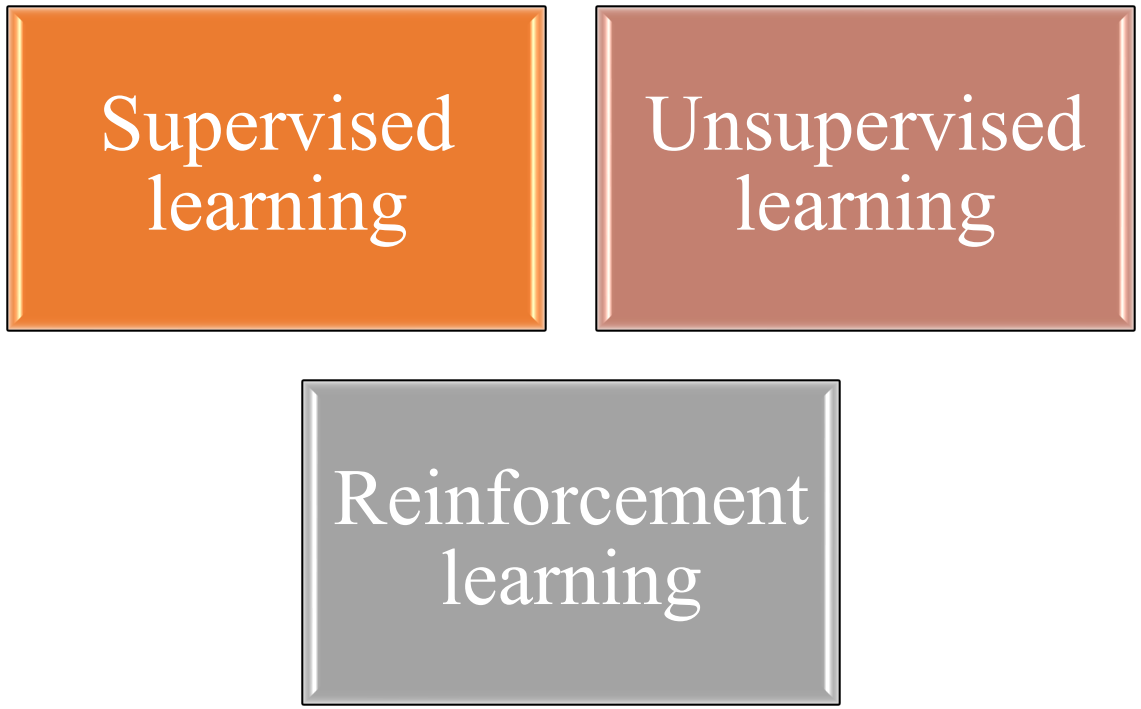
Yuni Zeng yunizeng@zstu.edu.cn

2022-2023-1

The Introduction to Artificial Intelligence

- Part I Brief Introduction to AI & Different AI tribes
- Part II Knowledge Representation & Reasoning
- Part III AI GAMES and Searching
- Part IV Model Evaluation and Selection
- Part V Machine Learning

Machine Learning

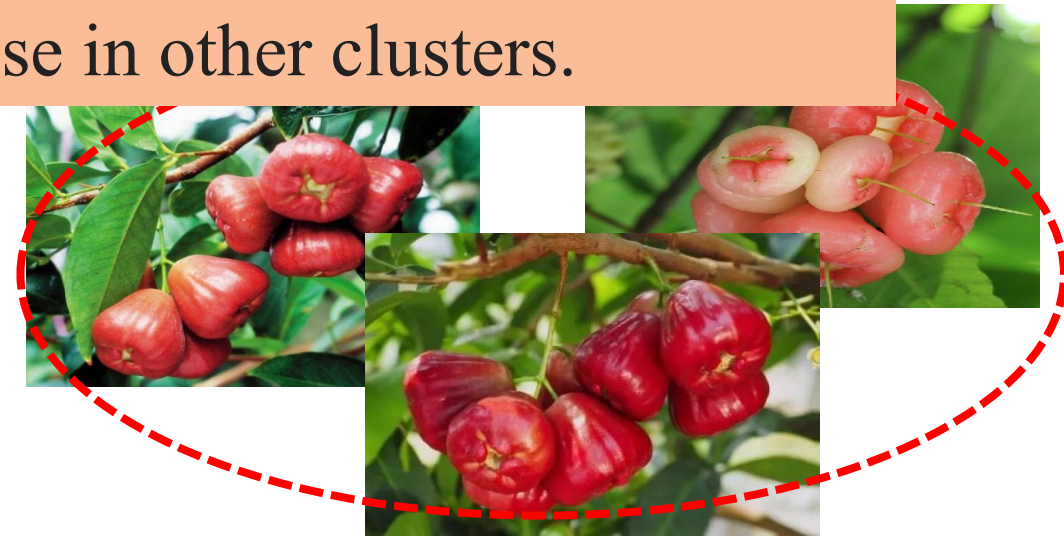


Unsupervised learning

- Clustering
  - K-means method
  - Spectral clustering
- Representation learning

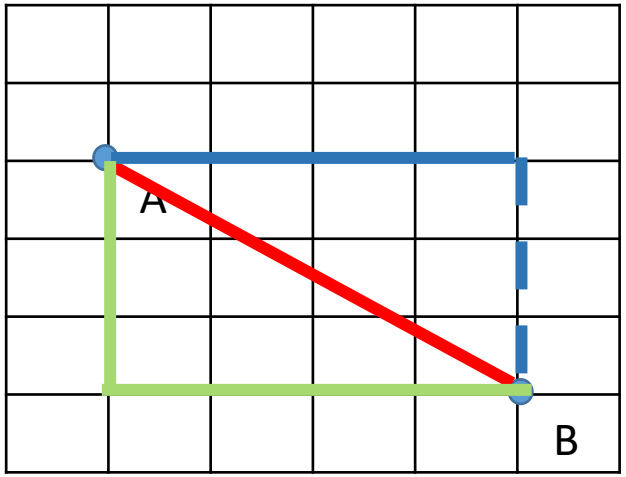
Clustering

Clustering: Grouping a set of data in such a way that data in the same cluster are more similar to each other than to those in other clusters.



Clustering

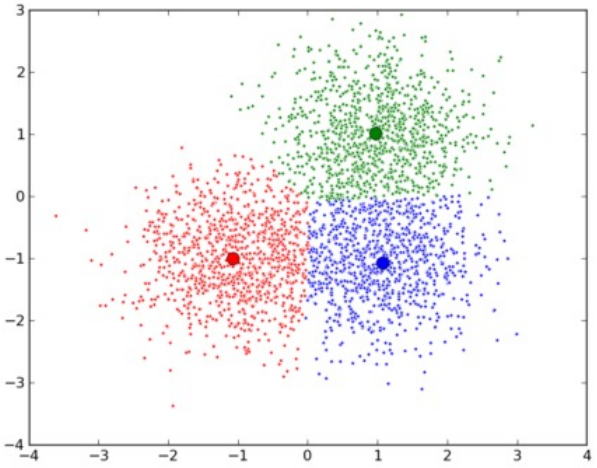
- Distance Metrics
  - Euclidean distance
  - $d_e(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$
  - Sum of squared distance
  - $d_q(x, y) = \sum_{i=1}^{n}(x_i - y_i)^2$
  - Manhattan distance
  - $d_m(x, y) = \sum_{i=1}^{n}|x_i - y_i|$
  - Chebyshev distance
  - $d_c(x, y) \max_{i=1,\cdots,n}|x_i - y_i|$



Clustering

K-means Clustering

- K-means clustering is a sort of clustering algorithm, and it is popular for cluster analysis in data mining.
- K-means clustering aims to partition N observations into K clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

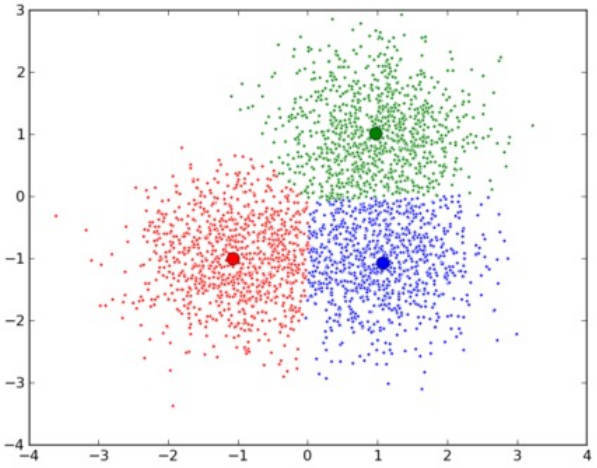


Clustering

K-means Clustering

- High intra-clustering similarity
- Low inter-clustering similarity.
- So,

$$object : \sum_{i=1}^{N} \min_{u_j \in C} \|x_i - u_j\|^2$$



Clustering

- K-means Clustering

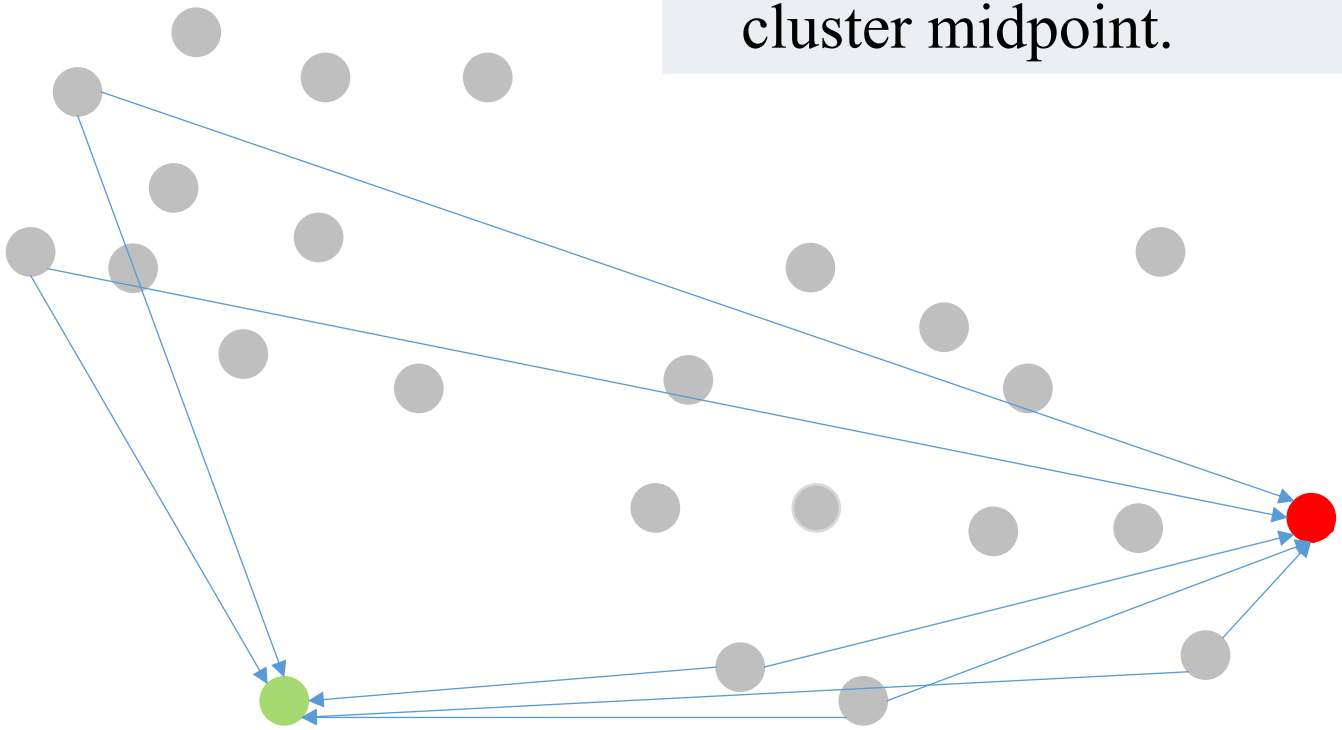
e.g. $k = 2$

First the k cluster midpoints $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ are randomly or manually initialized.

Clustering

- K-means Clustering

e.g. k =2



Then the following two steps are repeatedly carried out:

- Classify all data to their nearest cluster midpoint.

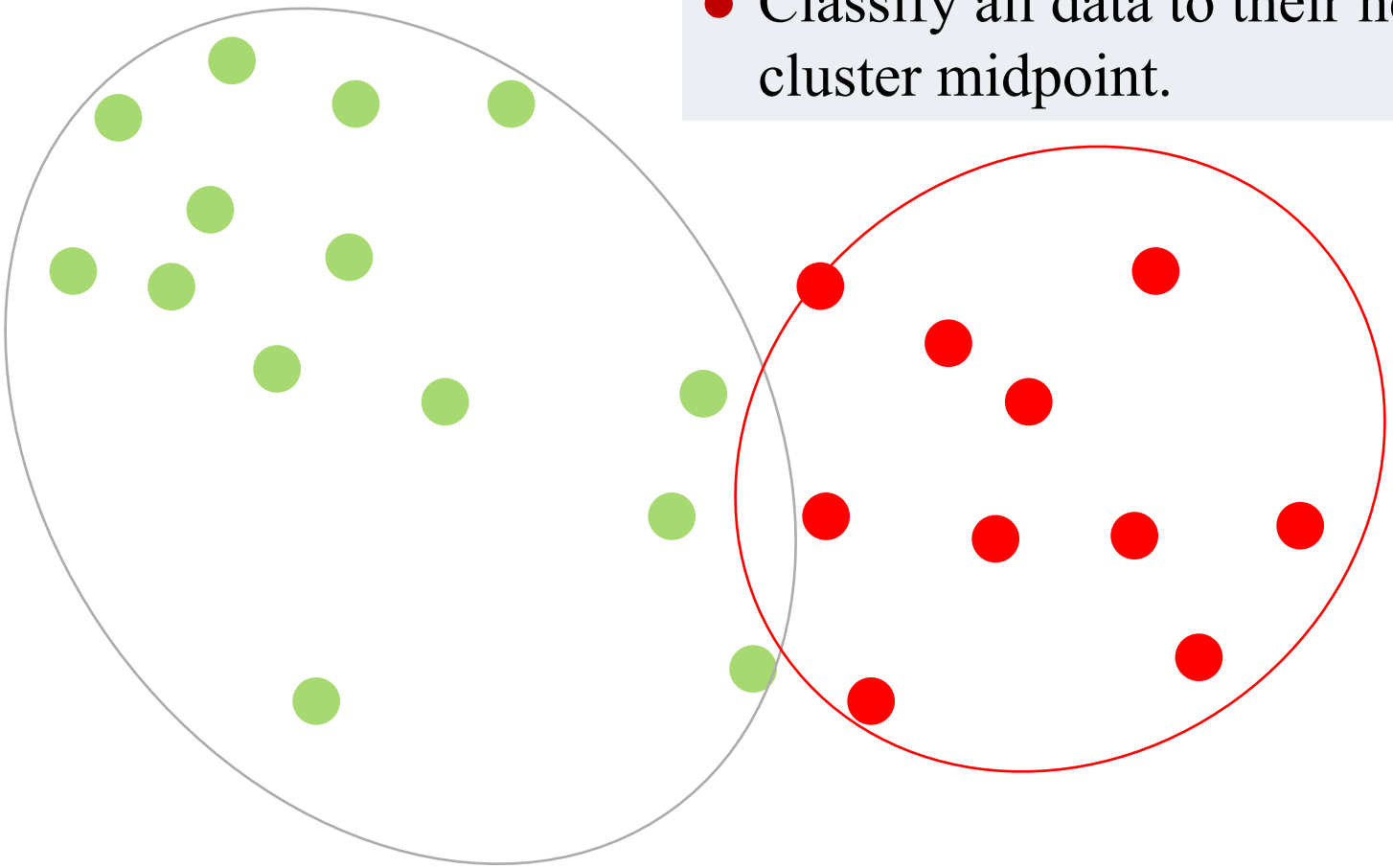
Clustering

- K-means Clustering

e.g. k =2

Then the following two steps are repeatedly carried out:

- Classify all data to their nearest cluster midpoint.



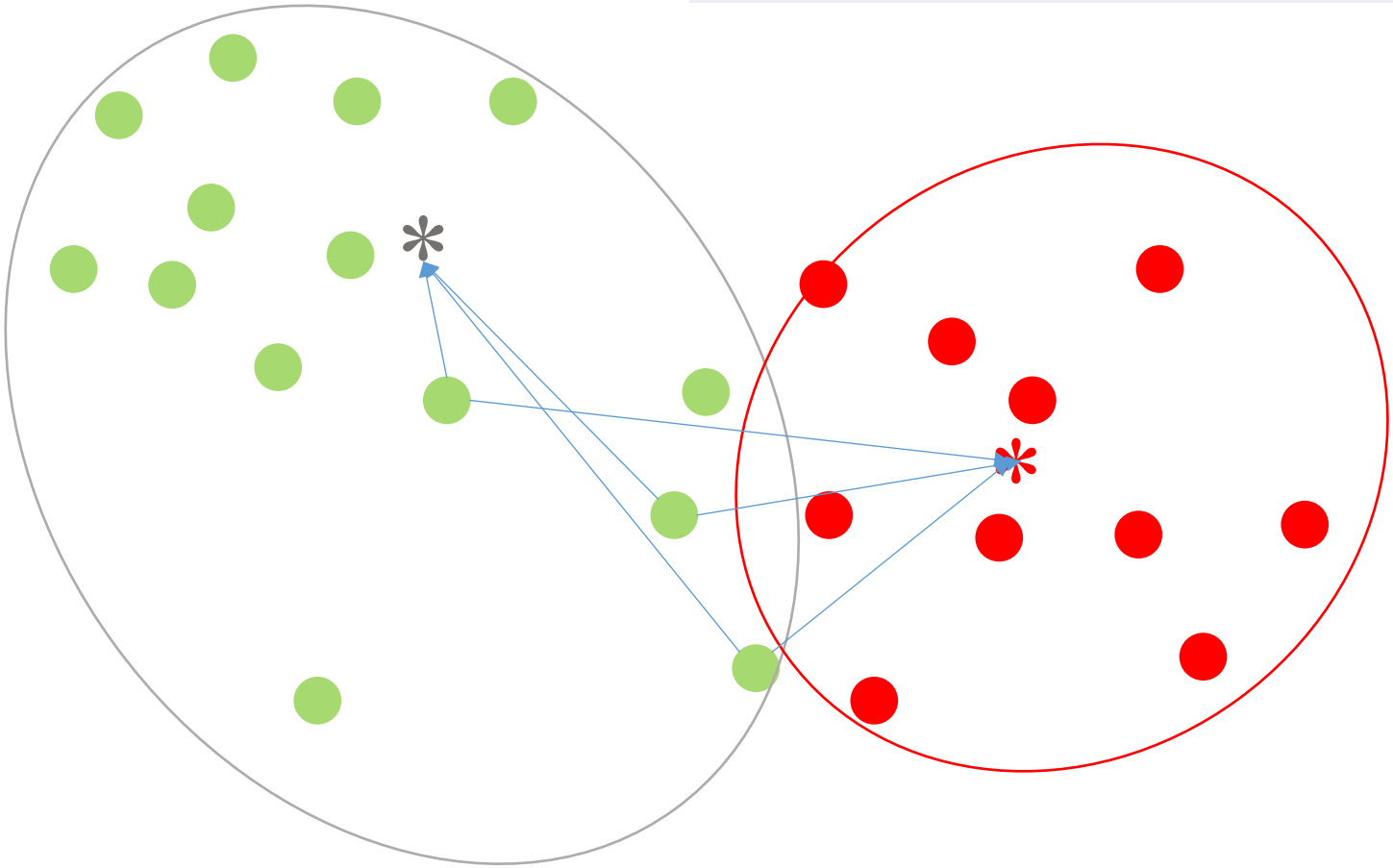
Clustering

- K-means Clustering

e.g. k =2

Then the following two steps are repeatedly carried out:

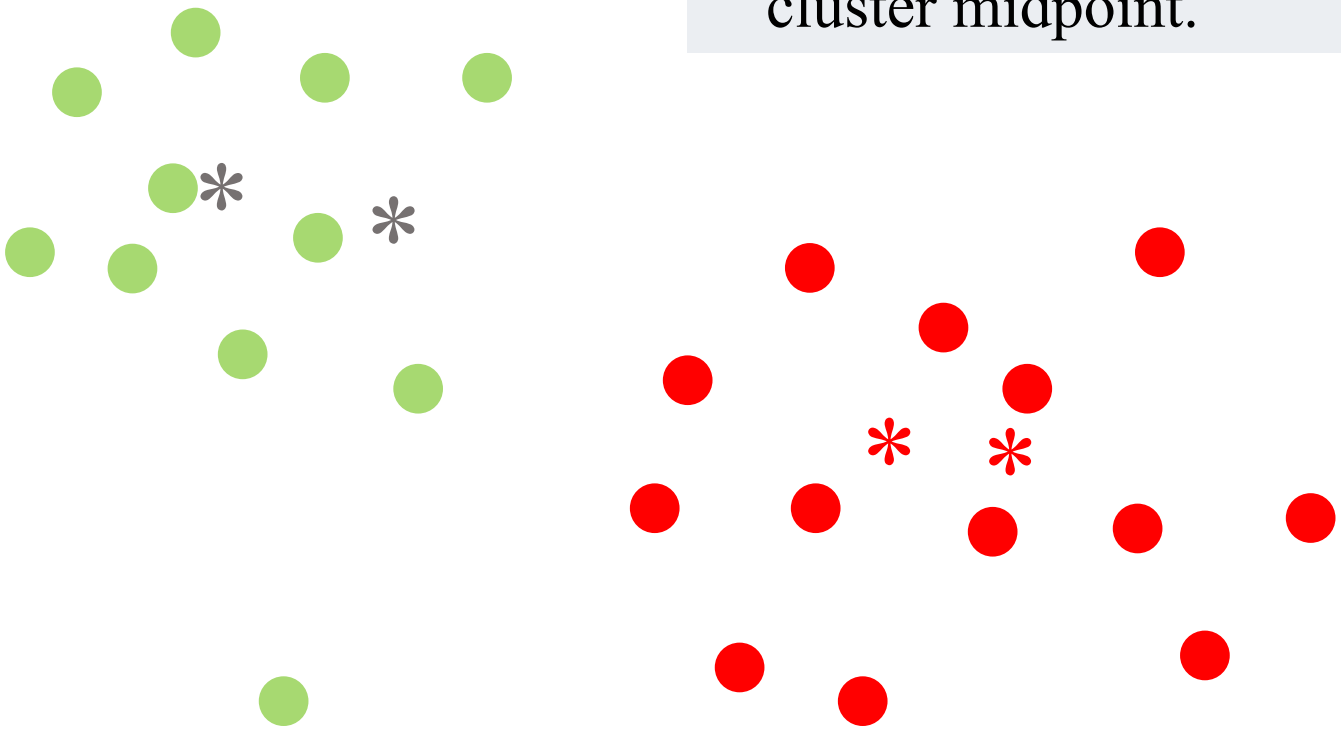
- Re-compute of the cluster midpoint.



Clustering

- K-means Clustering

e.g. $k = 2$



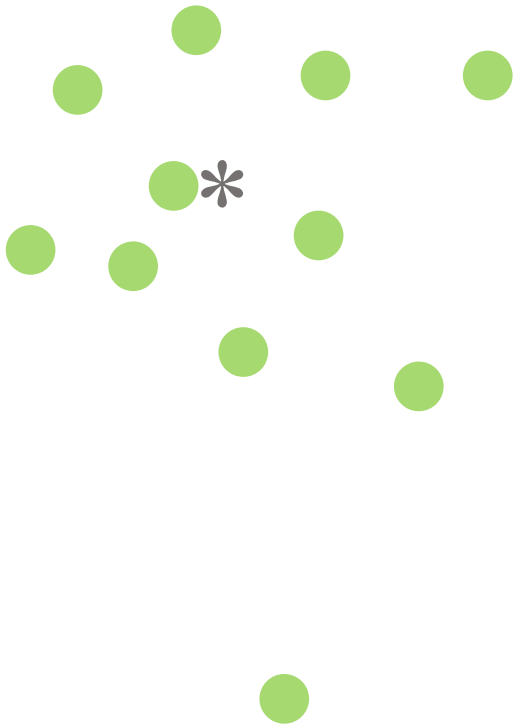
Then the following two steps are repeatedly carried out:

- Classify all data to their nearest cluster midpoint.

Clustering

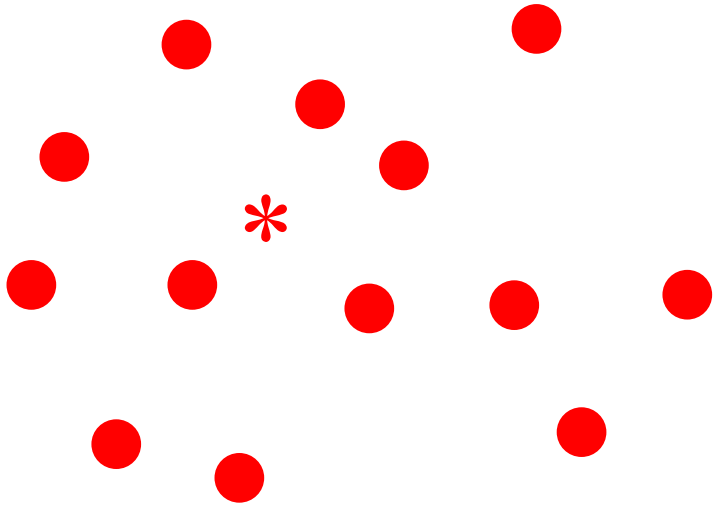
- K-means Clustering

e.g. $k = 2$



Then the following two steps are repeatedly carried out:

- Re-compute of the cluster midpoint.



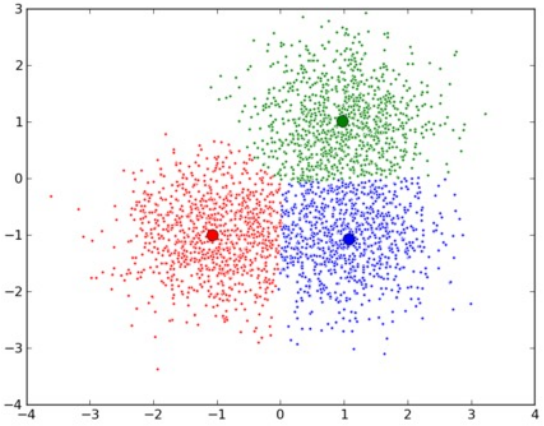
The algorithm converges

Clustering

K-means Clustering

The following two steps are repeatedly carried out:

- Initialize the midpoints
- Repeat the following 2 steps
  - Classify all data to their nearest cluster midpoint.
  - Re-compute of the cluster midpoint.
- Until the algorithm converges



Clustering

K-means Clustering -- Example

Try to cluster these samples X by k-means clustering, when k =2,

$$X = \begin{bmatrix} 0 & 0 & 1 & 5 & 5 \\ 2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

- Initialize the midpoints
- Repeat the following 2 steps
  - Classify all data to their nearest cluster midpoint.
  - Re-compute of the cluster midpoint.
- Until the algorithm converges

Clustering

K-means Clustering -- Example

Try to cluster these samples X by k-means clustering, when k =2,

$$X = \begin{bmatrix} 0 & 0 & 1 & 5 & 5 \\ 2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

Step 1: First the k cluster midpoints $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ are randomly or manually initialized.

→ Suppose $m_1^{(0)} = (0,2)^T$ *is the mindpoint of cluster* $G_1^{(0)}$, $m_2^{(0)} = (0,0)^T$ *is the mindpoint of cluster* $G_2^{(0)}$

Clustering

K-means Clustering -- Example

Try to cluster these samples X by k-means clustering, when k =2,

$$X = \begin{bmatrix} 0 & 0 & 1 & 5 & 5 \\ 2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

Step 2: Classify all data to their nearest cluster midpoint.

→ Calculate the distance from $x_3 = (1,0)^T, x_4 = (5,\ 0)^T, x_5 = (5,2)^{\mathrm{T}}$ to the midpoints $m_1^{(0)}, m_2^{(0)}$:

$x_3 = (1,0)^T, d\left(x_3, m_1^{(0)}\right) = 5, d\left(x_3, m_2^{(0)}\right) = 1,$ So x_3 is $G_2^{(0)}$.

$x_4 = (5,5)^T, d\left(x_4, m_1^{(0)}\right) = 29, d\left(x_3, m_2^{(0)}\right) = 25,$ So x_4 is $G_2^{(0)}$.

$x_5 = (5,2)^{\mathrm{T}}, d\left(x_5, m_1^{(0)}\right) = 25, d\left(x_3, m_2^{(0)}\right) = 29,$ So x_5 is $G_1^{(0)}$.

Clustering

K-means Clustering -- Example

Try to cluster these samples X by k-means clustering, when k =2,

$$X = \begin{bmatrix} 0 & 0 & 1 & 5 & 5 \\ 2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

Step 3: New cluster $G_1^{(1)} = \{x_1, x_5\}$ and $G_2^{(1)} = \{x_2, x_3, x_4\}$. So, re-compute of the cluster midpoint.

$\rightarrow m_1^{(1)} = (2.5, 2.0)^T$ *is the mindpoint of cluster* $G_1^{(1)}$,

$m_2^{(1)} = (2, 0)^T$ *is the mindpoint of cluster* $G_2^{(1)}$

Clustering

K-means Clustering -- Example

Try to cluster these samples X by k-means clustering, when k =2,

$$X = \begin{bmatrix} 0 & 0 & 1 & 5 & 5 \\ 2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

Step 4: Repeat step 2 and step 3

→ Have new clusters $G_1^{(2)} = \{x_1, x_5\}$ and $G_2^{(2)} = \{x_2, x_3, x_4\}$.

Because the clusters is not change, the clustering stops! The final results is:

$$G_1^* = \{x_1, x_5\} \text{ and } G_2^* = \{x_2, x_3, x_4\}.$$

Clustering

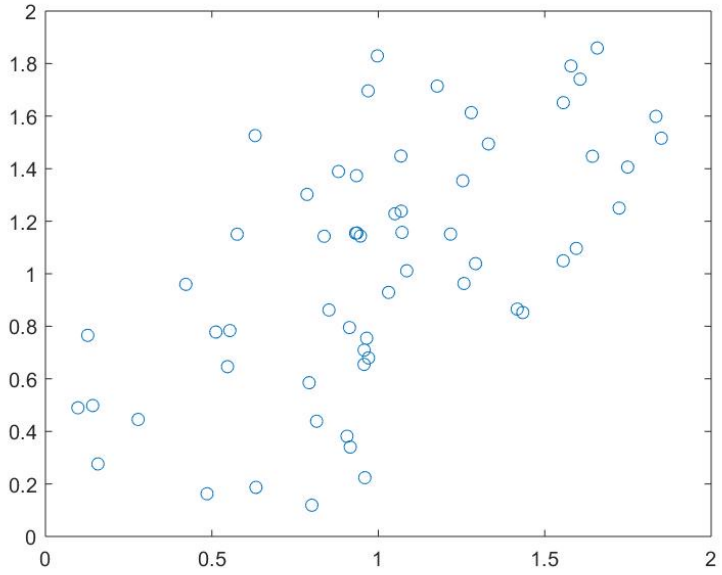
- K-means Clustering --- Example

X

2x60 double

	1	2	3	4	5	6	7	8	9	10
1	0.8147	0.9058	0.1270	0.9134	0.6324	0.0975	0.2785	0.5469	0.9575	0.9649
2	0.4387	0.3816	0.7655	0.7952	0.1869	0.4898	0.4456	0.6463	0.7094	0.7547

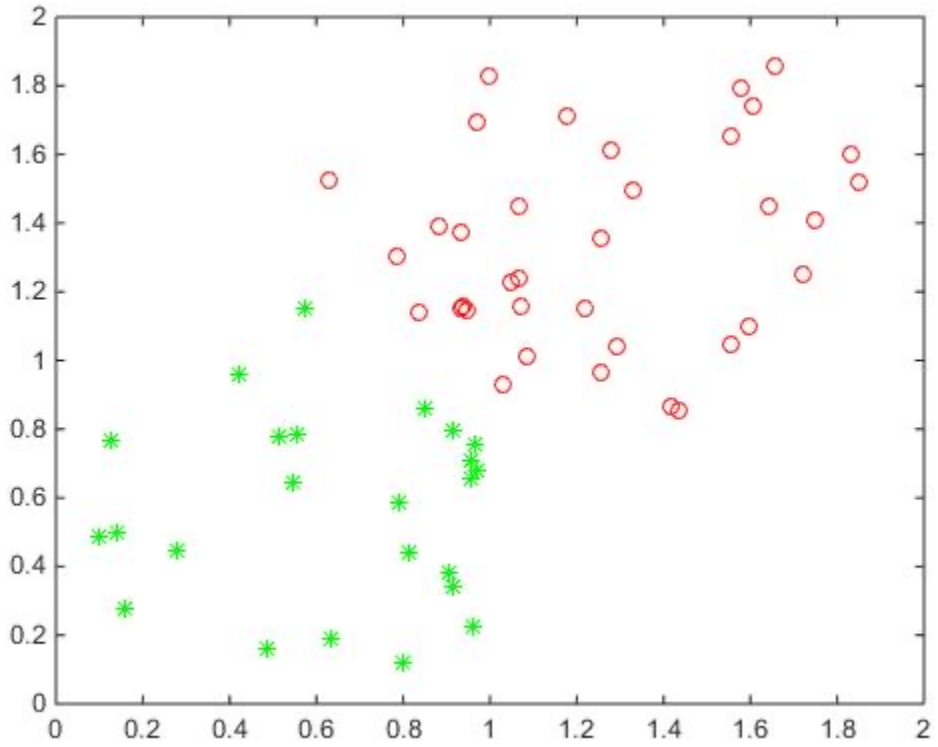
```
plot(X(1,:),X(2,:),'o')
```



Clustering

- K-means Clustering --- Example

```
IDX = kmeans(X,2);
…… % plot the results
```



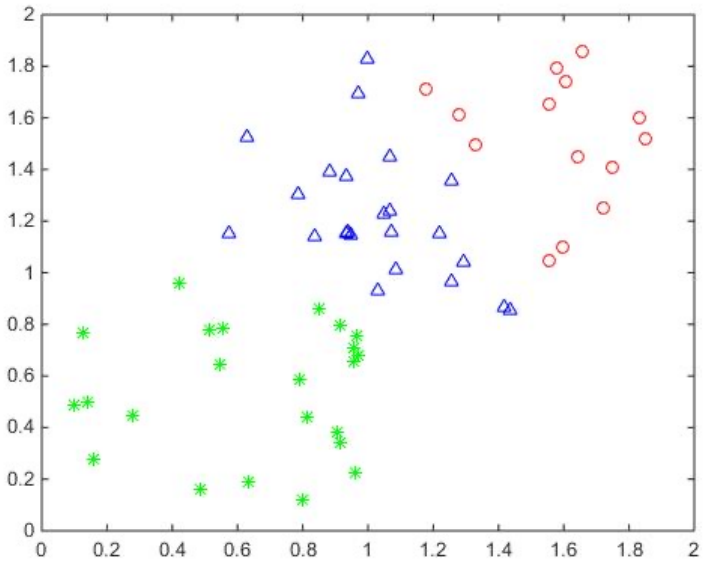
Clustering

K-means Clustering --- Example

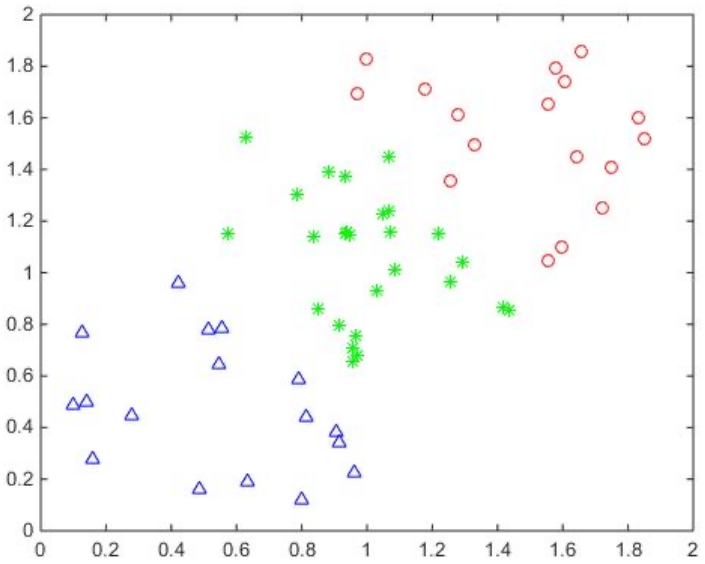
```
IDX = kmeans(X,3);
…… % plot the results
```

The results are different!
Why?

Result 1

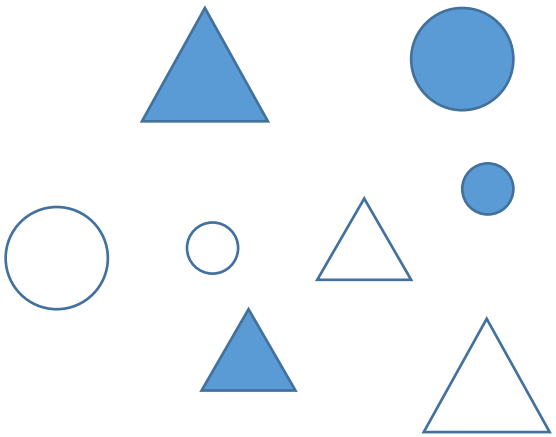


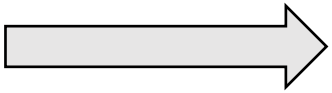
Result 2



Clustering

- K-means Clustering -Example





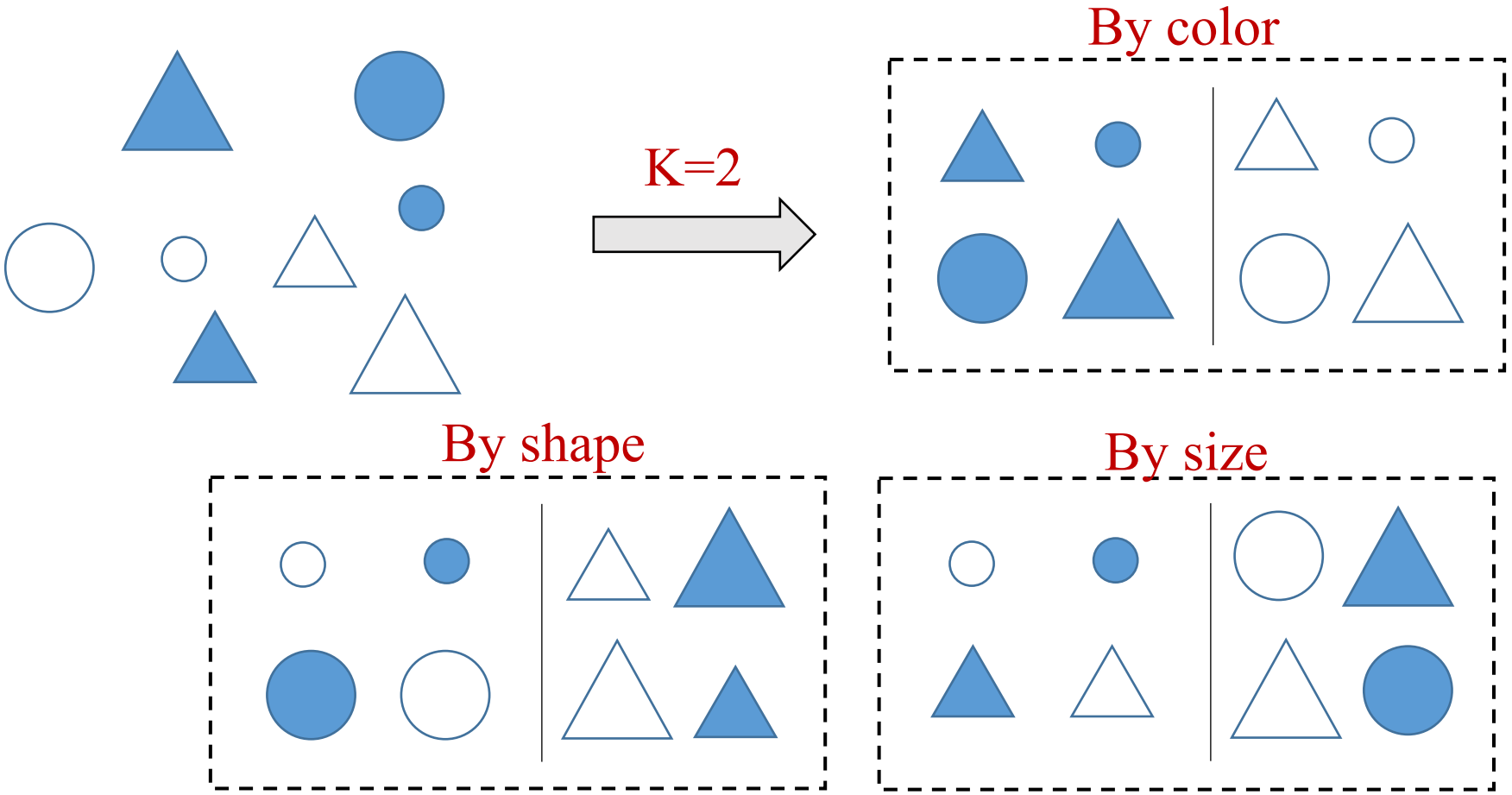
Clustering

K=2?

K=4?

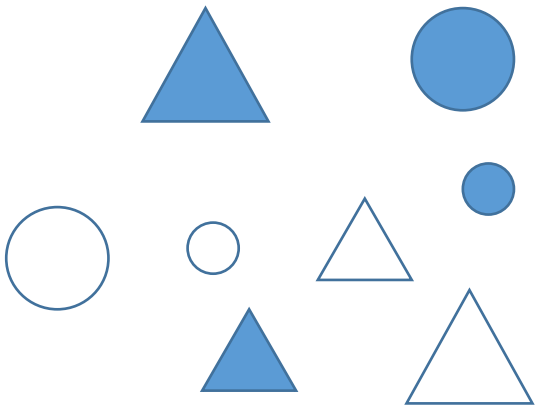
Clustering

- K-means Clustering -Example

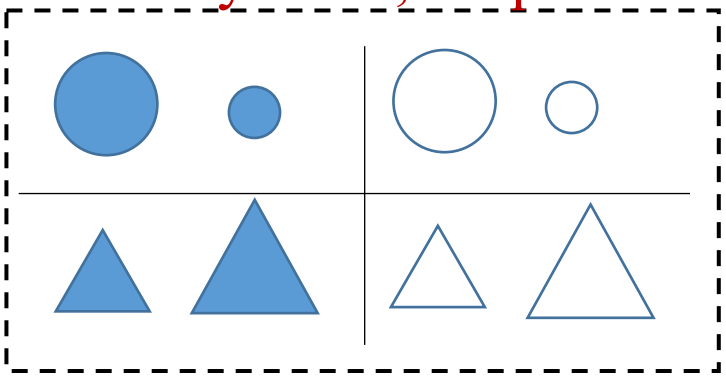


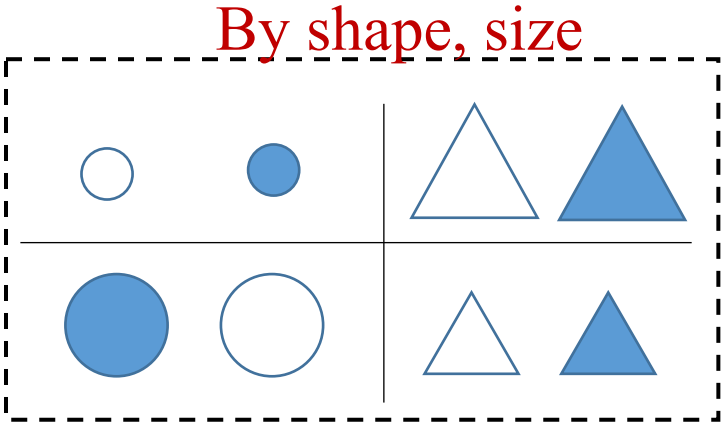
Clustering

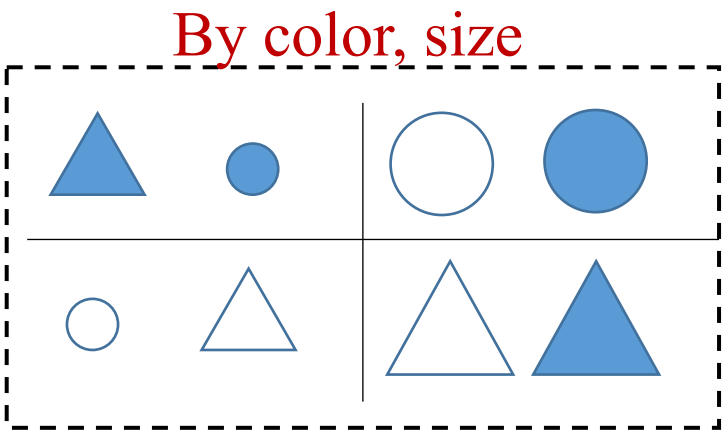
- K-means Clustering -Example



K=4







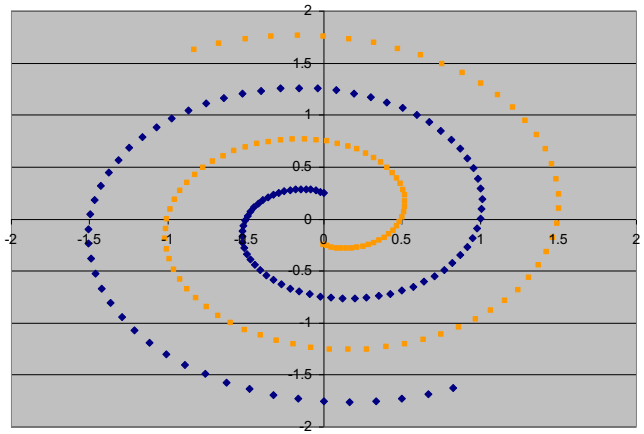
Clustering

K-means Clustering --- Disadvantage

- 1. Time complexity:
  - O(NKT), where N is the number of data, K is the number of clusters, and T is the number of iterations.
- 2. Sensitive to noise
- 3. Different results are obtained with different initial centers.

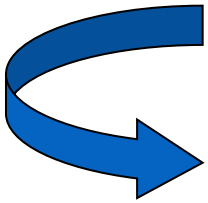
Clustering

K-means Clustering --- Example

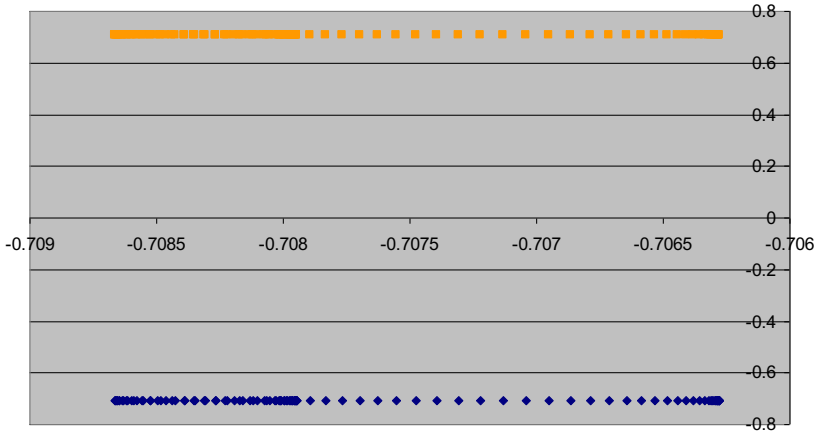


Dataset exhibits complex cluster shapes

⇒ K-means performs very poorly in this space due bias toward dense spherical clusters.



In the embedded space given by two leading eigenvectors, clusters are trivial to separate.



Clustering

- Spectral Clustering
  - Algorithms that cluster points using eigenvectors of matrices derived from the data.
  - Obtain data representation in the low-dimensional space that can be easily clustered.
  - Variety of methods that use the eigenvectors differently
  - Disadvantage: difficult to understand….

Clustering

Spectral Clustering

- Three basic stages:
  1. Pre-processing
     - Construct a matrix representation of the dataset.
  2. Decomposition
     - Compute eigenvalues and eigenvectors of the matrix.
     - Map each point to a lower-dimensional representation based on one or more eigenvectors.
  3. Grouping
     - Assign points to two or more clusters (e.g. by k means method), based on the new representation.

Clustering

- Spectral Clustering

1 . Pre-processing

- Construct a matrix representation of the dataset. (Build Laplacian matrix ***L)***

	x_1	x_2	x_3	x_4	x_5	x_6
x_1	1.5	-0.8	-0.6	0	-0.1	0
x_2	-0.8	1.6	-0.8	0	0	0
x_3	-0.6	-0.8	1.6	-0.2	0	0
x_4	0	0	-0.2	1.7	-0.8	-0.7
x_5	-0.1	0	0	-0.8	1.7	-0.8
x_6	0	0	0	-0.7	-0.8	1.5

2. Decomposition

- Find eigenvalues Λ and eigenvectors X of the matrix ***L***

- Map vertices to corresponding components of smallest two eigenvalues.

$\Lambda =$

0.0
0.4
2.2
2.3
2.5
3.0

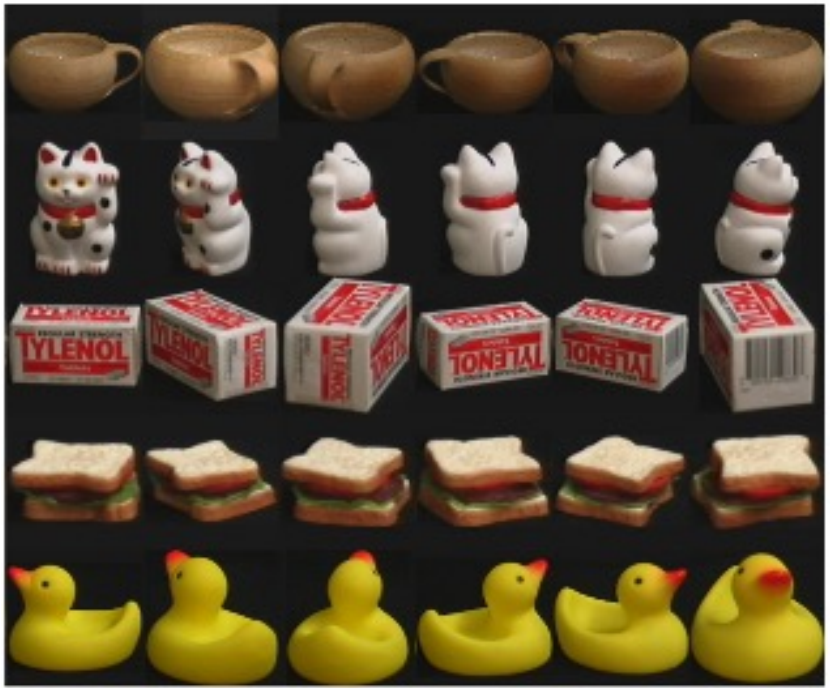
$X =$

0.4	0.2	0.1	0.4	-0.2	-0.9
0.4	0.2	0.1	-0.	0.4	0.3
0.4	0.2	-0.2	0.0	-0.2	0.6
0.4	-0.4	0.9	0.2	-0.4	-0.6
0.4	-0.7	-0.4	-0.8	-0.6	-0.2
0.4	-0.7	-0.2	0.5	0.8	0.9

It is easier to divide these six points into two clusters using this new representation.

Clustering

- Spectral Clustering—Applications: image clustering



Clustering

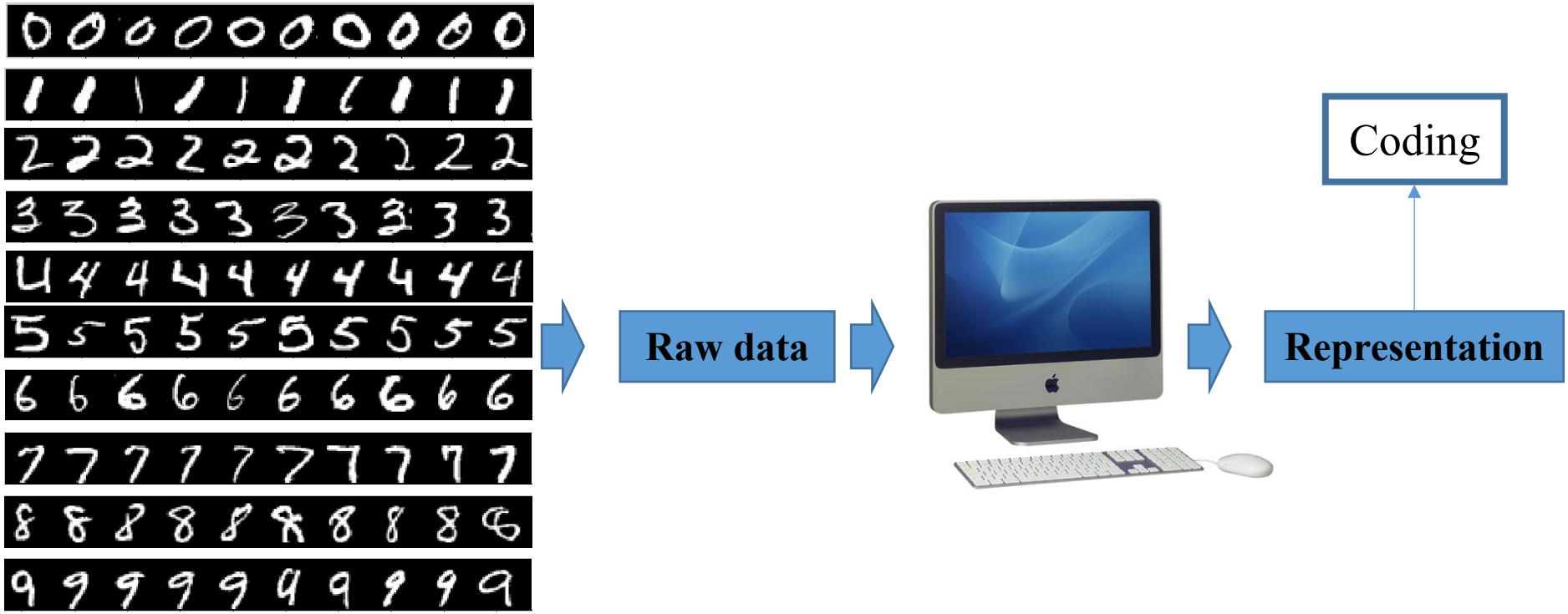
- Spectral Clustering ---Applications: Motion segmentation

Unsupervised learning

- Clustering
  - K-means method
  - Spectral clustering
- Representation learning

Representation Learning

- Representation = re + presentation



Unsupervised learning

- Clustering
  - K-means method
  - Spectral clustering
- Representation learning
  - *Linear coding*
  - PCA
  - Applications

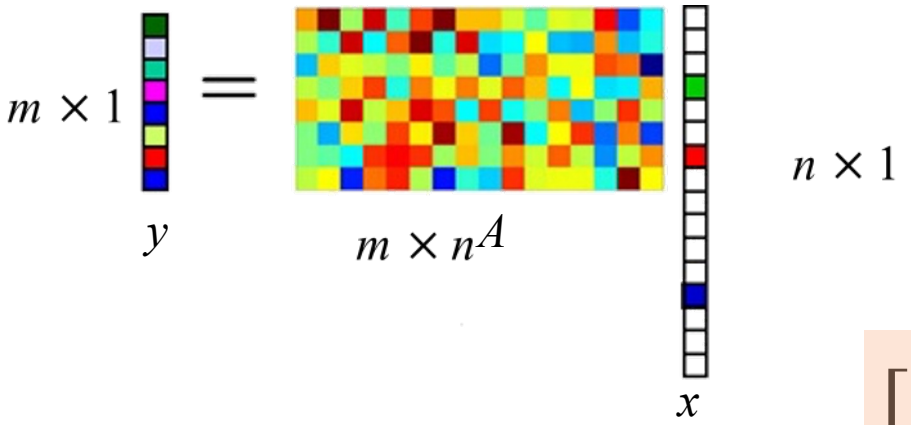
Representation Learning

- Linear coding

representation

A: "dictionary"

$$y = Ax \quad \in \mathbb{R}^m$$



e.g.

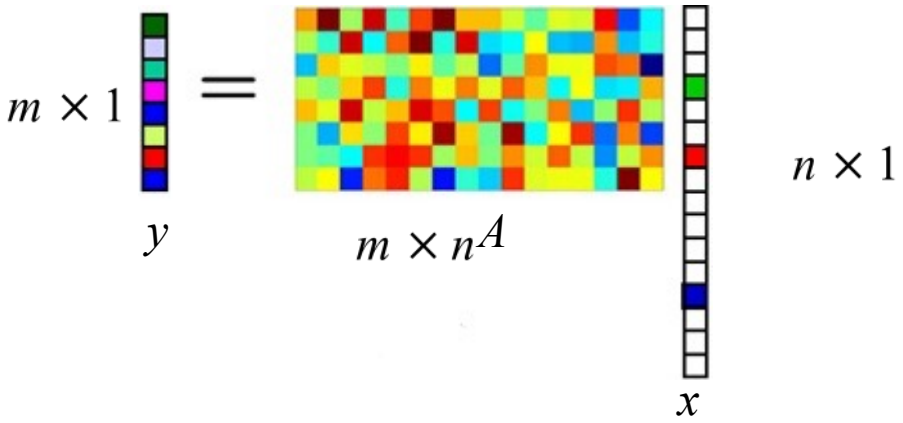
$$y = \begin{bmatrix} 0.75 \\ 0.27 \\ 0.3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0.5 & 0.1 & 0 & 0 \\ 0.2 & 0.1 & 0 & 0.1 & 1 & 0 \\ 0 & 0.1 & 1 & 0.2 & 0.1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.6 \\ 0 \\ 0 \\ 1.5 \\ 0 \\ 0 \end{bmatrix}$$

Representation Learning

- Linear coding

representation

$$y = Ax \in R^m$$



Considering the solution to the linear equation:

If m>n => overdetermined => no solution / unique solution

If m<n => underdetermined => no solution / infinite solutions

Unsupervised learning

- Clustering
  - K-means method
  - Spectral clustering
- Representation learning
  - Linear coding
  - *PCA*
  - Applications

Representation Learning

- Linear Feature Extraction

➢ Given the original d-dimension feature space $X = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_m) \in \mathbb{R}^{d \times m}$

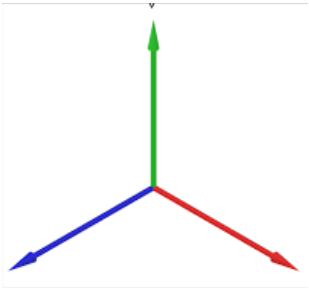
➢ Get the reduced d'-dimension feature space $Z = (\boldsymbol{z}_1, \boldsymbol{z}_2, \dots, \boldsymbol{z}_m) \in \mathbb{R}^{d' \times m}$ after transformation ($d' < d$)

➢ Transformation process:

$$Z = \boldsymbol{W}^T X$$

Where $\boldsymbol{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}) \in \mathbb{R}^{d \times d'}$ is the transformation matrix, $\boldsymbol{w}_i \in \mathbb{R}^{d \times 1}$, and $Z \in \mathbb{R}^{d' \times m}$ is the coordinate expression of sample X in low dimension space.

➢ If $\boldsymbol{w}_i^T \boldsymbol{w}_j = 0 (\mathrm{i} \neq j)$, then $\boldsymbol{w}_i$ is orthogonal to $\boldsymbol{w}_j$ ($\boldsymbol{w}_i$ is independent from $\boldsymbol{w}_j$), the new coordinate system $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}\}$ is orthogonal, and $\boldsymbol{W}$ is the orthogonal matrix.



Representation Learning

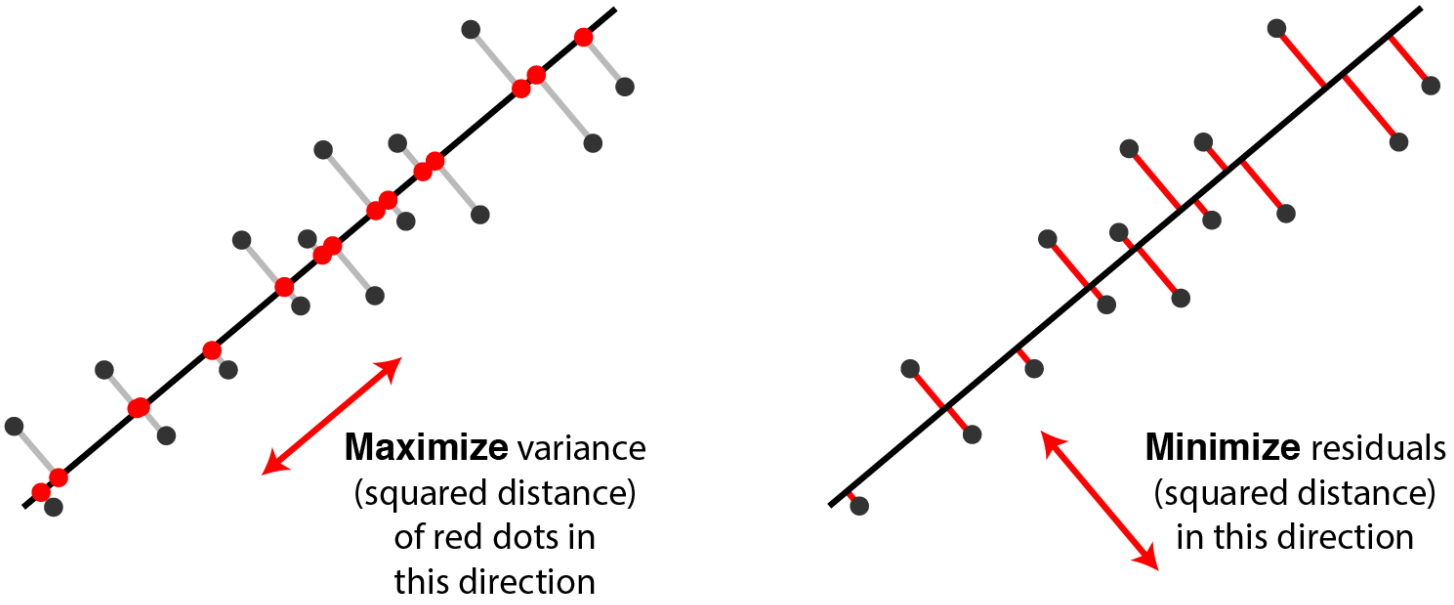
Principal Component Analysis (PCA)

- Linear combination of original features: use the orthogonal transform matrix $\boldsymbol{W}$ to transform the d relevant features into d' $(d' < d)$ irrelevant features. These d' irrelevant features are called principal components for classification.
- Use principal components to approximate the original sample.
- Realize the dimension reduction by replace original sample using few principal components.

Representation Learning

Principal Component Analysis (PCA)

- **Minimize reconstruction error (residual):** the sample $\tilde{\boldsymbol{x}}$ reconstructed from the reduced (projected) space is close enough to the original sample $\boldsymbol{x}$.
- **Maximum class separability (variance):** ensure that projected data from different classes can be separated well.



Representation Learning

Principal Component Analysis (PCA)

- Minimize reconstruction error

➢ Data standardization. Centralization the original data (subtract mean vector). That is, set $\boldsymbol{x}_i = \boldsymbol{x}_i - \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_i$, then $\sum_{i=1}^{m} \boldsymbol{x}_i = 0$, $\boldsymbol{x}_i \in \mathbb{R}^{d\times 1}$.

➢ Assume transformation matrix $\boldsymbol{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}) \in \mathbb{R}^{d\times d'}$, $\boldsymbol{w}_i \in \mathbb{R}^{d\times 1}$ is the standard orthogonal basis vector. That is, $\|\boldsymbol{w}_i\|_2 = 1, \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}$, $\boldsymbol{w}_i^T\boldsymbol{w}_j = 0 (i \neq j)$.

➢ The projection of $\boldsymbol{x}_i$ in low dimension coordinate system is $\boldsymbol{z}_i = \boldsymbol{W}^T\boldsymbol{x}_i$, $\boldsymbol{z}_i = (z_{i1}; z_{i2}; \dots; z_{id'}) \in \mathbb{R}^{d'\times 1}$. Where $z_{ij} = \boldsymbol{w}_j^T\boldsymbol{x}_i$ is the jth-coordinate of $\boldsymbol{x}_i$ in low dimension space.

➢ Reconstruct $\boldsymbol{x}_i$ by $\boldsymbol{z}_i$, then $\tilde{\boldsymbol{x}}_i = \sum_{j=1}^{d'} z_{ij}\boldsymbol{w}_j$

Minimize reconstruction error : $\tilde{\boldsymbol{x}}_i$, $\boldsymbol{x}_i$

Representation Learning

Principal Component Analysis (PCA)

- Minimize reconstruction error

➢For the entire training set, the distance between original sample $\boldsymbol{x}_i$ and reconstructed sample $\tilde{\boldsymbol{x}}_i$ is

$$\sum_{i=1}^{m} \|\tilde{\boldsymbol{x}}_i - \boldsymbol{x}_i\|_2^2 = -\sum_{i=1}^{m} \boldsymbol{z}_i^T \boldsymbol{z}_i + \sum_{i=1}^{m} \boldsymbol{x}_i^T \boldsymbol{x}_i$$

$$\propto -tr\left(\boldsymbol{W}^T \left(\sum_{i=1}^{m} \boldsymbol{x}_i \boldsymbol{x}_i^T\right) \boldsymbol{W}\right)$$

$$tr(A) = \sum_{i=1}^{n} a_{ii}$$

➢Take minimizing the reconstruction distance as the objective function, since $\sum_{i=1}^{m} \boldsymbol{x}_i \boldsymbol{x}_i^T$ is the covariance matrix $\boldsymbol{X}\boldsymbol{X}^T$, then the objective function is

$$\min_{\boldsymbol{W}} -tr(\boldsymbol{W}^T \boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W})$$
$$s.t. \boldsymbol{W}^T \boldsymbol{W} = \boldsymbol{I}$$

Representation Learning

Principal Component Analysis (PCA)

- Proof

$$\sum_{i=1}^{m}\|\hat{\boldsymbol{x}}_i-\boldsymbol{x}_i\|_2^2=\sum_{i=1}^{m}\left\|\sum_{j=1}^{d'}z_{ij}\boldsymbol{w}_j-\boldsymbol{x}_i\right\|_2^2=\sum_{i=1}^{m}\|\boldsymbol{W}\boldsymbol{z}_i-\boldsymbol{x}_i\|_2^2$$

$$=\sum_{i=1}^{m}(\boldsymbol{W}\boldsymbol{z}_i-\boldsymbol{x}_i)^T(\boldsymbol{W}\boldsymbol{z}_i-\boldsymbol{x}_i)$$

$$=-\sum_{i=1}^{m}\boldsymbol{z}_i^T\boldsymbol{z}_i+\sum_{i=1}^{m}\boldsymbol{x}_i^T\boldsymbol{x}_i$$

$$=-\sum_{i=1}^{m}tr\left(\boldsymbol{z}_i\boldsymbol{z}_i^T\right)+\text{const}$$

$$=-tr\left(\sum_{i=1}^{m}\boldsymbol{z}_i\boldsymbol{z}_i^T\right)+\text{const}$$

$$=-tr\left(\sum_{i=1}^{m}\boldsymbol{W}^T\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{W}\right)+\text{const}$$

$$\approx-tr\left(\boldsymbol{W}^T\left(\sum_{i=1}^{m}\boldsymbol{x}_i\boldsymbol{x}_i^T\right)\boldsymbol{W}\right)+\text{const}$$

$$\propto-tr\left(\boldsymbol{W}^T\left(\sum_{i=1}^{m}\boldsymbol{x}_i\boldsymbol{x}_i^T\right)\boldsymbol{W}\right)$$

Representation Learning

- Principal Component Analysis (PCA)
  - Maximum class separability

$$\max_{\boldsymbol{W}} tr(\boldsymbol{W}^T \boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W})$$
$$s.t. \boldsymbol{W}^T \boldsymbol{W} = \boldsymbol{I}$$

$$\min_{\boldsymbol{W}} -tr(\boldsymbol{W}^T \boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W})$$
$$s.t. \boldsymbol{W}^T \boldsymbol{W} = \boldsymbol{I}$$

Representation Learning

Principal Component Analysis (PCA)

Solve PCA to get $\boldsymbol{W = (w_1, w_2, \dots, w_{d'})}$

$$\min_{\boldsymbol{W}} -tr(\boldsymbol{W}^T \boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W})$$
$$s.t. \boldsymbol{W}^T \boldsymbol{W} = \boldsymbol{I}$$

Where $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_m) \in \mathbb{R}^{d \times \boldsymbol{m}}, \boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}\} \in \mathbb{R}^{d \times d'}, \boldsymbol{I} \in \mathbb{R}^{d' \times d'}$

- **S1.** Define the Lagrange function using Lagrange multiplier matrix $\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2, \dots, \lambda_{d'}) \in \mathbb{R}^{d' \times d'}$, $\boldsymbol{\Lambda}$ is the diagonal matrix.

$$L(\boldsymbol{W}, \boldsymbol{\Lambda}) = -tr(\boldsymbol{W}^T \boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W}) + tr\left(\boldsymbol{\Lambda}^T (\boldsymbol{W}^T \boldsymbol{W} - \boldsymbol{I})\right)$$

- **S2.** Set the partial of $L(\boldsymbol{W}, \boldsymbol{\Lambda})$ on $\boldsymbol{W}$ as 0.

$$\frac{\partial L(\boldsymbol{W}, \boldsymbol{\Lambda})}{\partial \boldsymbol{W}} = -2\boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W} + 2\boldsymbol{W}\boldsymbol{\Lambda} = \boldsymbol{0}$$

$$\boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W} = \boldsymbol{W}\boldsymbol{\Lambda}$$

$$\boldsymbol{X}\boldsymbol{X}^T \boldsymbol{w}_i = \lambda_i \boldsymbol{w}_i$$

The definition of eigenvalue and eigenvector for matrix.
Then $\boldsymbol{X}\boldsymbol{X}^T = \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^T$

Solve eigenvalue $\boldsymbol{\Lambda}$ and corresponding eigenvector $\boldsymbol{W}$ for matrix $\boldsymbol{X}\boldsymbol{X}^T$

Representation Learning

Principal Component Analysis (PCA)

Solve PCA to get $\boldsymbol{W = (w_1, w_2, \dots, w_{d'})}$

$$\min_{\boldsymbol{W}} -tr(\boldsymbol{W}^T \boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W})$$
$$s.t. \boldsymbol{W}^T \boldsymbol{W} = \boldsymbol{I}$$

Where $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_m) \in \mathbb{R}^{d \times \boldsymbol{m}}, \boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}\} \in \mathbb{R}^{d \times d'}, \boldsymbol{I} \in \mathbb{R}^{d' \times d'}$

- **S3.** Substitute $\boldsymbol{X}\boldsymbol{X}^T \boldsymbol{w}_i = \lambda_i \boldsymbol{w}_i$ in the objective function

$$\min_{\boldsymbol{W}} -tr(\boldsymbol{W}^T \boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W}) = \max_{\boldsymbol{W}} tr(\boldsymbol{W}^T \boldsymbol{X}\boldsymbol{X}^T \boldsymbol{W}) = \max_{\boldsymbol{W}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{d'}} \boldsymbol{w}_i^T \boldsymbol{X}\boldsymbol{X}^T \boldsymbol{w}_i$$
$$= \max_{\boldsymbol{W}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{d'}} \boldsymbol{w}_i^T \lambda_i \boldsymbol{w}_i = \max_{\boldsymbol{W}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{d'}} \lambda_i \boldsymbol{w}_i^T \boldsymbol{w}_i = \max_{\boldsymbol{W}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{d'}} \lambda_i$$

- **S4.** We will get the optimal solution by letting $\lambda_1, \lambda_2, \dots, \lambda_{d'}$ and $\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}$ be the top maximum d' eigenvalue and corresponding eigenvector.
- **S5.** Solve $\lambda_1, \lambda_2, \dots, \lambda_{d'}$ and $\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}$ by eigenvalue decomposing the covariance matrix $\boldsymbol{X}\boldsymbol{X}^T$. Rank the eigenvalue $\lambda_1 > \lambda_2 > \cdots > \lambda_d$, and take the top maximum d' eigenvalues and its corresponding eigenvectors. And compose the transformation matrix $\boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}\}$, which is the solution of PCA.

Representation Learning

Principal Component Analysis (PCA)

- d' can be specified by user.
- Or specified the minimum d' by setting the threshold $t(80\%\ or\ 90\%)$, which satisfies the following inequivalent relation

$$\frac{\sum_{i=1}^{d'} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \geq t$$

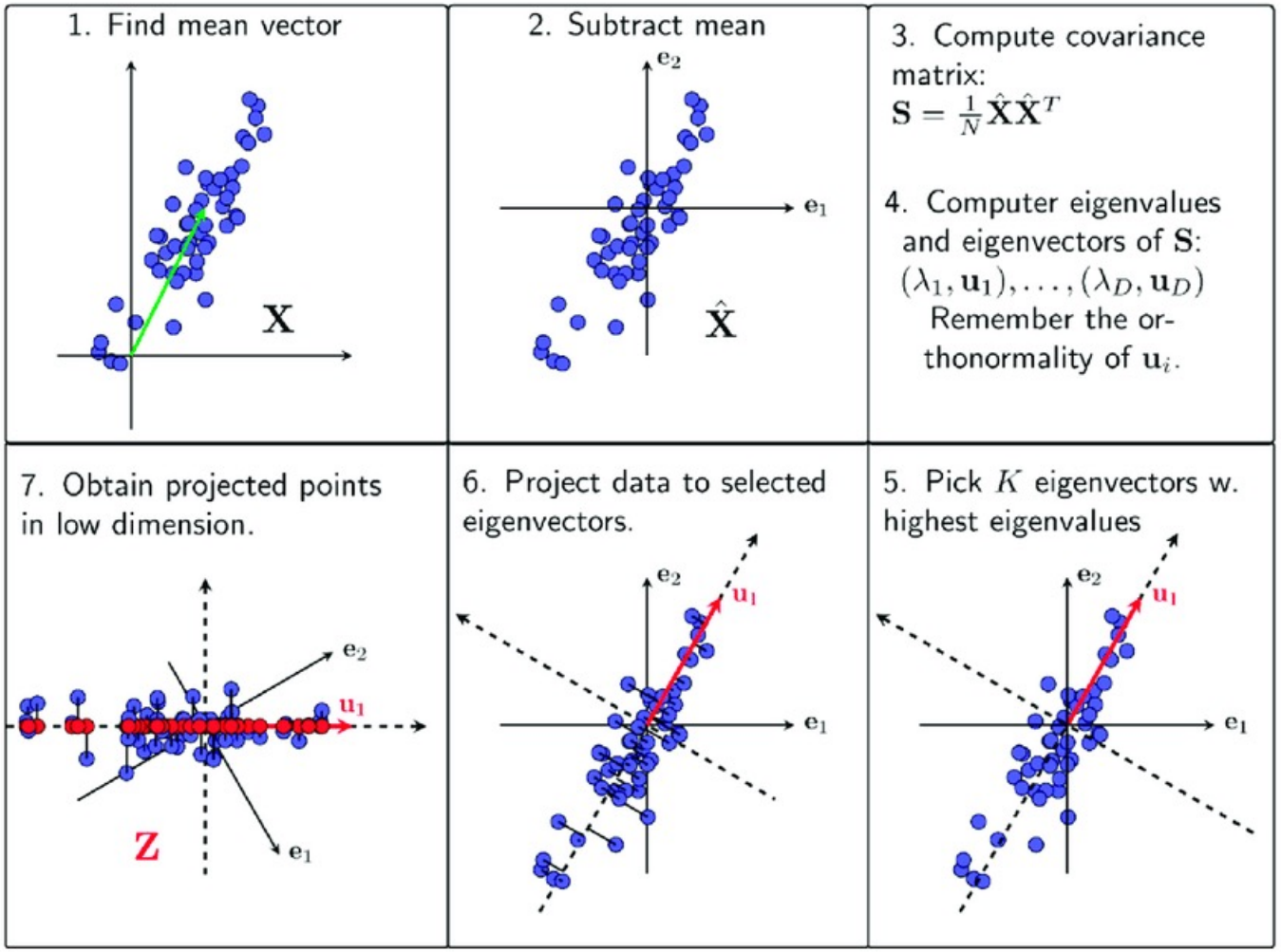
Representation Learning

Principal Component Analysis (PCA)

- Process to implement dimension reduction using PCA
- **S1:** Standardization. $\boldsymbol{x}_i = \boldsymbol{x}_i - \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_i$
- **S2:** Compute the covariance matrix $\frac{\mathbf{1}}{m}\boldsymbol{X}\boldsymbol{X}^T$
- **S3:** Eigenvalue decompose the matrix $\frac{\mathbf{1}}{m}\boldsymbol{X}\boldsymbol{X}^T$
- **S4:** Take the top maximum d' eigenvalues and corresponding eigenvectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}$
- **S5:** Get the transformation matrix $\boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}\}$, and output the reduced feature vectors $\boldsymbol{z}_i$ for original samples.

Representation Learning

- Principal Component Analysis (PCA)



Representation Learning

Principal Component Analysis (PCA)

- For PCA, only need to retain mean vector and $\boldsymbol{W}$ to project new sample on the low dimension space by the vector minus (standardization) and matrix-vector multiplication (projection).
- Discarding $(d - d')$ eigenvectors will lead to information loss. However, dimension reduction can increase the density of samples. Since the discarded eigenvalues and eigenvectors are always related to noise, which achieves de-noising for samples.

Representation Learning

Principal Component Analysis (PCA)

Example

- Problem: Given the original sample set $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4, \boldsymbol{x}_5) = \begin{pmatrix} -1 & -1 & 0 & 2 & 0 \\ -2 & 0 & 0 & 1 & 1 \end{pmatrix}$. Reduce sample from 2-D to 1-D using PCA.

- Process to implement dimension reduction using PCA
- **S1:** Standardization. $\boldsymbol{x}_i = \boldsymbol{x}_i - \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_i$
- **S2:** Compute the covariance matrix $\frac{\mathbf{1}}{m}\boldsymbol{X}\boldsymbol{X}^T$
- **S3:** Eigenvalue decompose the matrix $\frac{\mathbf{1}}{m}\boldsymbol{X}\boldsymbol{X}^T$
- **S4:** Take the top maximum d' eigenvalues and corresponding eigenvectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}$
- **S5:** Get the transformation matrix $\boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d'}\}$, and output the reduced feature vectors $\boldsymbol{z}_i$ for original samples.

Representation Learning

Answer:

- S1. Standardization. The mean vector $\frac{1}{m}\sum_{i=1}^{m}\boldsymbol{x}_i = 0$
- S2. Compute covariance matrix $\boldsymbol{A} = \frac{1}{5}\boldsymbol{X}\boldsymbol{X}^T = \frac{1}{5}\begin{pmatrix}6 & 4\\ 4 & 6\end{pmatrix}$
- S3. Eigenvalue decompose.

(1) Solve Eigenvalue by $|\lambda I - \boldsymbol{A}| = 0$

$$|\lambda I - \boldsymbol{A}| = \begin{vmatrix}\lambda - \frac{6}{5} & -\frac{4}{5}\\ -\frac{4}{5} & \lambda - \frac{6}{5}\end{vmatrix} = (\lambda - \frac{6}{5})^2 - \frac{16}{25} = (\lambda - 2)\left(\lambda - \frac{2}{5}\right) = 0$$

$$\lambda_1 = 2,\ \ \lambda_2 = \frac{2}{5}$$

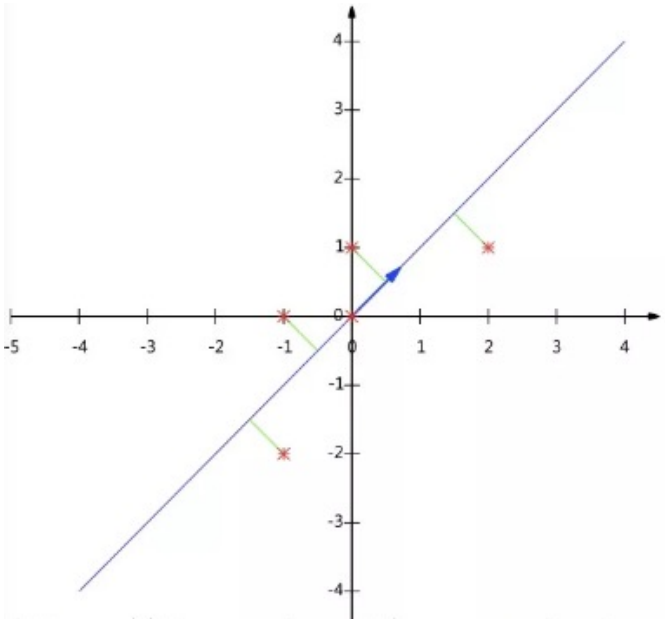
(2) Solve Eigenvector by $(\lambda I - A)\boldsymbol{w} = \boldsymbol{0}$

$\lambda_1 = 2 \rightarrow \boldsymbol{w}_1 = \binom{1}{1}, \quad \lambda_2 = \frac{2}{5} \rightarrow \boldsymbol{w}_2 = \binom{-1}{1}$

Representation Learning

- S4. Rank the eigenvalues, and select top $d' = 1$ eigenvalue λ_1 and its corresponding eigenvector $\boldsymbol{w}_1$. Standardization eigenvector to get $\boldsymbol{w}_1 = \frac{1}{\sqrt{2}}\binom{1}{1}$
- S5. Dimension reduction.

$$Z = W^T X = (-\frac{3}{\sqrt{2}} \quad -\frac{1}{\sqrt{2}} \quad 0 \quad \frac{3}{\sqrt{2}} \quad \frac{1}{\sqrt{2}})$$

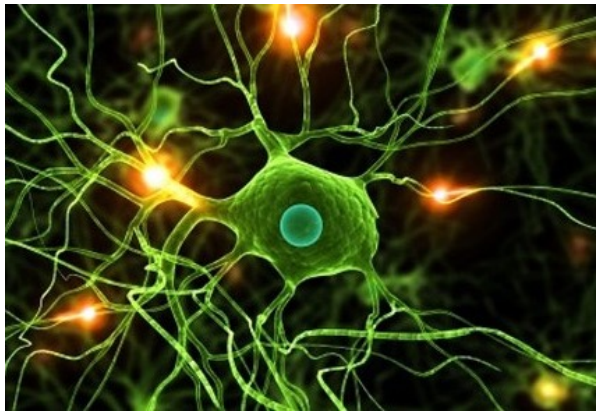


Unsupervised learning

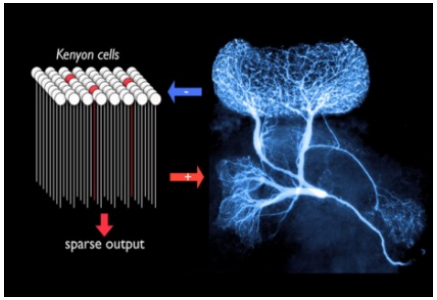
- Clustering
  - K-means method
  - Spectral clustering
- Representation learning
  - Linear coding
  - PCA
  - Others
  - Applications

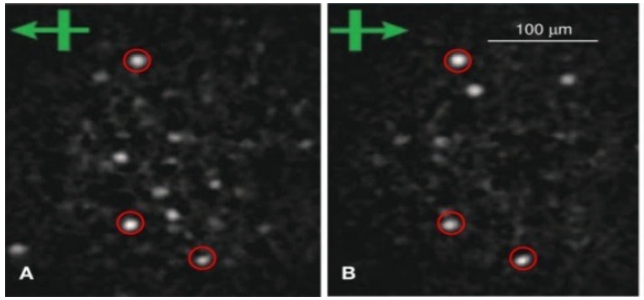
Representation Learning

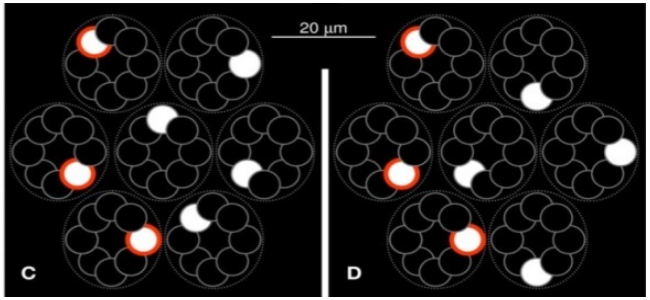
- Sparse coding



  - Information are coded sparsely in our brain.
  - Simulation of sparse coding mechanism in brain

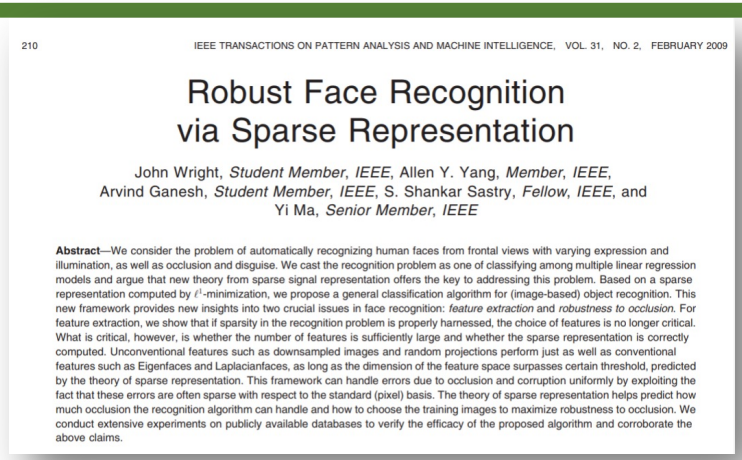






Representation Learning

- Sparse coding



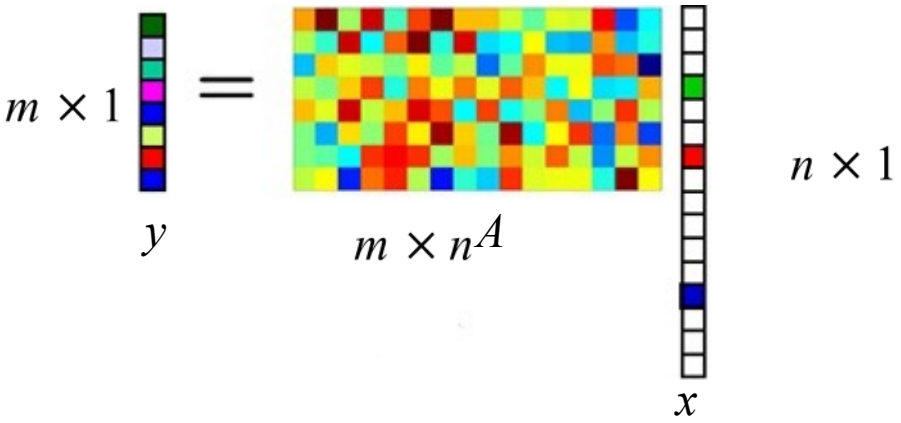
210 IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 31, NO. 2, FEBRUARY 2009

Robust Face Recognition via Sparse Representation

John Wright, *Student Member, IEEE*, Allen Y. Yang, *Member, IEEE*, Arvind Ganesh, *Student Member, IEEE*, S. Shankar Sastry, *Fellow, IEEE*, and Yi Ma, *Senior Member, IEEE*

Abstract—We consider the problem of automatically recognizing human faces from frontal views with varying expression and illumination, as well as occlusion and disguise. We cast the recognition problem as one of classifying among multiple linear regression models and argue that new theory from sparse signal representation offers the key to addressing this problem. Based on a sparse representation computed by ℓ^1-minimization, we propose a general classification algorithm for (image-based) object recognition. This new framework provides new insights into two crucial issues in face recognition: *feature extraction* and *robustness to occlusion*. For feature extraction, we show that if sparsity in the recognition problem is properly harnessed, the choice of features is no longer critical. What is critical, however, is whether the number of features is sufficiently large and whether the sparse representation is correctly computed. Unconventional features such as downsampled images and random projections perform just as well as conventional features such as Eigenfaces and Laplacianfaces, as long as the dimension of the feature space surpasses certain threshold, predicted by the theory of sparse representation. This framework can handle errors due to occlusion and corruption uniformly by exploiting the fact that these errors are often sparse with respect to the standard (pixel) basis. The theory of sparse representation helps predict how much occlusion the recognition algorithm can handle and how to choose the training images to maximize robustness to occlusion. We conduct extensive experiments on publicly available databases to verify the efficacy of the proposed algorithm and corroborate the above claims.

$$y = Ax \in R^m$$



Objective function

$$\min_x \|x\|_1, s.t. \|y - Ax\|_2^2 \leq \epsilon$$

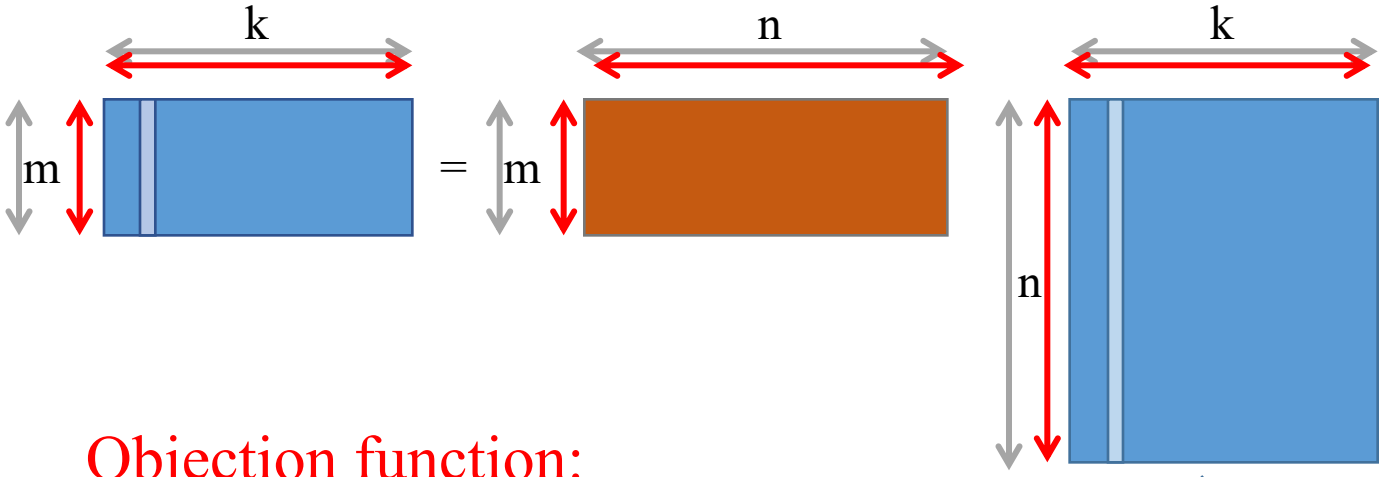
- Sparse coding is a challenging and promising theme in data analysis.
- Its main goal is to learn a sparse representation from an underdetermined dictionary.

Representation Learning

- Low rank representation

$$A \in \mathbb{R}^{m \times n}, \qquad m \ll n$$

$$\mathrm{Y} = AX, Y = [y_1, \ldots, y_k] \in \mathbb{R}^{m \times k}, X = [x_1, \ldots, x_k] \in \mathbb{R}^{n \times k}$$



Objection function:

$$\min_X rank(X), \qquad \text{s.t. } \mathrm{Y} = AX$$

Low rank

Unsupervised learning

- Clustering
  - K-means method
  - Spectral clustering
- Representation learning
  - Linear coding
  - PCA
  - Others
  - Applications

Representation Learning

- Applications

Facial Image Compression

Source image

JPEG image

JPEG2000 image

K-SVD image

Representation Learning

- Applications

Image Deblurring

Source image

Blurred image

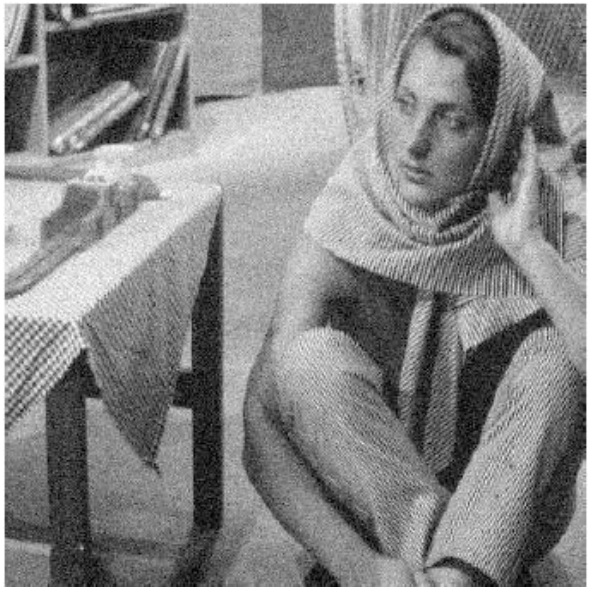
After deblurring

Representation Learning

- Applications

Image Denoising

Source image



Noisy image

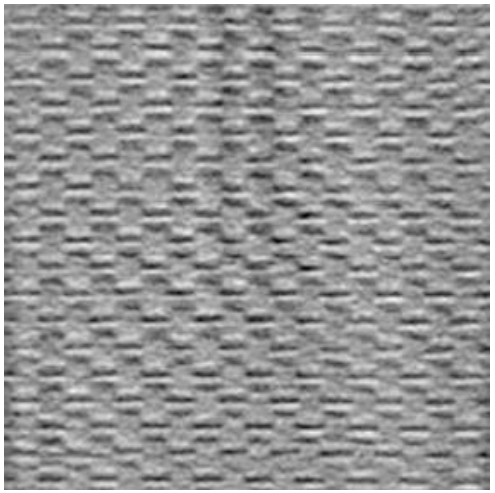
Denoising result

Representation Learning

- Applications

Morphological Component Analysis

Source image



Texture component

Cartoon component

Representation Learning

- Applications

Image Inpainting

Source image

Degraded image

Inpainting result