



Departamento de Eletrónica, Telecomunicações e
Informática

Machine Learning

LECTURE 7: UNSUPERVISED LEARNING (K-MEANS CLUSTERING AND PCA)

Petia Georgieva
(petia@ua.pt)

Outline

Unsupervised learning

1. K-means clustering

2. Data dimensionality reduction

- data compression / data visualization

3. Principal Component Analysis (PCA)

SUPERVISED vs. UNSUPERVISED LEARNING

Supervised Learning - (given DATA + LABELS):

ML method is trained with labeled data to predict the labels of new examples (learning by labeled examples)

| Matrix X | feature x_n | feature x_1 | | feature x_n | Vector y - output (label) |
|-----------|---------------|---------------|-------|---------------|-----------------------------|
| Example 1 | 1 | $x^{(1)}$ | | $x_n^{(1)}$ | $y^{(1)}$ |
| Example 2 | 1 | $x^{(2)}$ | | $x_n^{(2)}$ | $y^{(2)}$ |
| | 1 | | | | |
| | | | | $x_n^{(i)}$ | |
| | | | | | |
| | | | | | |
| Example m | 1 | $x^{(m)}$ | | $x_n^{(m)}$ | $y^{(m)}$ |

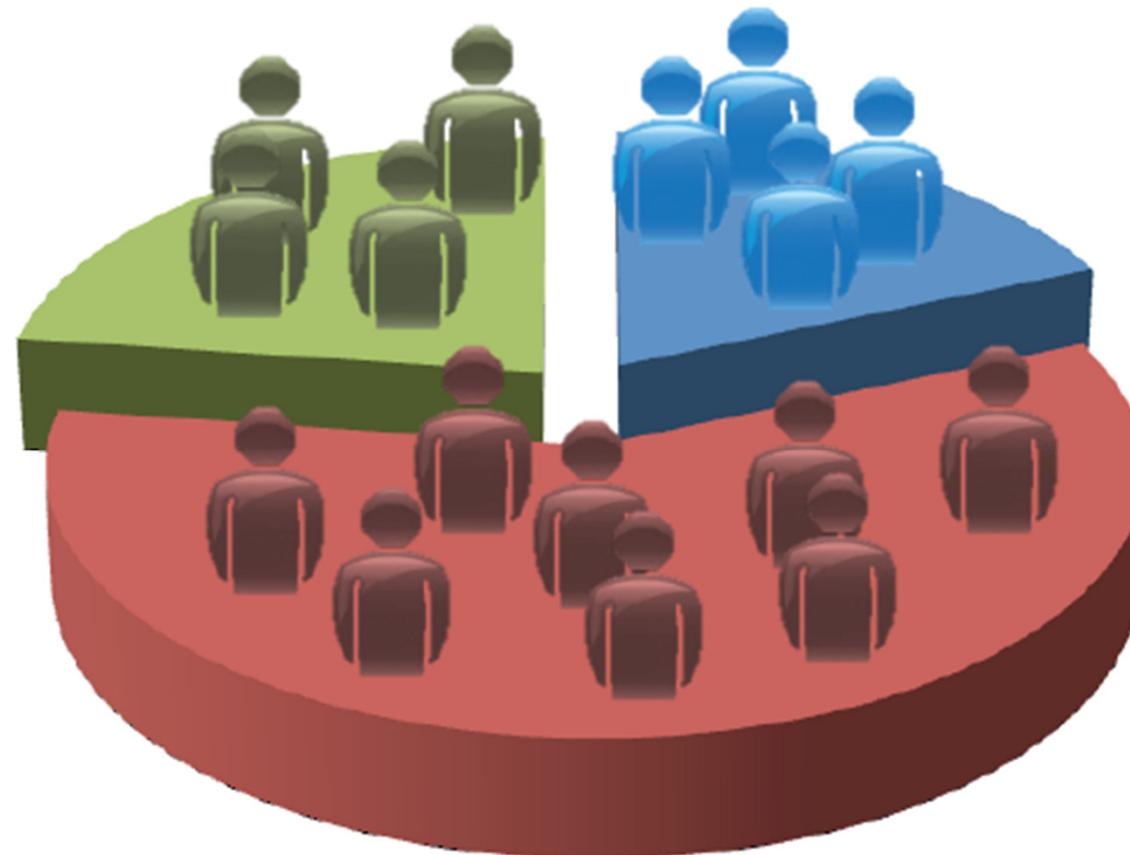
Unsupervised Learning - given UNLABELED DATA

ML method to discover the data internal (statistical) structure

| Matrix X | feature x_1 | feature x_2 | | feature x_n |
|-----------|---------------|---------------|-------|---------------|
| Example 1 | $x_1^{(1)}$ | $x_2^{(1)}$ | | $x_n^{(1)}$ |
| Example 2 | $x_1^{(2)}$ | $x_2^{(2)}$ | | $x_n^{(2)}$ |
| ... | | | | |
| Example i | $x_1^{(i)}$ | $x_2^{(i)}$ | | $x_n^{(i)}$ |
| ... | | | | |
| ... | | | | |
| Example m | $x_1^{(m)}$ | $x_2^{(m)}$ | | $x_n^{(m)}$ |

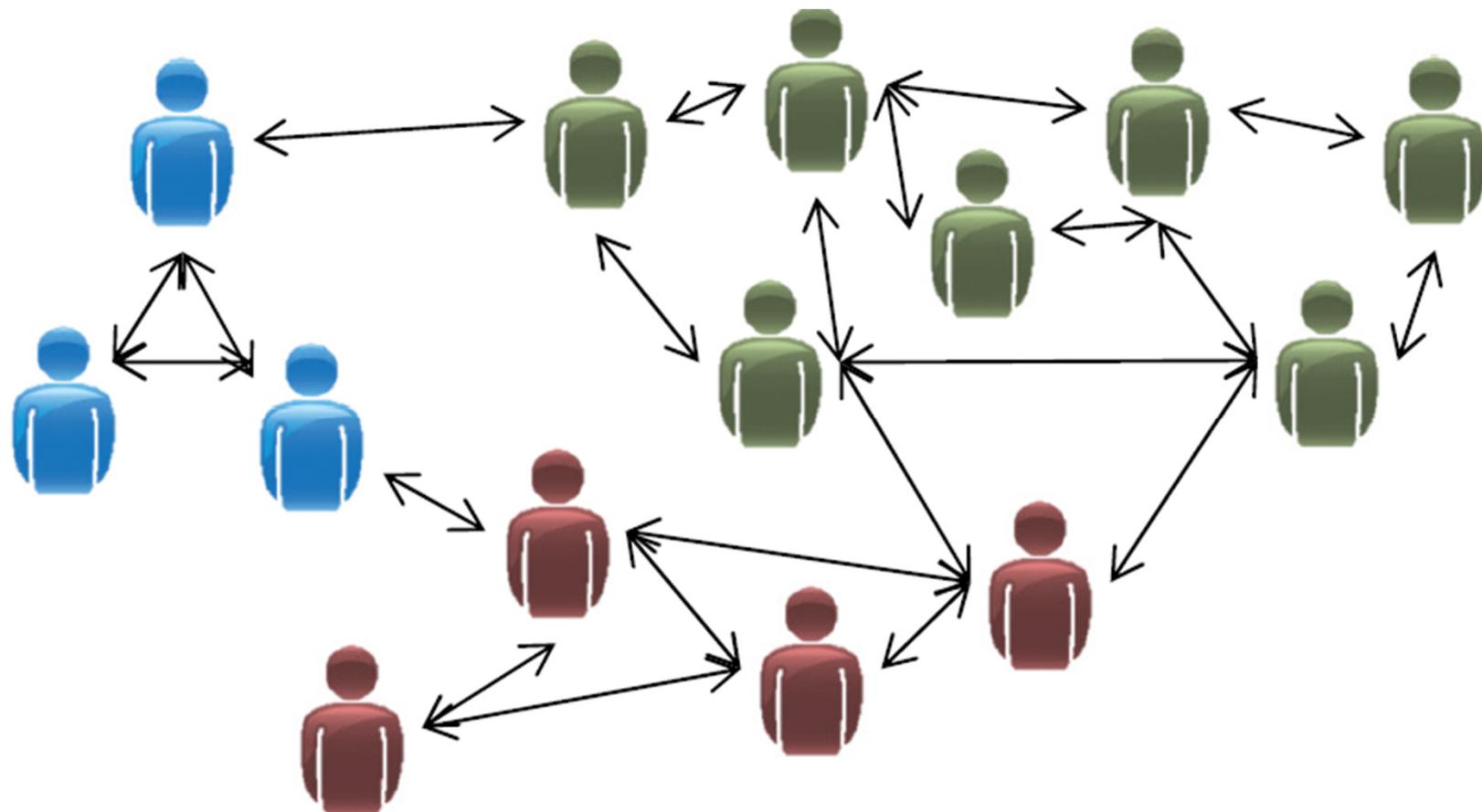
Unsupervised learning -

Market segmentation: data base of customers => division in target groups

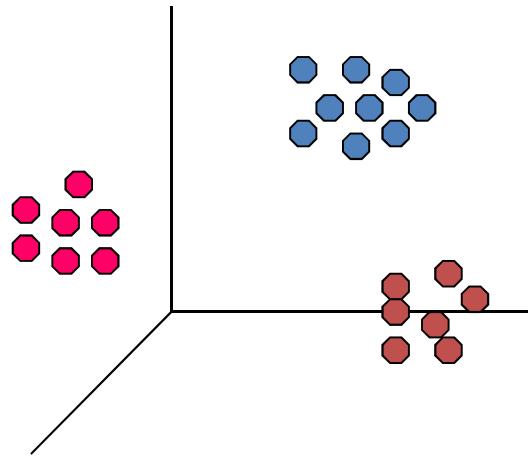


Unsupervised learning

Social network analysis: user grouping, group-specific advertising



Clustering intuition

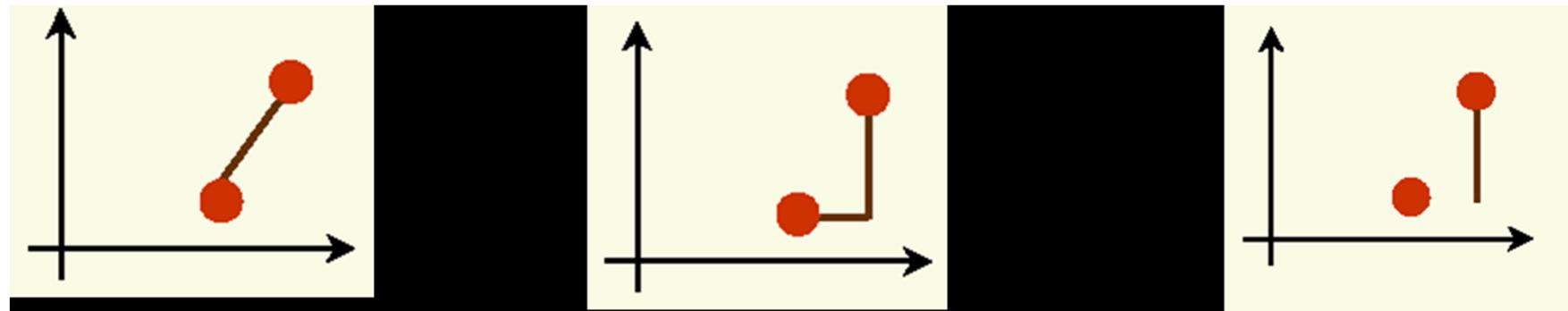


- Given a set of not labeled examples
- Find a relevant grouping of the examples into clusters such that:
 - Examples in the same cluster have **high similarity**
 - Examples from different clusters have **high dissimilarity**

Similarity measures –

Euclidian distance; Chebyshev distance; Manhattan distance

Distance (similarity) measures



**Euclidian Distance
(L2 norm)**

$$d(p,q)=\sqrt{(x_p-x_q)^2+(y_p-y_q)^2}$$

**Manhattan Distance
(L1 norm)**

$$d(p,q)=|x_p - x_q| + |y_p - y_q|$$

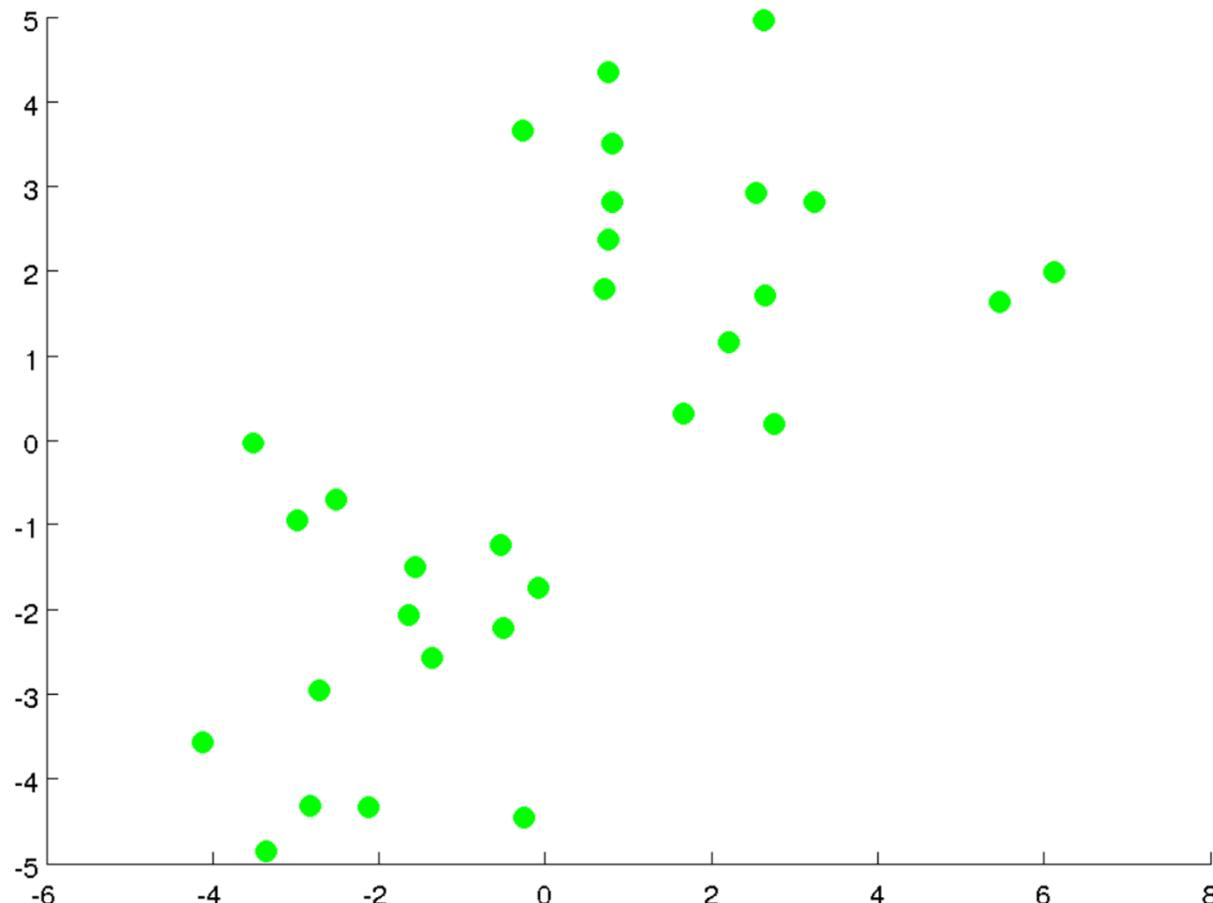
Chebyshev distance

$$d(p,q)=\max|(x_p - x_q),(y_p - y_q)|$$

K-means algorithm

Given:

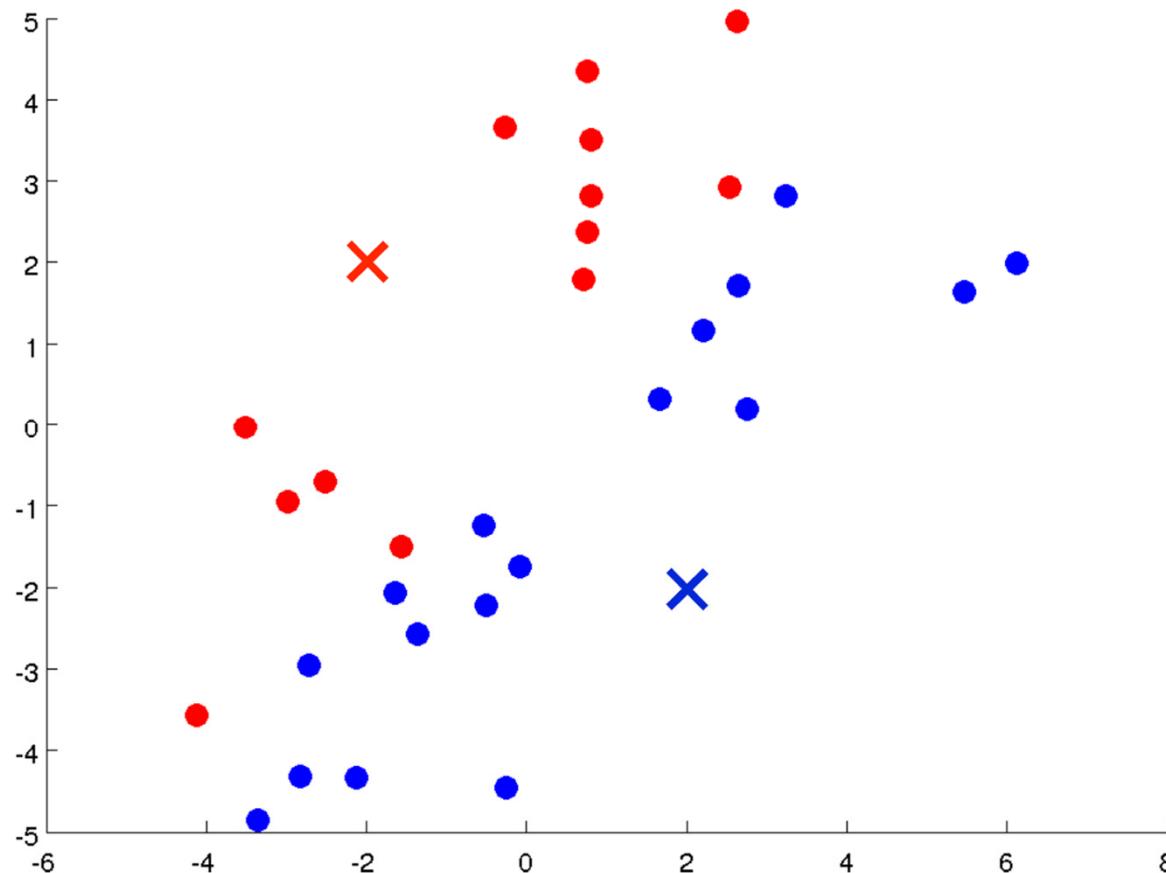
- K – the number of clusters
- Training set - no labels



K-means algorithm

Randomly initialize K cluster centroids (e.g. K=2)

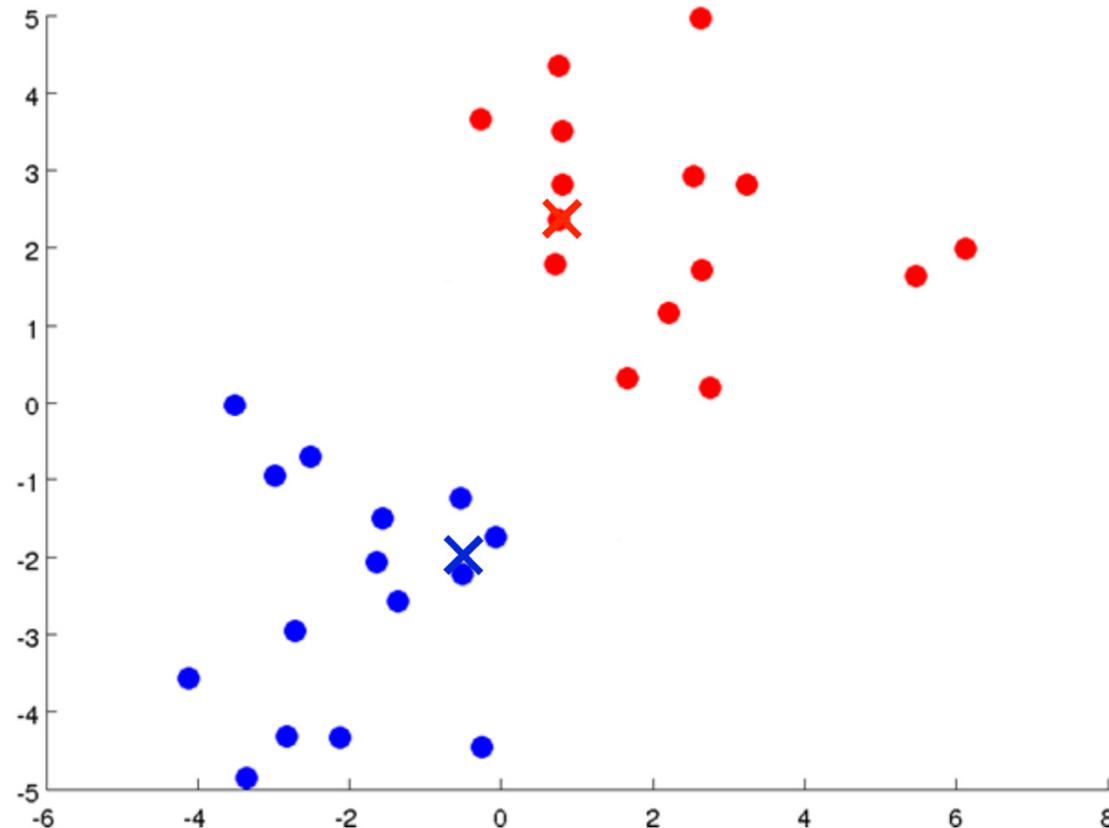
Assign data points to their closest centroid (Euclidian distance)



K-means algorithm

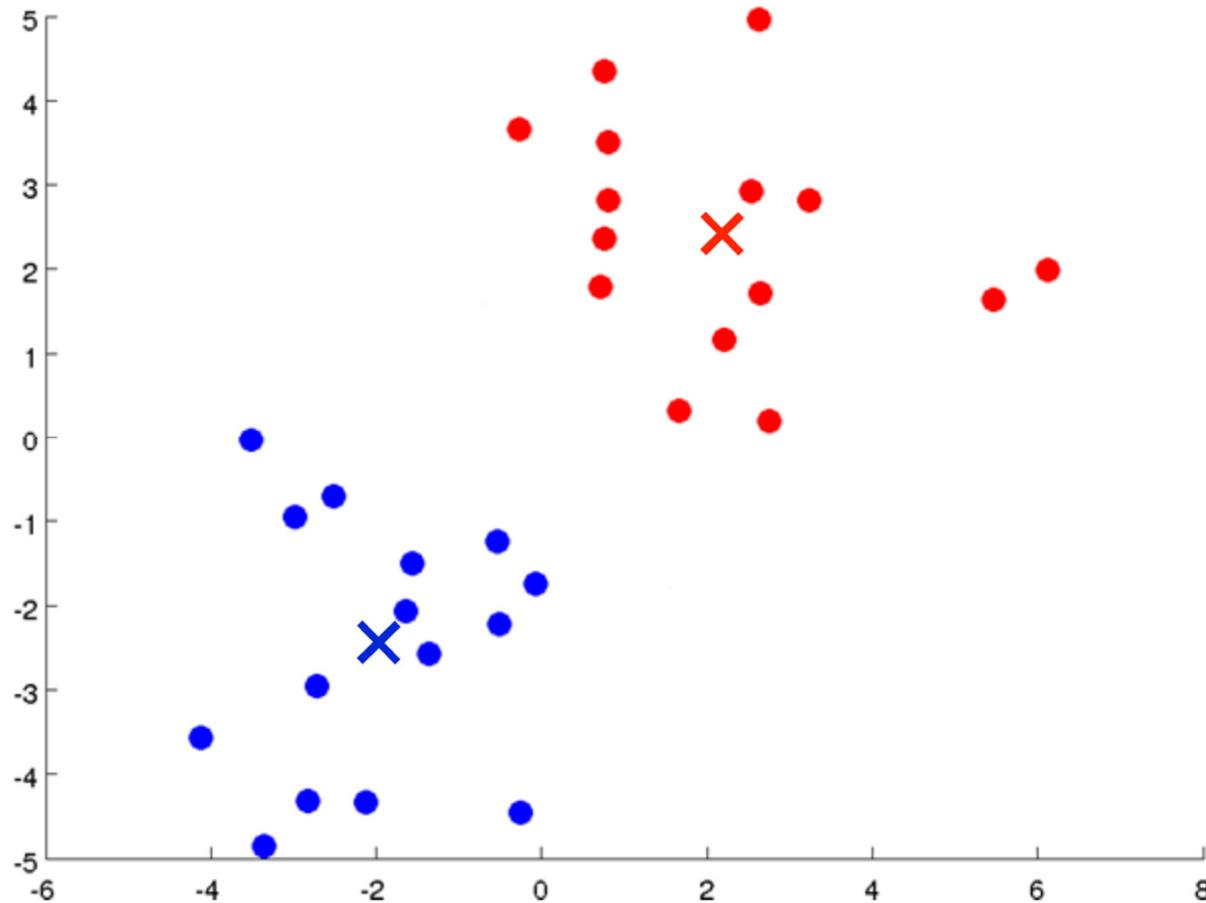
Compute new centroids = mean of the points assigned to that cluster.

Assign data points to the new closest centroid.



K-means algorithm

Repeat until convergence



K-means algorithms

Input:

- K (number of clusters)
- Training set (no labels)

Randomly initialize K cluster centroids $\mu_1, \mu_2, \dots, \mu_K \in \mathbb{R}^n$

Repeat {

*Cluster
assignment =>
step*

 for $i = 1$ to m
 $c^{(i)} :=$ index (from 1 to K) of cluster centroid
 closest to $x^{(i)}$

*Move centroid =>
step*

 for $k = 1$ to K
 $\mu_k :=$ average (mean) of points assigned to cluster k

}

K-means optimization objective (distortion = average distance)

$$J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K) = \frac{1}{m} \sum_{i=1}^m \|x^{(i)} - \mu_{c^{(i)}}\|^2$$
$$\min_{\substack{c^{(1)}, \dots, c^{(m)}, \\ \mu_1, \dots, \mu_K}} J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$$

Stop K-means learning (different criteria):

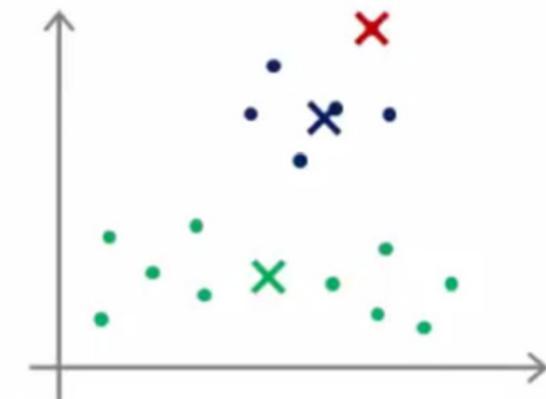
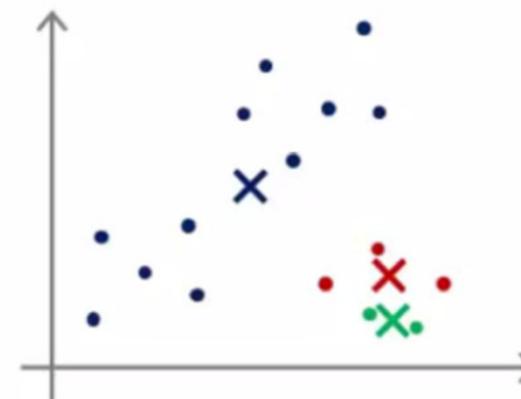
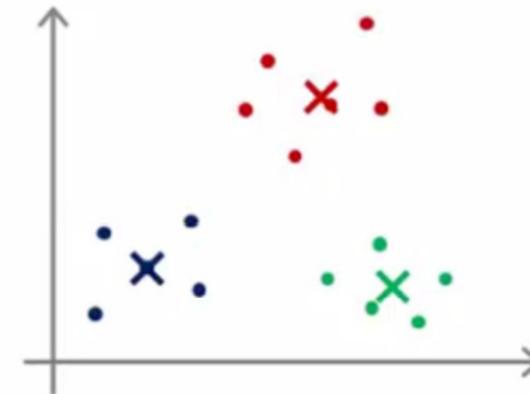
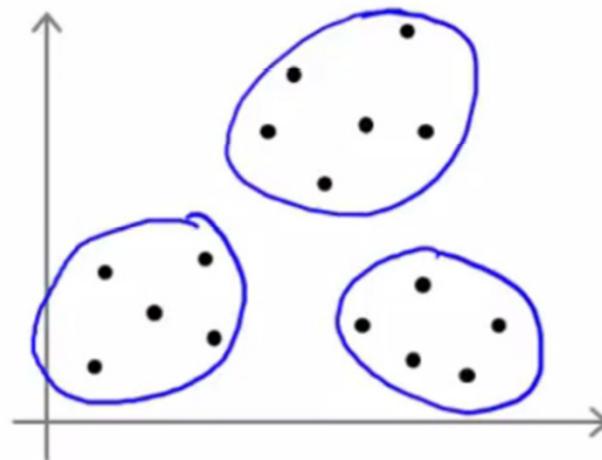
- Achieved Max number of iterations
- $J <$ some threshold
- No improvement of J between subsequent iterations

Single (Random) Initialization

Choose number of clusters K

Initialize K cluster centroids = randomly picked K training examples

Local optima



Repeat Random Initializations

For i = 1 to 100 {

 Randomly initialize K-means.

 Run K-means. Get $c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K$.

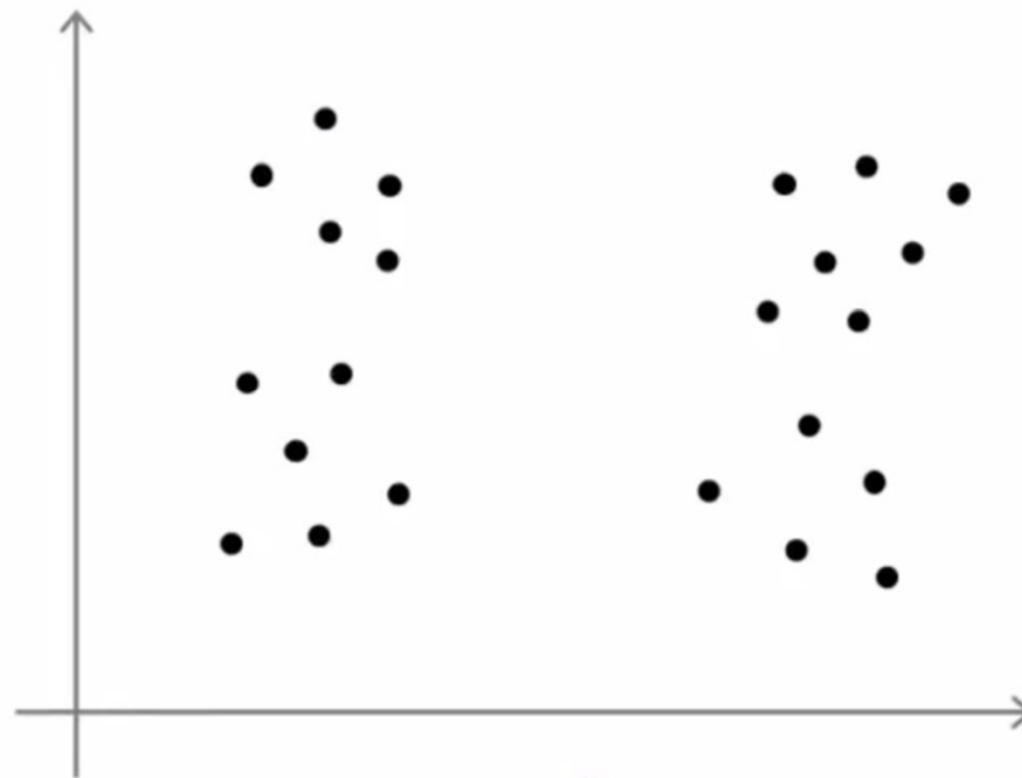
 Compute cost function (distortion)

$J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$

}

Pick clustering that gave lowest cost $J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$

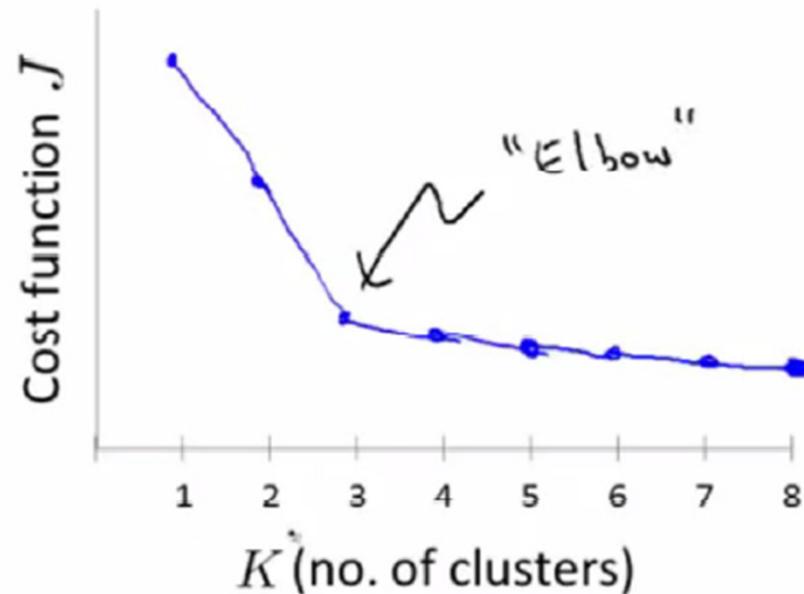
How to Choose # of clusters



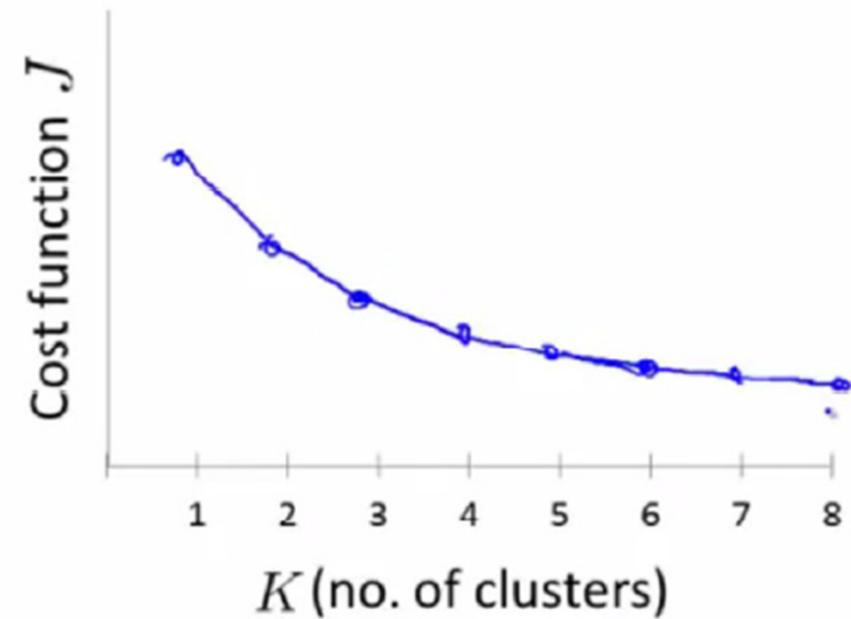
- Choose K by data visualisation (if possible)
- Ask domain experts (highly recommendable) , e.g. anomaly detection (experts should know how many types of anomalies are expected)
- Choose K automatically (e.g. Elbow method)

Choosing the number of clusters (Elbow method)

Appropriate for Elbow method

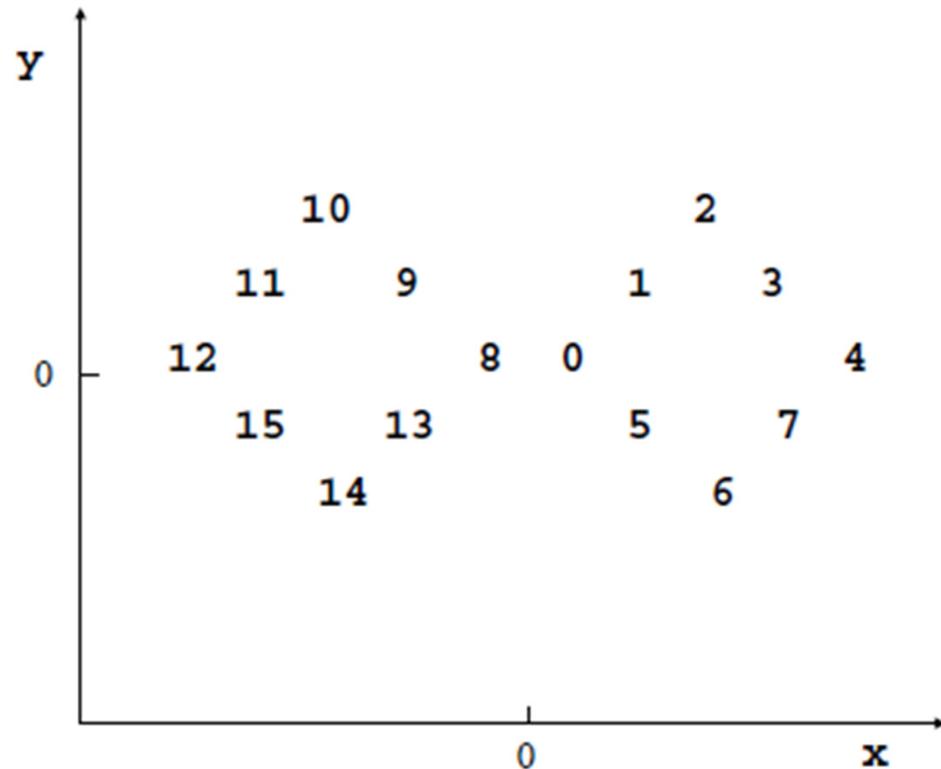


Not Appropriate for Elbow method



K-MEANS CLUSTERING – Example

| Id | x | y |
|-----|------|------|
| 0: | 1.0 | 0.0 |
| 1: | 3.0 | 2.0 |
| 2: | 5.0 | 4.0 |
| 3: | 7.0 | 2.0 |
| 4: | 9.0 | 0.0 |
| 5: | 3.0 | -2.0 |
| 6: | 5.0 | -4.0 |
| 7: | 7.0 | -2.0 |
| 8: | -1.0 | 0.0 |
| 9: | -3.0 | 2.0 |
| 10: | -5.0 | 4.0 |
| 11: | -7.0 | 2.0 |
| 12: | -9.0 | 0.0 |
| 13: | -3.0 | -2.0 |
| 14: | -5.0 | -4.0 |
| 15: | -7.0 | -2.0 |



- find the best 2 clusters

K-MEANS CLUSTERING – Example

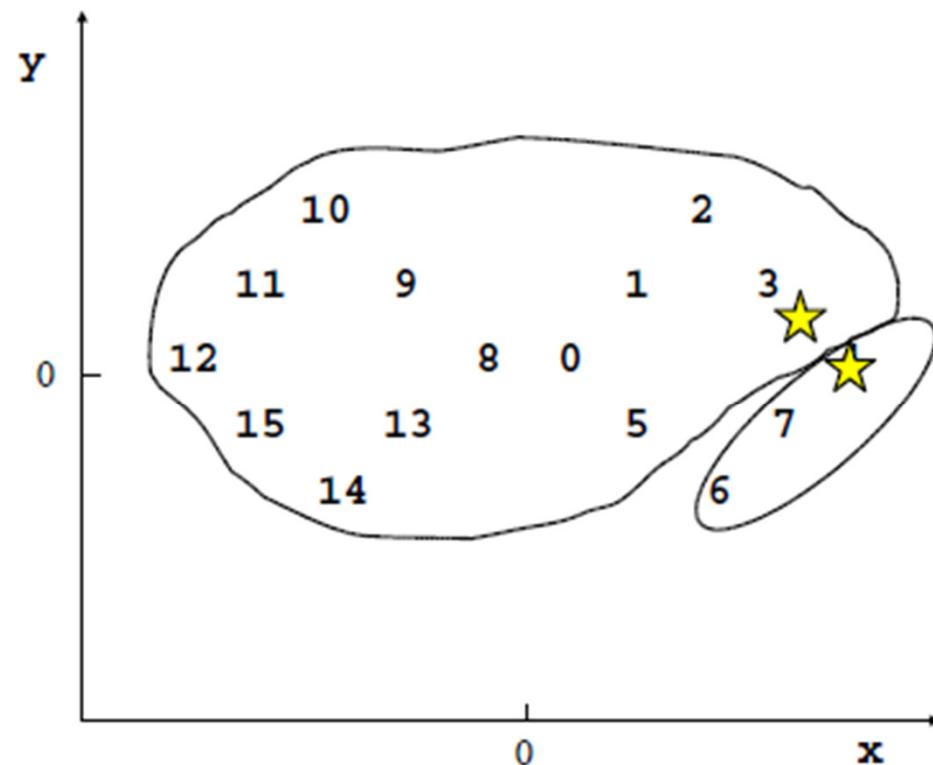
Seed: (9 0) (8 1)

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)

Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)

Average Distance: 4.35887

por causa do assignment inicial ,random



K-MEANS CLUSTERING – Example

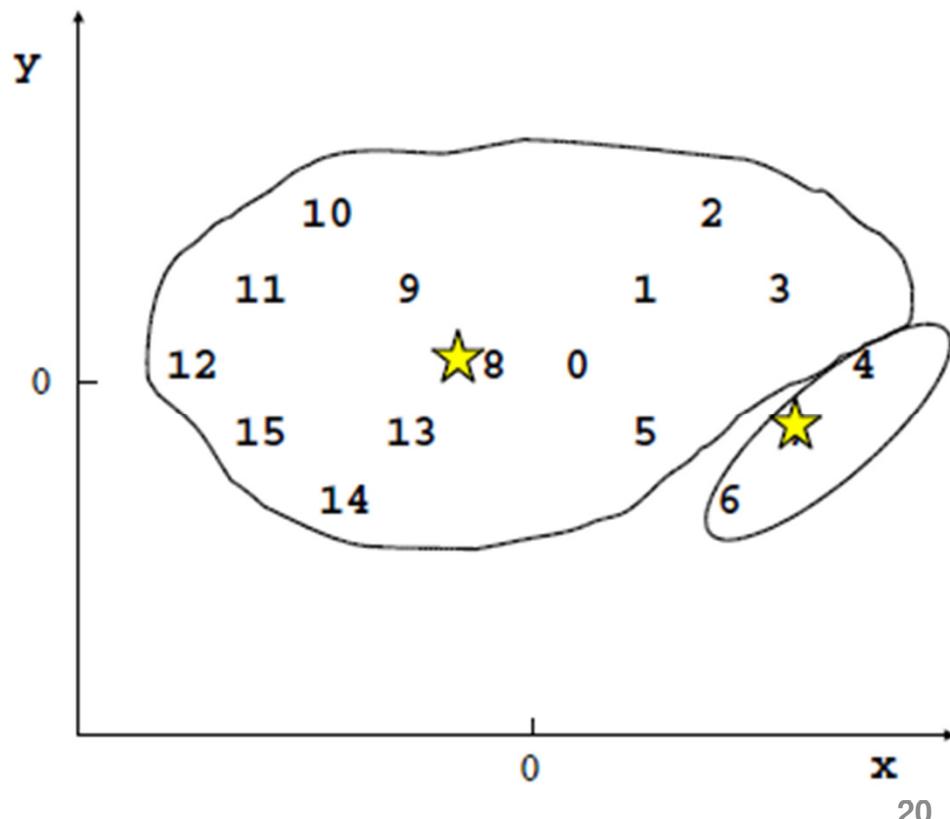
Seed: (9 0) (8 1)

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)

Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)

Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)



K-MEANS CLUSTERING – Example

Seed: (9 0) (8 1)

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)

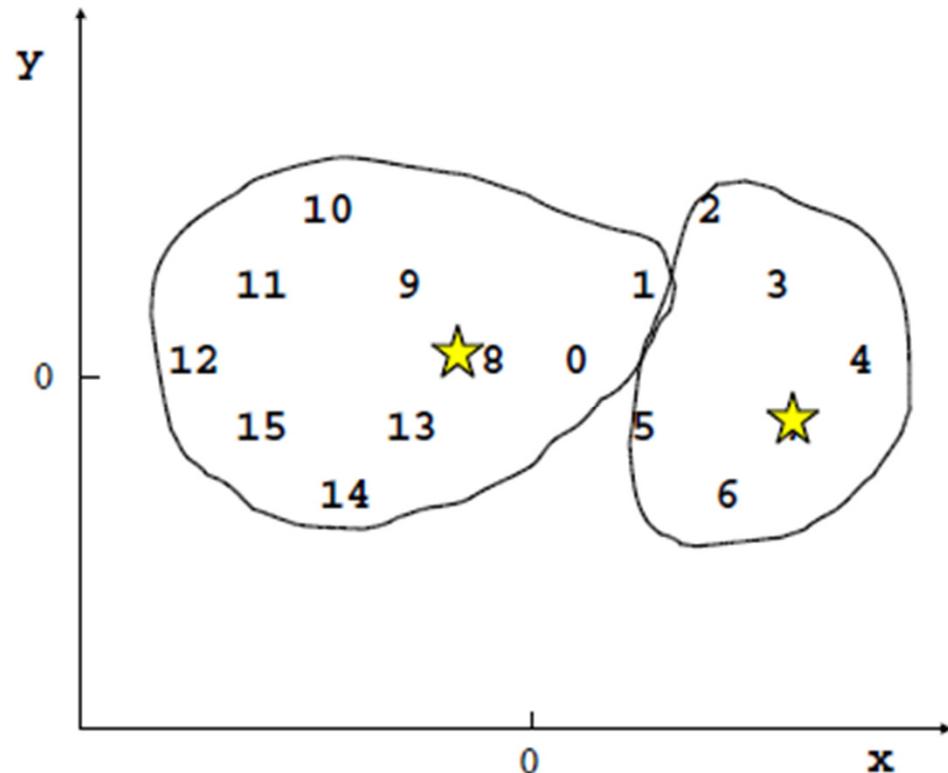
Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)

Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)

Cluster Centers: (6.0 -0.33334) (-3.6 0.2)

Average Distance: 3.6928



K-MEANS CLUSTERING – Example

Seed: (9 0) (8 1)

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)

Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)

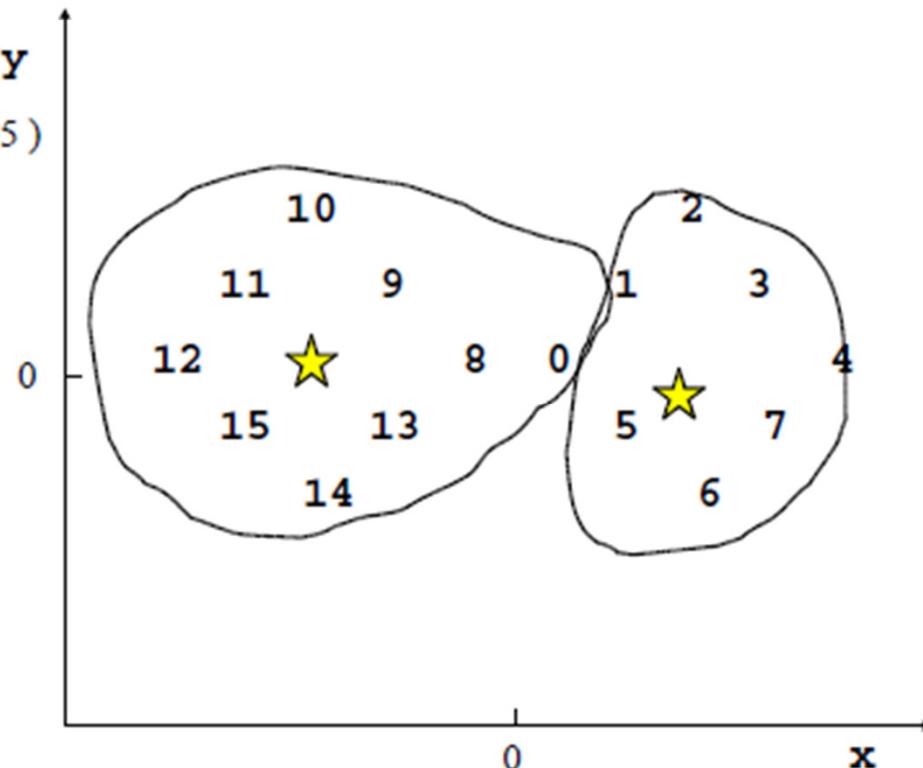
Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)

Cluster Centers: (6.0 -0.33334) (-3.6 0.2)

Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)



K-MEANS CLUSTERING – Example

Seed: (9 0) (8 1)

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)

Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)

Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)

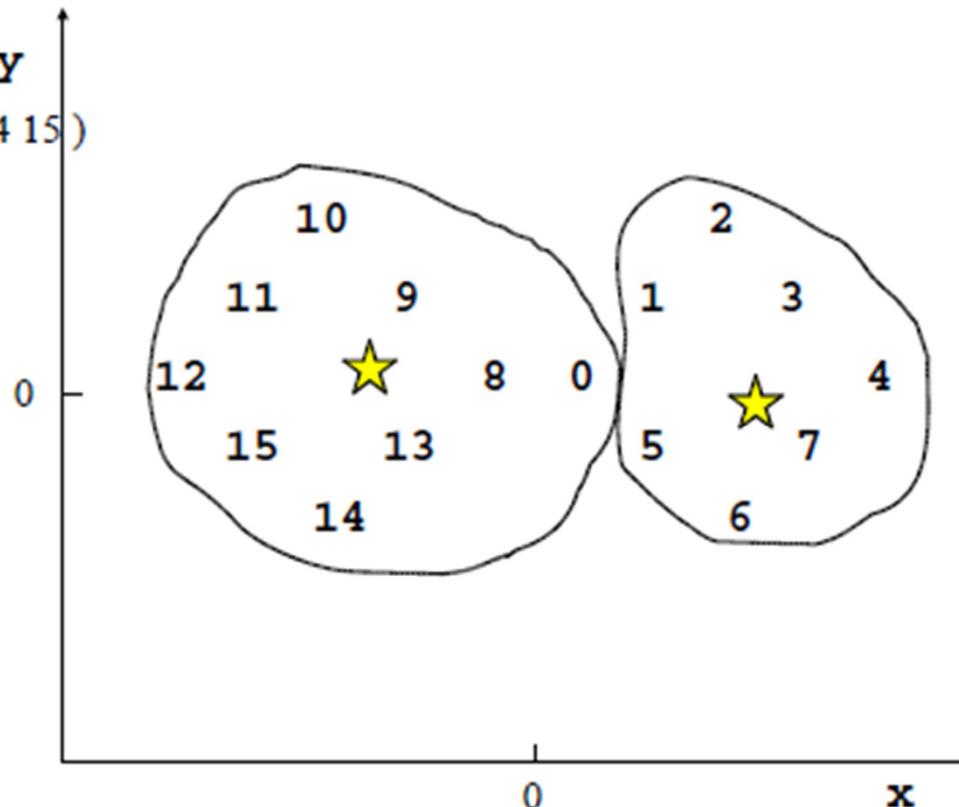
Cluster Centers: (6.0 -0.33334) (-3.6 0.2)

Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)

Cluster Centers: (5.57143 0.0) (-4.33334 0.0)

Average Distance: 3.49115



K-MEANS CLUSTERING – Example

Seed: (9 0) (8 1)

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)

Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)

Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)

Cluster Centers: (6.0 -0.33334) (-3.6 0.2)

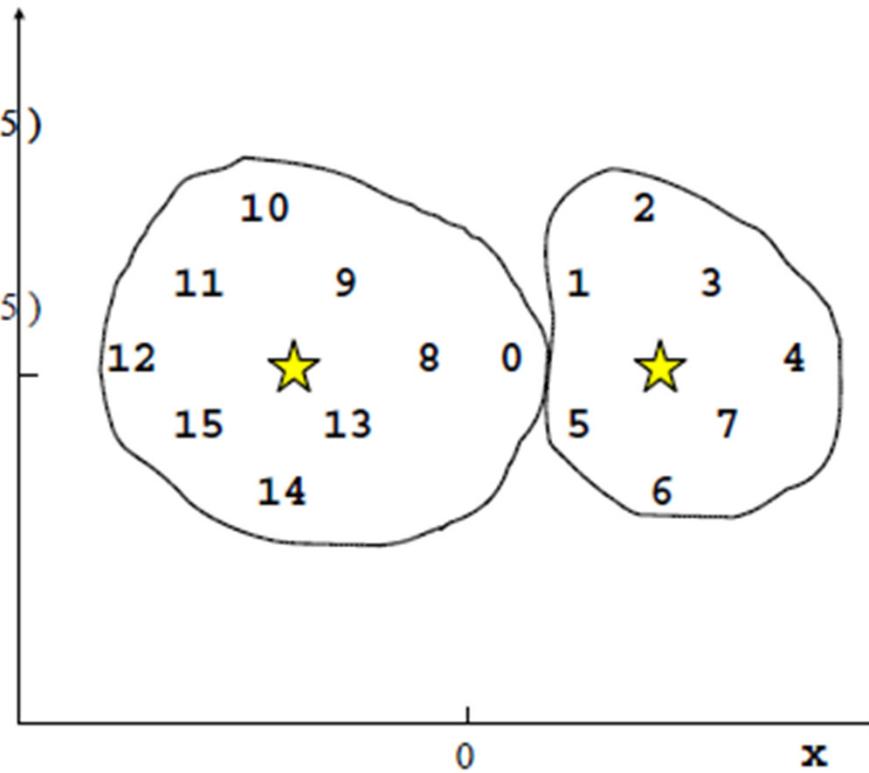
Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)

Cluster Centers: (5.57143 0.0) (-4.33334 0.0)

Average Distance: 3.49115

Clustering: (0 1 2 3 4 5 6 7) (8 9 10 11 12 13 14 15)



K-MEANS CLUSTERING – Example

Seed: (9 0) (8 1)

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)

Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)

Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)

Cluster Centers: (6.0 -0.33334) (-3.6 0.2)

Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)

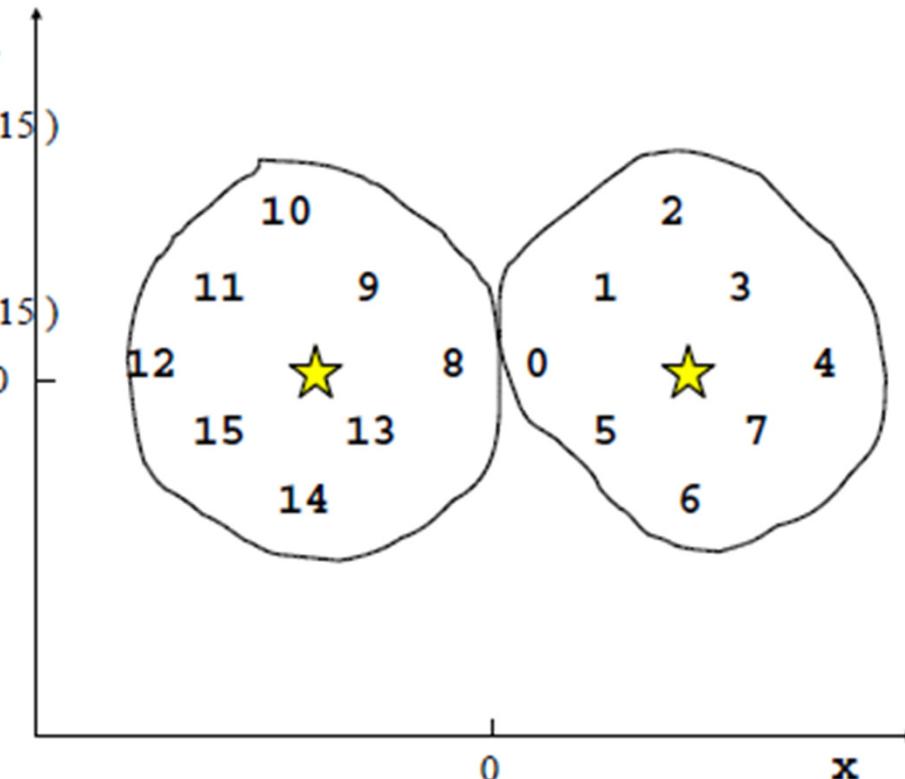
Cluster Centers: (5.57143 0.0) (-4.33334 0.0)

Average Distance: 3.49115

Clustering: (0 1 2 3 4 5 6 7) (8 9 10 11 12 13 14 15)

Cluster Centers: (5.0 0.0) (-5.0 0.0)

Average Distance: 3.41421



K-MEANS CLUSTERING – Example

Seed: (9 0) (8 1)

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)

Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)

Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)

Cluster Centers: (6.0 -0.33334) (-3.6 0.2)

Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)

Cluster Centers: (5.57143 0.0) (-4.33334 0.0)

Average Distance: 3.49115

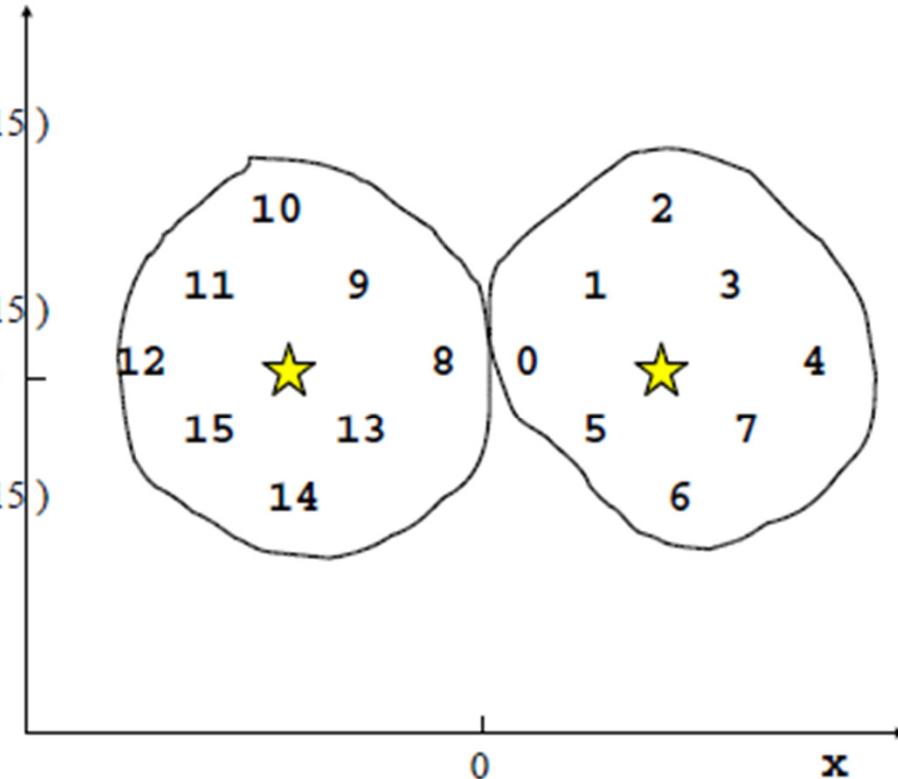
Clustering: (0 1 2 3 4 5 6 7) (8 9 10 11 12 13 14 15)

Cluster Centers: (5.0 0.0) (-5.0 0.0)

Average Distance: 3.41421

Clustering: (0 1 2 3 4 5 6 7) (8 9 10 11 12 13 14 15)

No improvement.



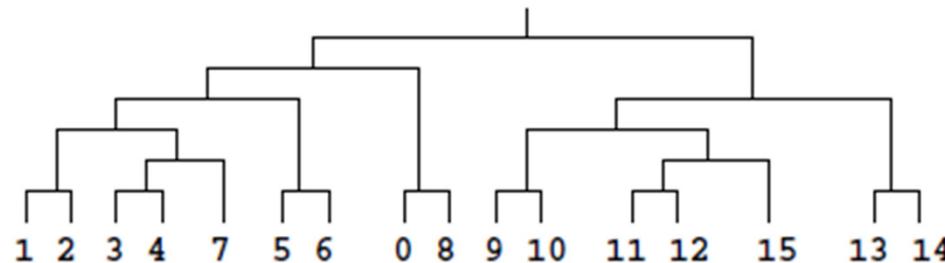
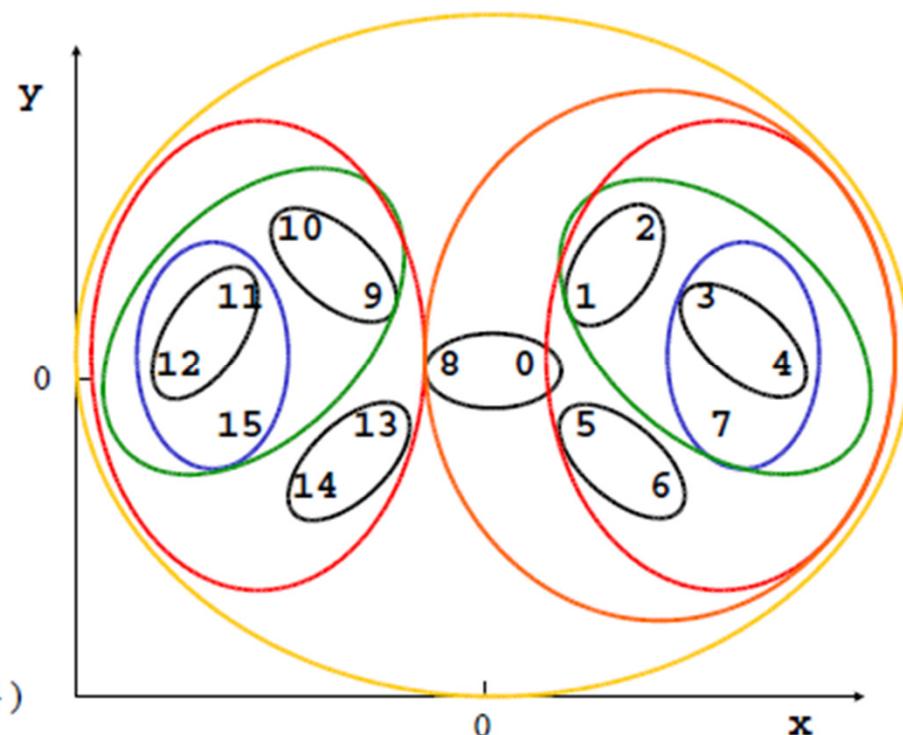
HIERARCHICAL CLUSTERING

- Two approaches
 - **Top-down:**
 - start with 1 cluster (all examples belong to one cluster)
 - successively split into new sub clusters (based on some threshold max distance between the centroid and the examples)
 - **Bottom-up:**
 - start with as many clusters as examples
 - successively merge existing clusters
 - (based on some similarity measure between clusters)

SIMILARITY BETWEEN CLUSTERS

Bottom-up clustering (average-link):

```
min distance = 2.00000  ( 8 )( 0 )
min distance = 2.82843  ( 2 )( 1 )
min distance = 2.82843  ( 4 )( 3 )
min distance = 2.82843  ( 6 )( 5 )
min distance = 2.82843  ( 10 )( 9 )
min distance = 2.82843  ( 12 )( 11 )
min distance = 2.82843  ( 14 )( 13 )
min distance = 3.16228  ( 7 )( 3 4 )
min distance = 3.16228  ( 15 )( 11 12 )
min distance = 4.73756  ( 3 4 7 )( 1 2 )
min distance = 4.73756  ( 11 12 15 )( 9 10 )
min distance = 4.74131  ( 1 2 3 4 7 )( 5 6 )
min distance = 4.74131  ( 9 10 11 12 15 )( 13 14 )
min distance = 5.57143  ( 0 8 )( 5 6 1 2 3 4 7 )
min distance = 9.90476  ( 13 14 9 10 11 12 15 )( 5 6 1 2 3 4 7 0 8 )
```



K-MEANS -summary

- The most popular clustering method.
- Need to know K.
- May converge to a Local Minimum .
- High number of computations.

K-means for dimensionality reduction

dimensionality reduction is useful for:

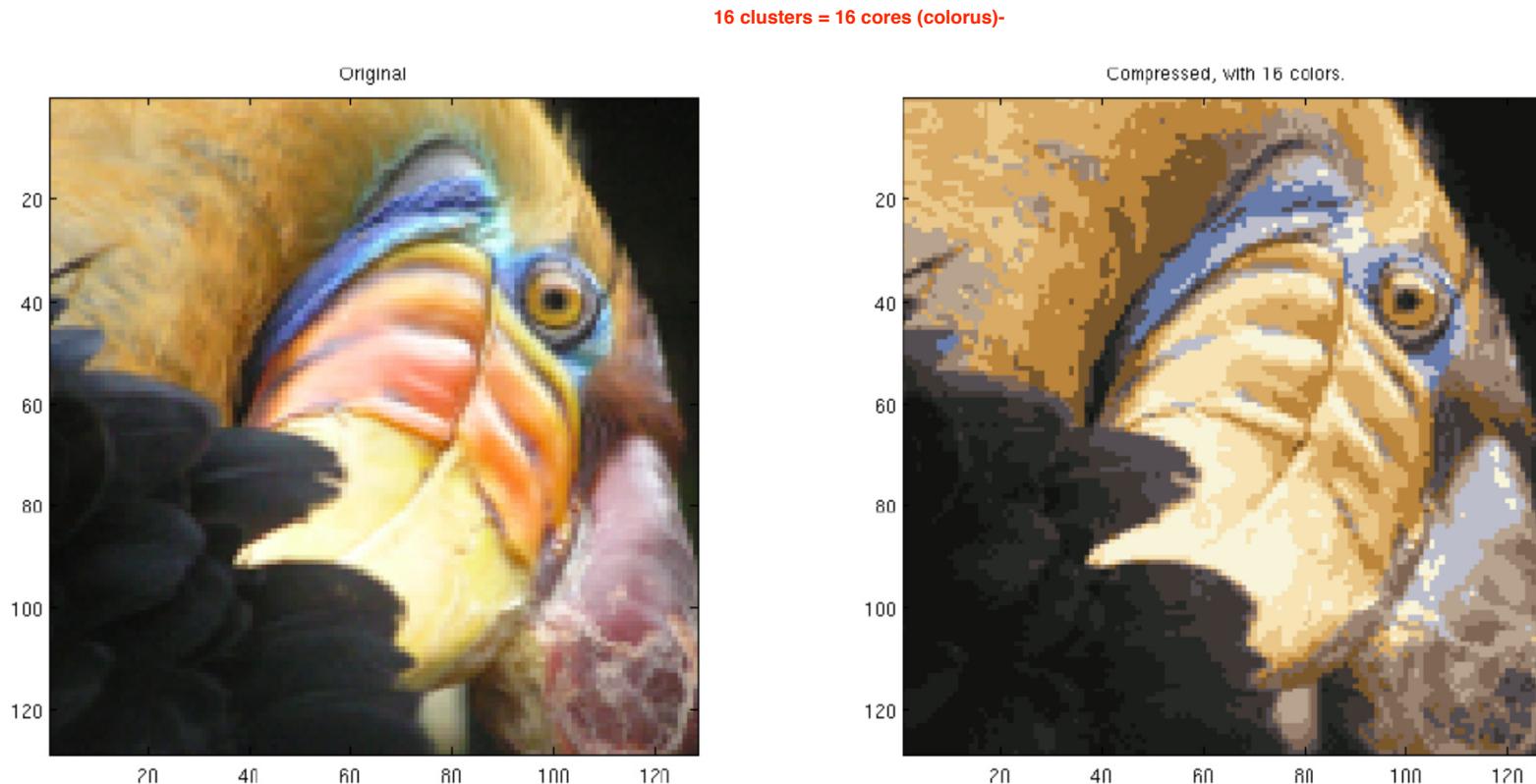
Data compression (from 10000-1000 D to) 100D

Reduce memory/ disk needed to store data
Speed up learning algorithm

Data visualization (from 100-50D to 2-3D)

Image compression with K-means

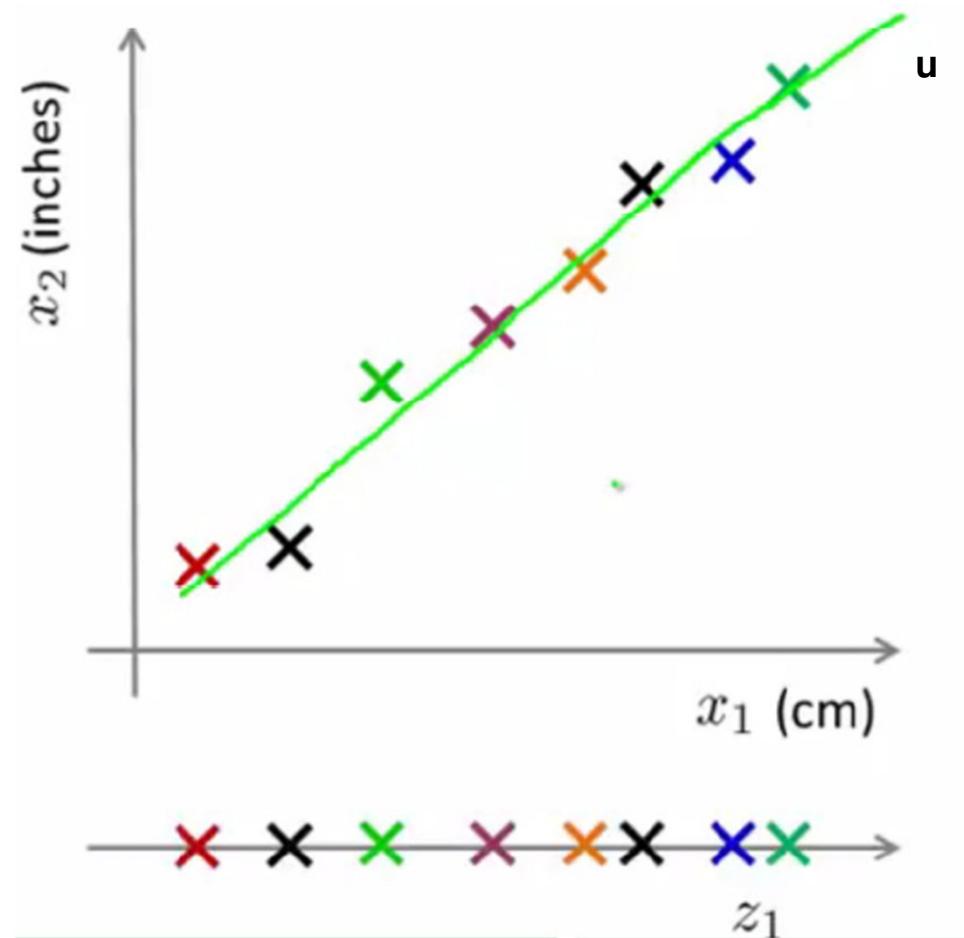
RGB image: 3*8 bits/pixel Compressed image: 16 colors(clusters) => 4 bits



DATA COMPRESSION

Example: reduce data from 2D to 1D

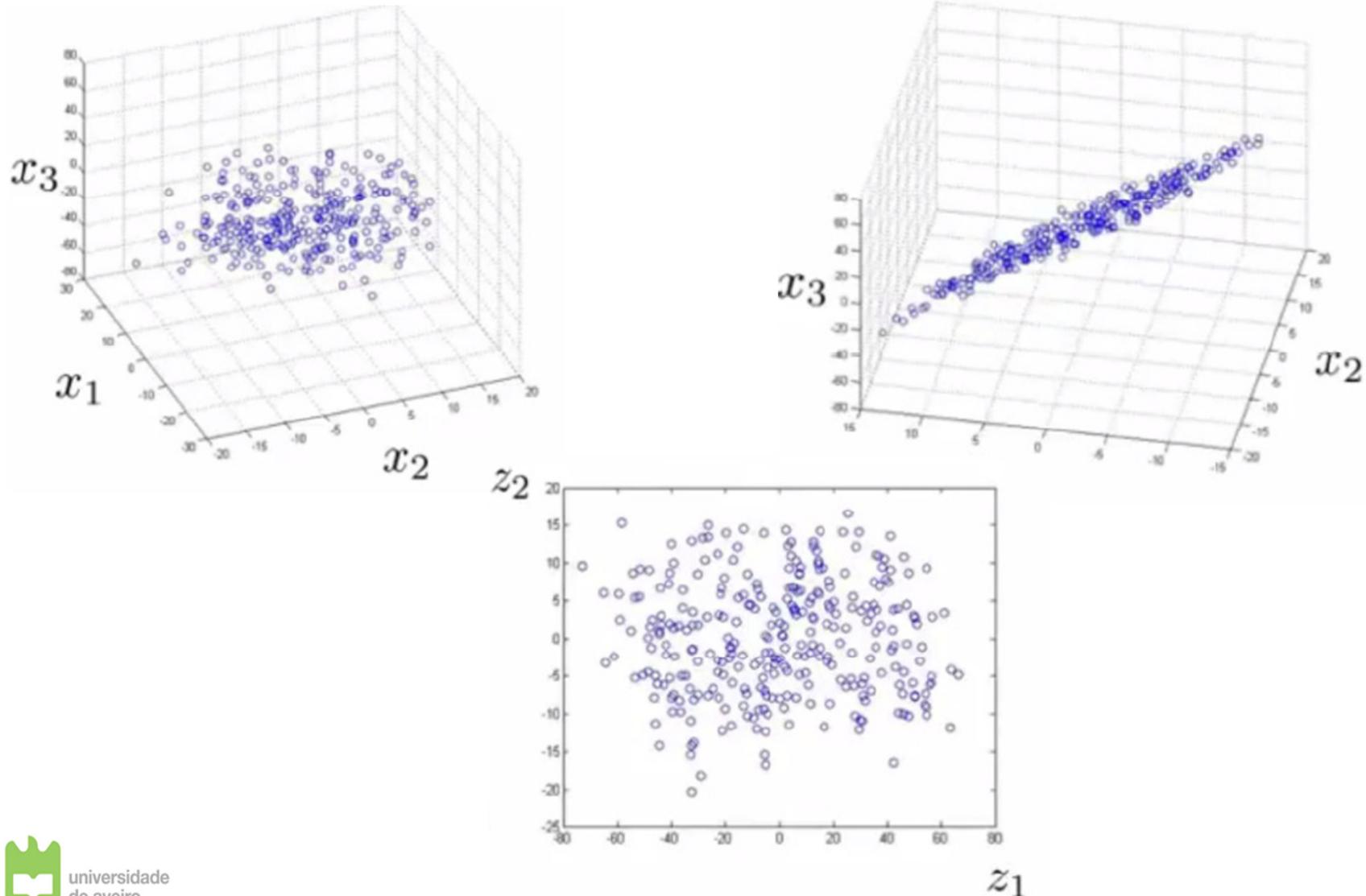
If 2D data is located along a line, the second dimension is redundant



DATA COMPRESSION

Example: reduce data from 3D to 2D

If 3D data is located along a plane, the 3rd dimension is redundant



DATA VISUALIZATION

Example: Reduce data from high dimension to 2D or 3D

| Country | GDP (trillions of US\$) | Per capita GDP (thousands of intl. \$) | Human Develop- ment Index | Life expectancy | Poverty Index (Gini as percentage) | Mean household income (thousands of US\$) | ... |
|-----------|-------------------------------|---|---------------------------------|--------------------|---|---|-----|
| Canada | 1.577 | 39.17 | 0.908 | 80.7 | 32.6 | 67.293 | ... |
| China | 5.878 | 7.54 | 0.687 | 73 | 46.9 | 10.22 | ... |
| India | 1.632 | 3.41 | 0.547 | 64.7 | 36.8 | 0.735 | ... |
| Russia | 1.48 | 19.84 | 0.755 | 65.5 | 39.9 | 0.72 | ... |
| Singapore | 0.223 | 56.69 | 0.866 | 80 | 42.5 | 67.1 | ... |
| USA | 14.527 | 46.86 | 0.91 | 78.3 | 40.8 | 84.3 | ... |
| ... | ... | ... | ... | ... | ... | ... | ... |

DATA VISUALIZATION

Reduce data from high dimension to 2D or 3D

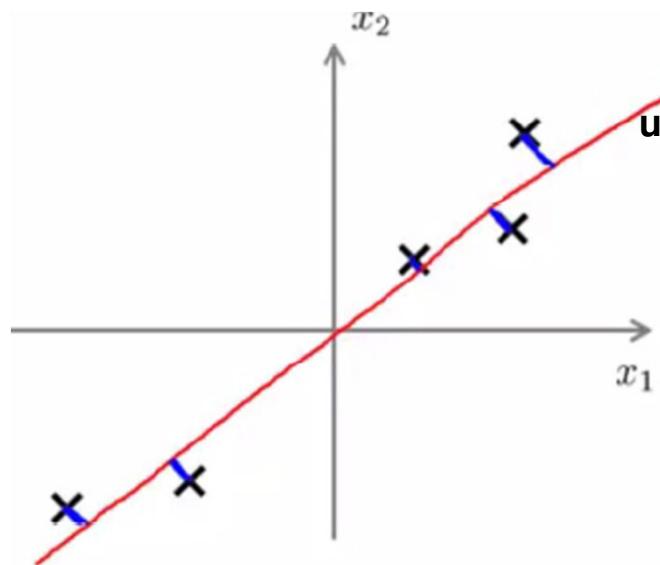
for example: z_1 (country size/GDP) and z_2 (GDP/per person)

| Country | z_1 | z_2 |
|-----------|-------|-------|
| Canada | 1.6 | 1.2 |
| China | 1.7 | 0.3 |
| India | 1.6 | 0.2 |
| Russia | 1.4 | 0.5 |
| Singapore | 0.5 | 1.7 |
| USA | 2 | 1.5 |
| ... | ... | ... |

PRINCIPAL COMPONENT ANALYSIS (PCA)

Reduce from 2D to 1D:

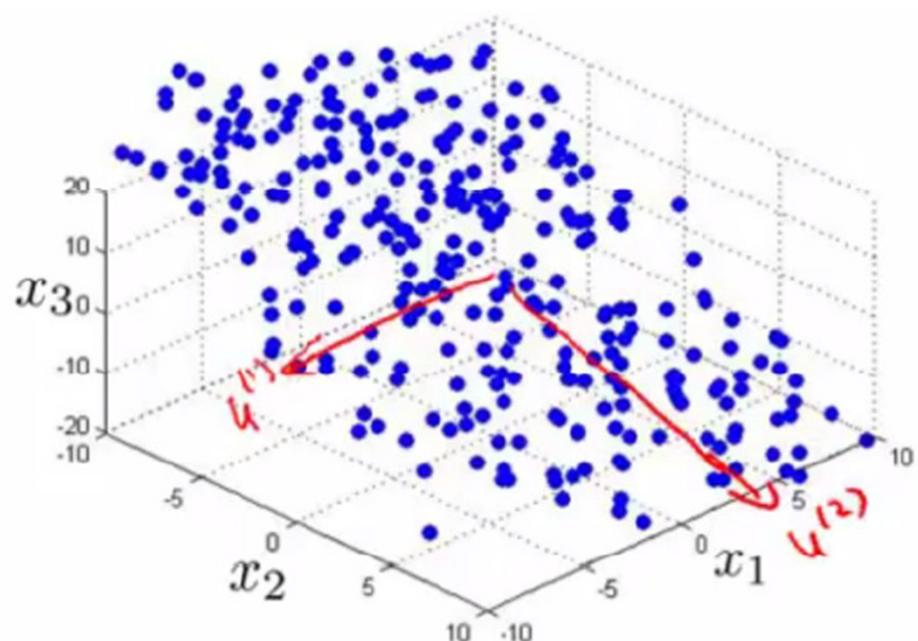
find the best direction (vector u) onto which to project data such that to minimize the projection error



Reduce from 3D to 2D:

find the orientation of the best plane (vectors u_1, u_2) onto which to project data such that to minimize the projection error

perdemos informação, mas ganhamos um melhor understanding a cerca da data...

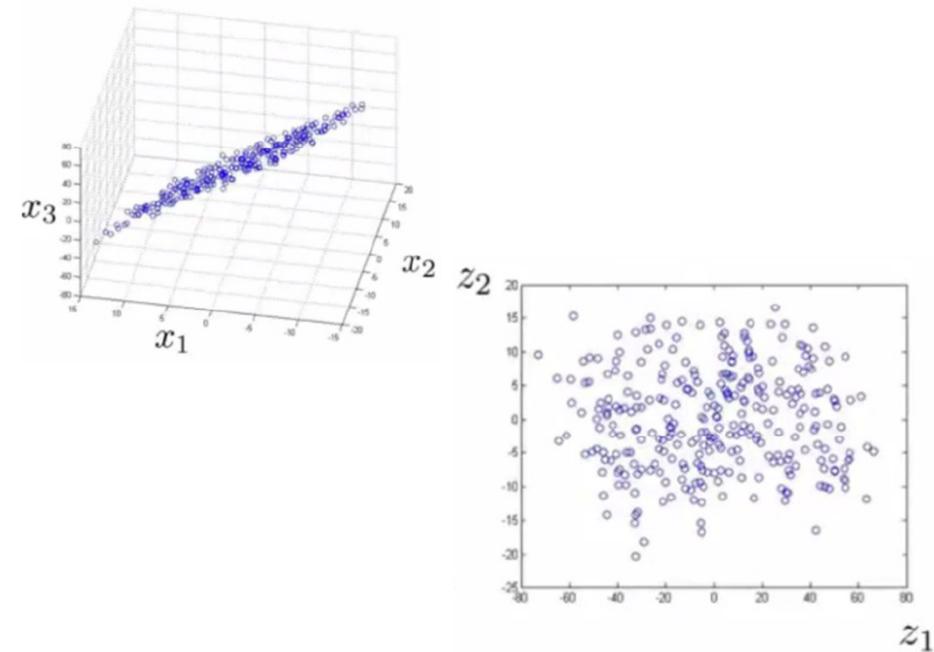
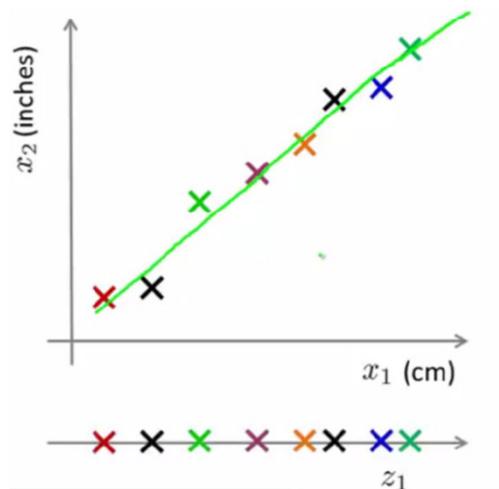


PRINCIPAL COMPONENT ANALYSIS (PCA)

Reduce from n-dimension to k-dimension ($k < n$):

Two tasks:

- Compute the coordinate system of the best k-dimensional surface (represented by vectors u_1, \dots, u_k) onto which to project data such that to minimize the projection error.
- Compute the values of the projected data in the lower dimensional space (z values)



PCA – DATA PREPROCESSING (step 1)

Training set: $x^{(1)}, x^{(2)}, \dots, x^{(m)}$

Preprocessing (feature scaling/mean normalization):

$$\mu_j = \frac{1}{m} \sum_{i=1}^m x_j^{(i)}$$

Replace each $x_j^{(i)}$ with $x_j - \mu_j$.

Thus, all features have zero mean !

If the features have significantly different range of values, normalize them., e.g. in the interval [0,1] or [-1,1] or mean=0 & std=1

| matrix \mathbf{X} ($m \times n$) | feature x_1 | feature x_2 | | feature x_n |
|---|---------------|---------------|-------|---------------|
| Example 1 | $x_1^{(1)}$ | $x_2^{(1)}$ | | $x_n^{(1)}$ |
| Example 2 | $x_1^{(2)}$ | $x_2^{(2)}$ | | $x_n^{(2)}$ |
| ... | | | | |
| Example i | $x_1^{(i)}$ | $x_2^{(i)}$ | | $x_n^{(i)}$ |
| ... | | | | |
| ... | | | | |
| Example m | $x_1^{(m)}$ | $x_2^{(m)}$ | | $x_n^{(m)}$ |

PCA – Singular Value Decomposition (step 2)

- Compute Covariance matrix of the mean normalized data matrix X (dimension $m \times n$ - m examples, n features):

$$\mathbf{Cov} = \mathbf{X}^T * \mathbf{X}$$

- Compute Singular Value Decomposition(SVD) of Covariance matrix:

$$\mathbf{Cov} = \mathbf{U} * \mathbf{S} * \mathbf{V}$$

U ($n \times n$) - matrix of eigenvectors:

$$U = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \dots & u^{(n)} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times n}$$

S ($n \times n$)– diagonal matrix of singular values in decreasing order:

$$S_{n \times n} = \begin{bmatrix} S_{11} & 0 & \dots & 0 \\ 0 & S_{22} & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ 0 & \dots & 0 & S_{nn} \end{bmatrix}$$

PCA

The projection vectors are the first k columns of U ($k < n$):

$$U = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \dots & u^{(n)} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times n} \quad Ureduce_{nxk} = U(:,1:k)$$

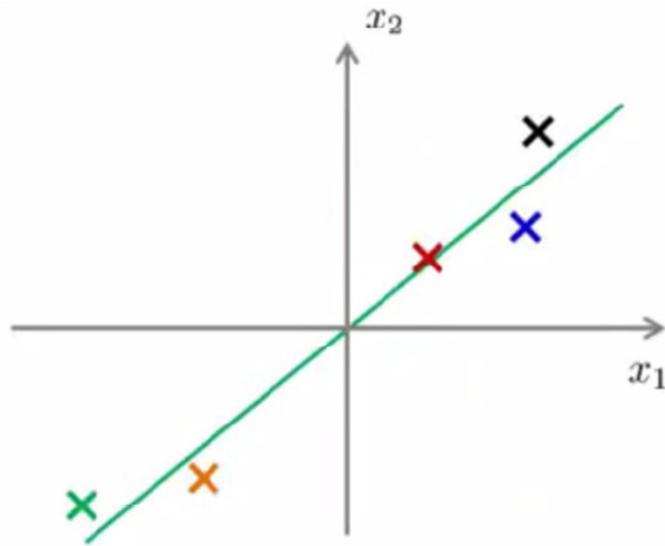
Step 3: Compute the new (projected) data matrix Z (m examples, k features):

$$Z_{mxk} = X_{mxn} * Ureduce_{nxk}$$

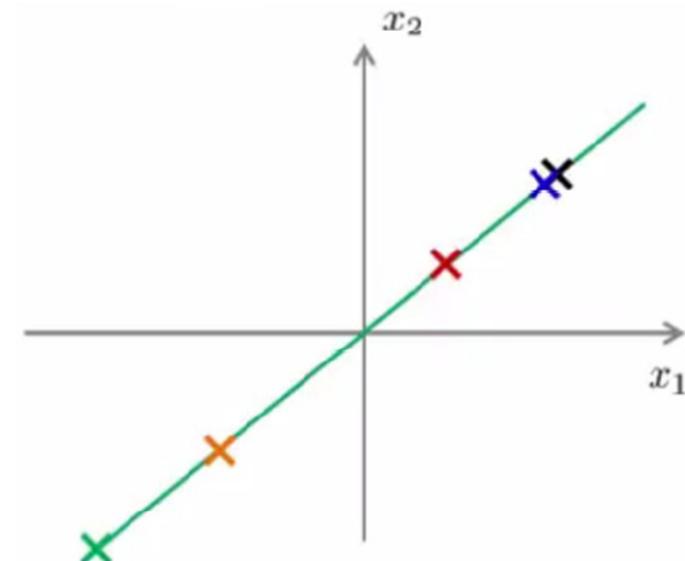
Step 4: Reconstruct data matrix X from the projected Z matrix :

$$X_{approx(mx)} = Z_{mxk} * Ureduce_{kxn}^T$$

Reconstruction from compressed representation



$$z = U_{reduce}^T x$$



$$x_{approx} = U_{reduce} z$$

Choosing k (number of principal components)

Average squared approximation error:

$$\frac{1}{m} \sum_{i=1}^m \|x^{(i)} - x_{approx}^{(i)}\|^2$$

Total data variation:

$$\frac{1}{m} \sum_{i=1}^m \|x^{(i)}\|^2$$

Typically, choose k to be smallest value so that

$$\frac{\frac{1}{m} \sum_{i=1}^m \|x^{(i)} - x_{approx}^{(i)}\|^2}{\frac{1}{m} \sum_{i=1}^m \|x^{(i)}\|^2} \leq 0.01 \quad (1\%)$$

“99% of variance is retained”

(typically the desired retained variance is between 90-99%)

Choosing k (number of principal components)

Compute SVD once: $[U, S, V] = \text{svd}(\text{Cov})$

Choose $k < n$ such that you get the desired retained variance
(e.g. 99 %):

$$\frac{\sum_{i=1}^k S_{ii}}{\sum_{i=1}^n S_{ii}} \geq 0.99$$

$$S_{nxn} = \begin{bmatrix} S_{11} & 0 & \cdots & 0 \\ 0 & S_{22} & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ 0 & \cdots & 0 & S_{nn} \end{bmatrix}$$