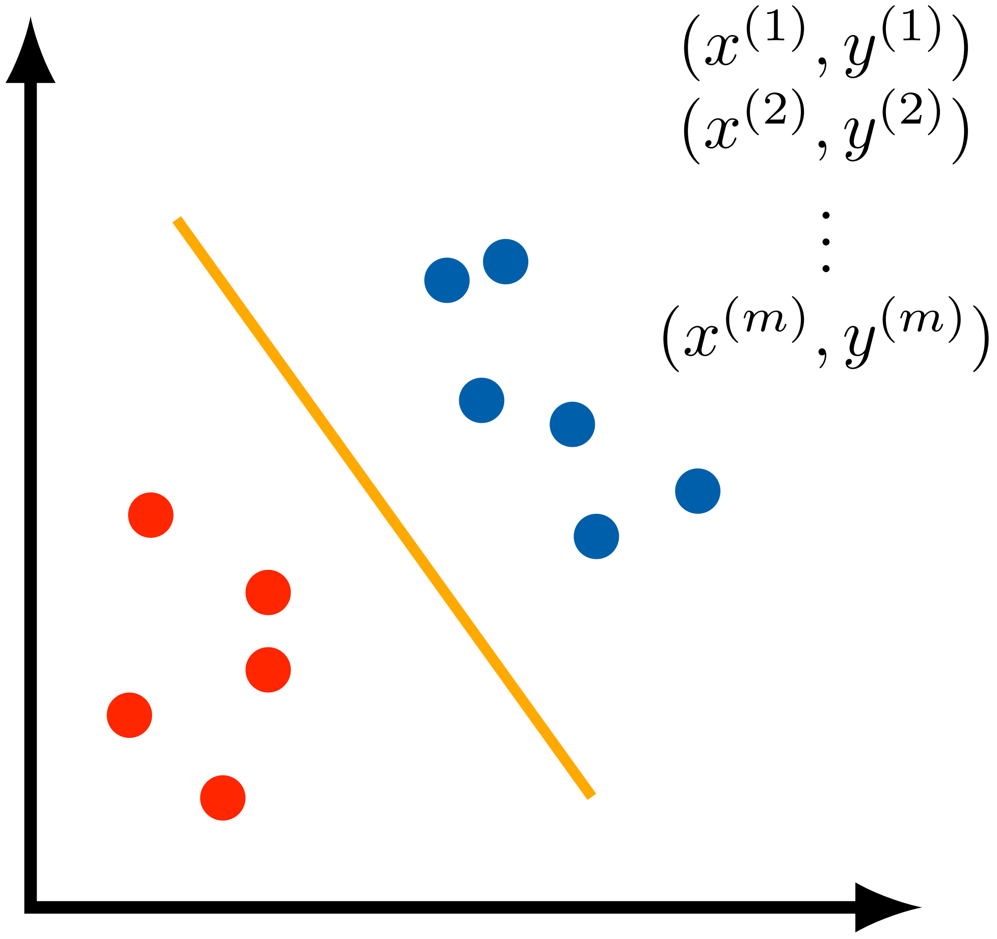
Machine Learning Lab
University of Freiburg

Clustering

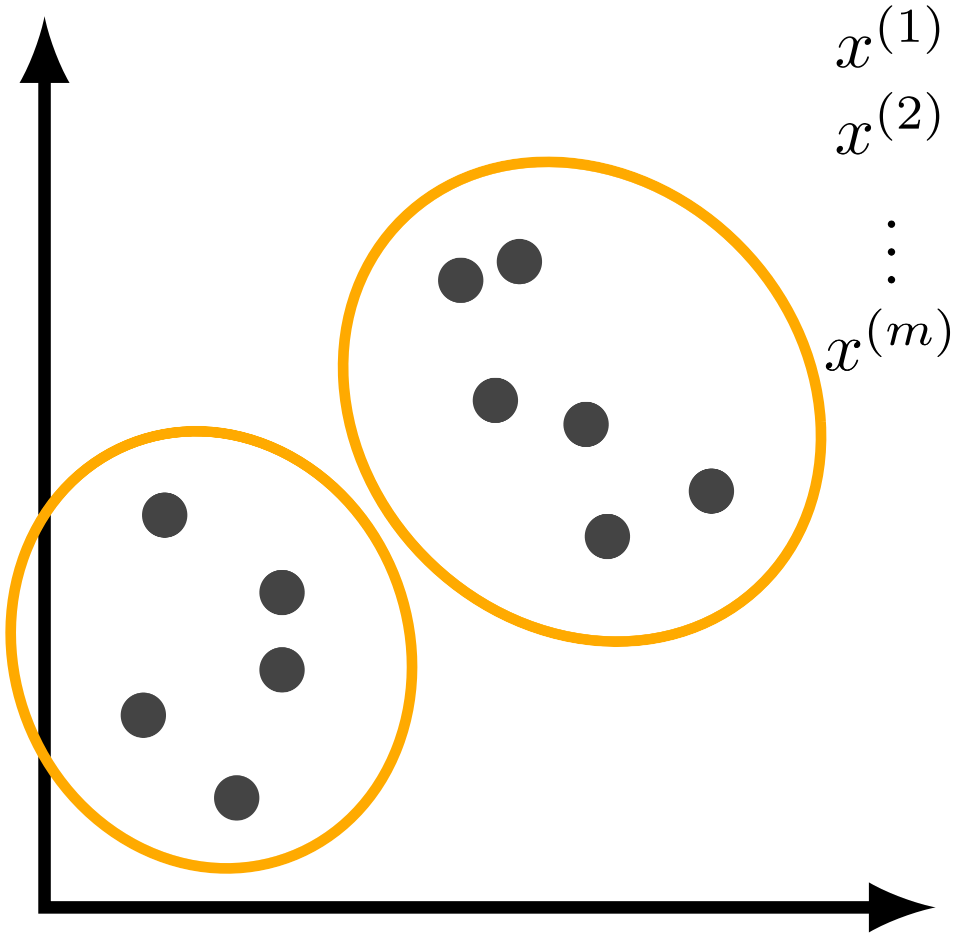
Machine Learning
Summer 2012

Manuel Blum

Supervised vs. Unsupervised Learning

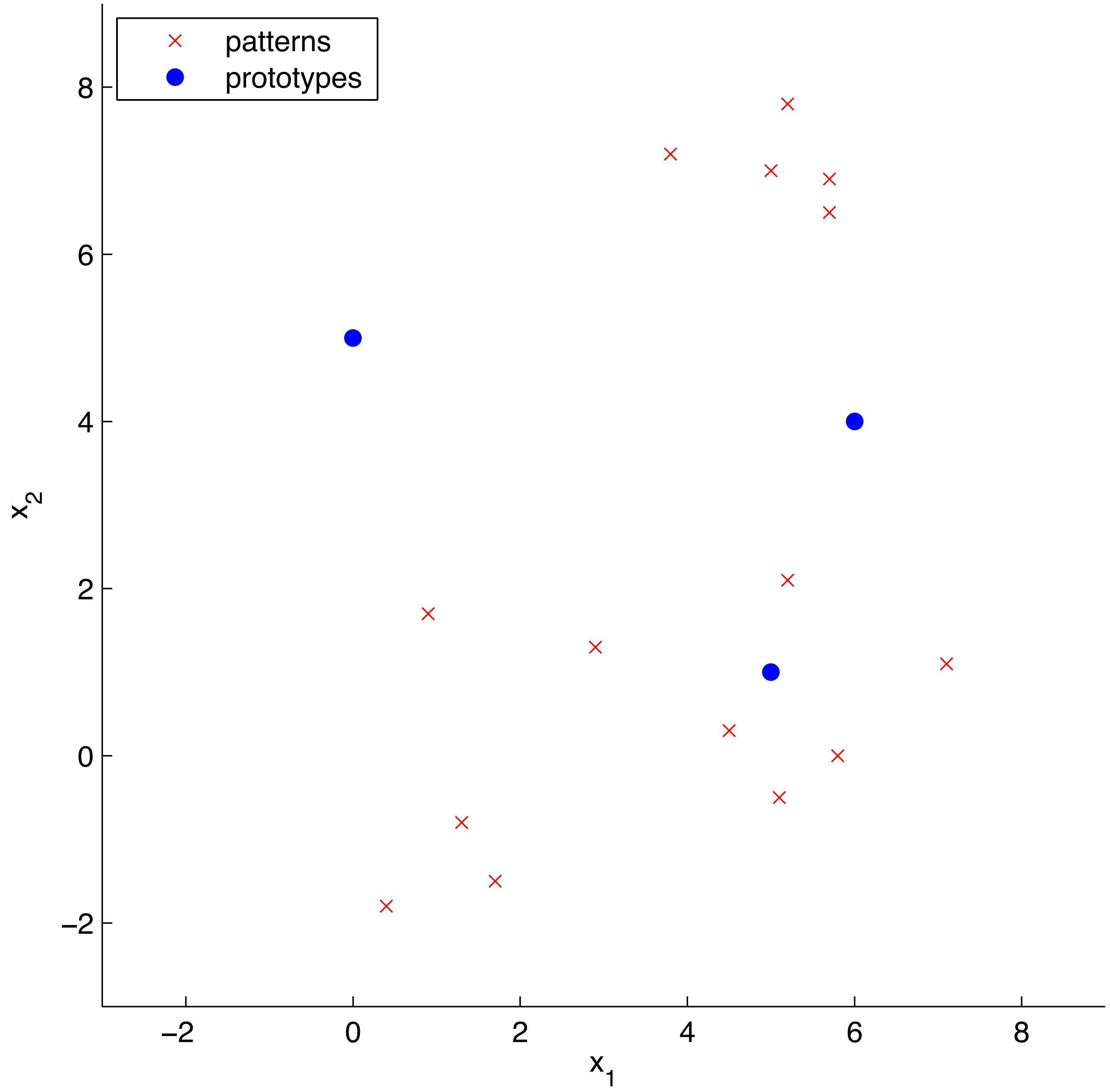


Supervised Learning



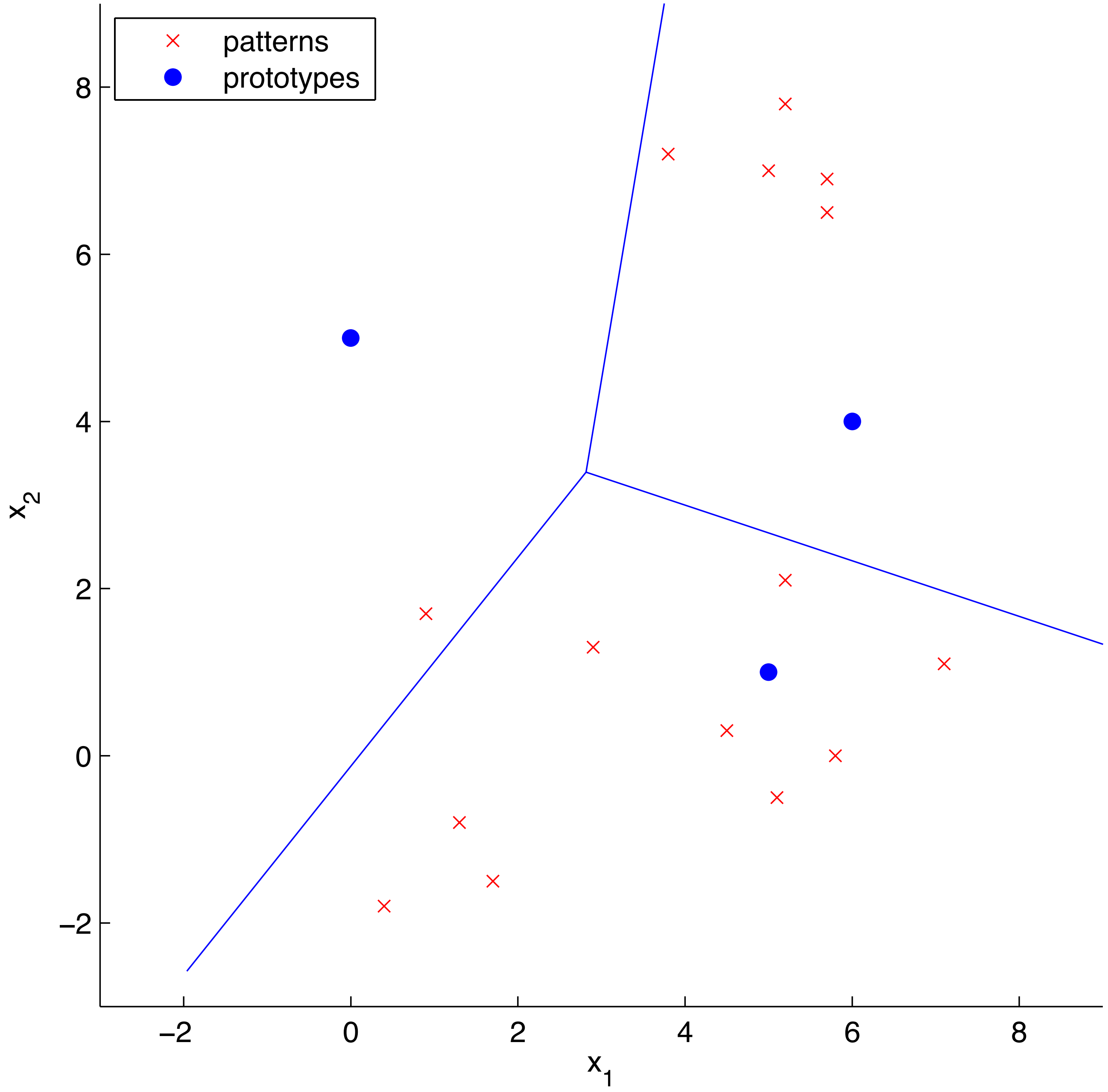
Unsupervised Learning

K-means Algorithm Example



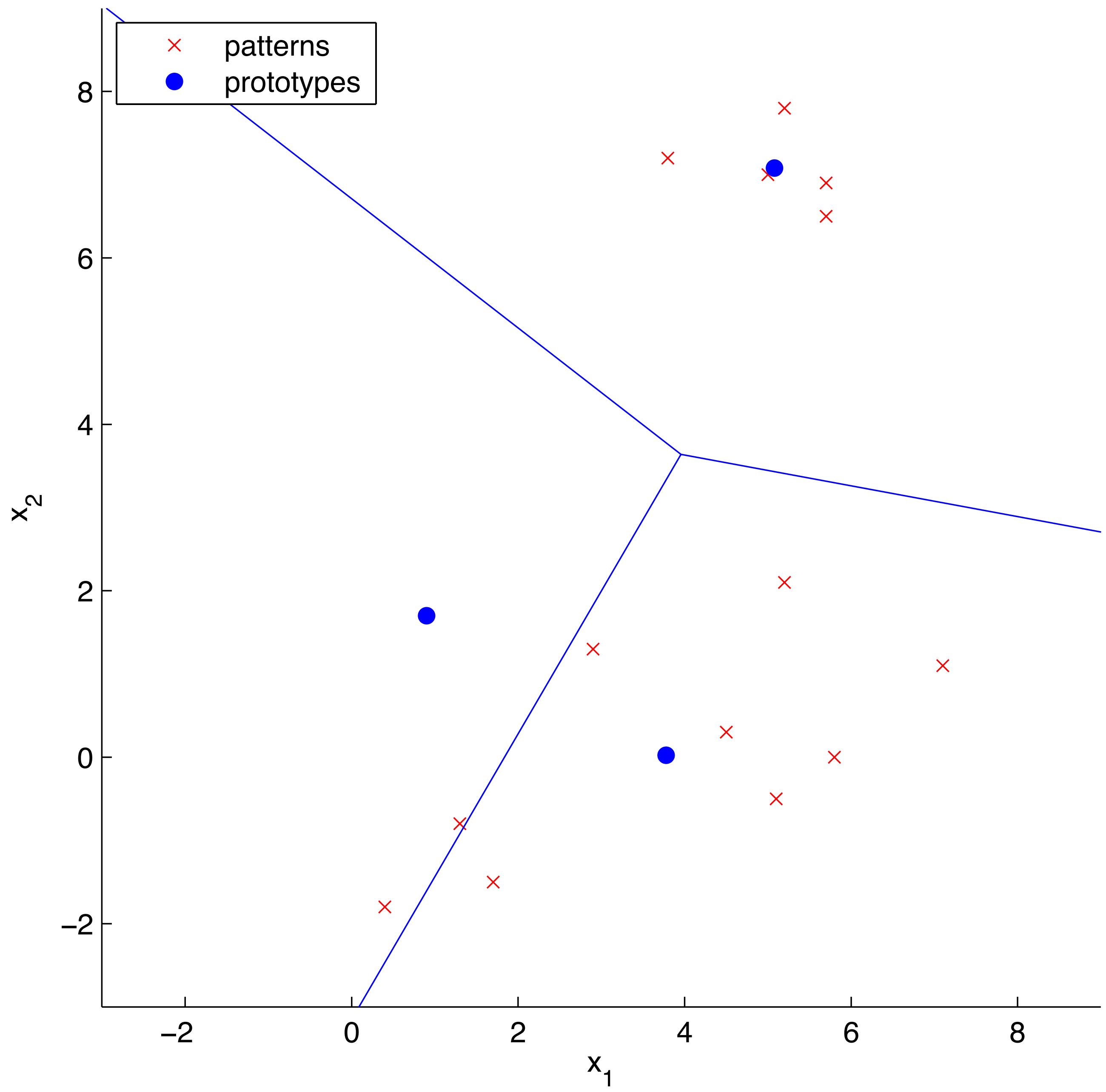
Initialization of cluster centroids

K-means Algorithm Example



Iteration 1:
compute closest centroids

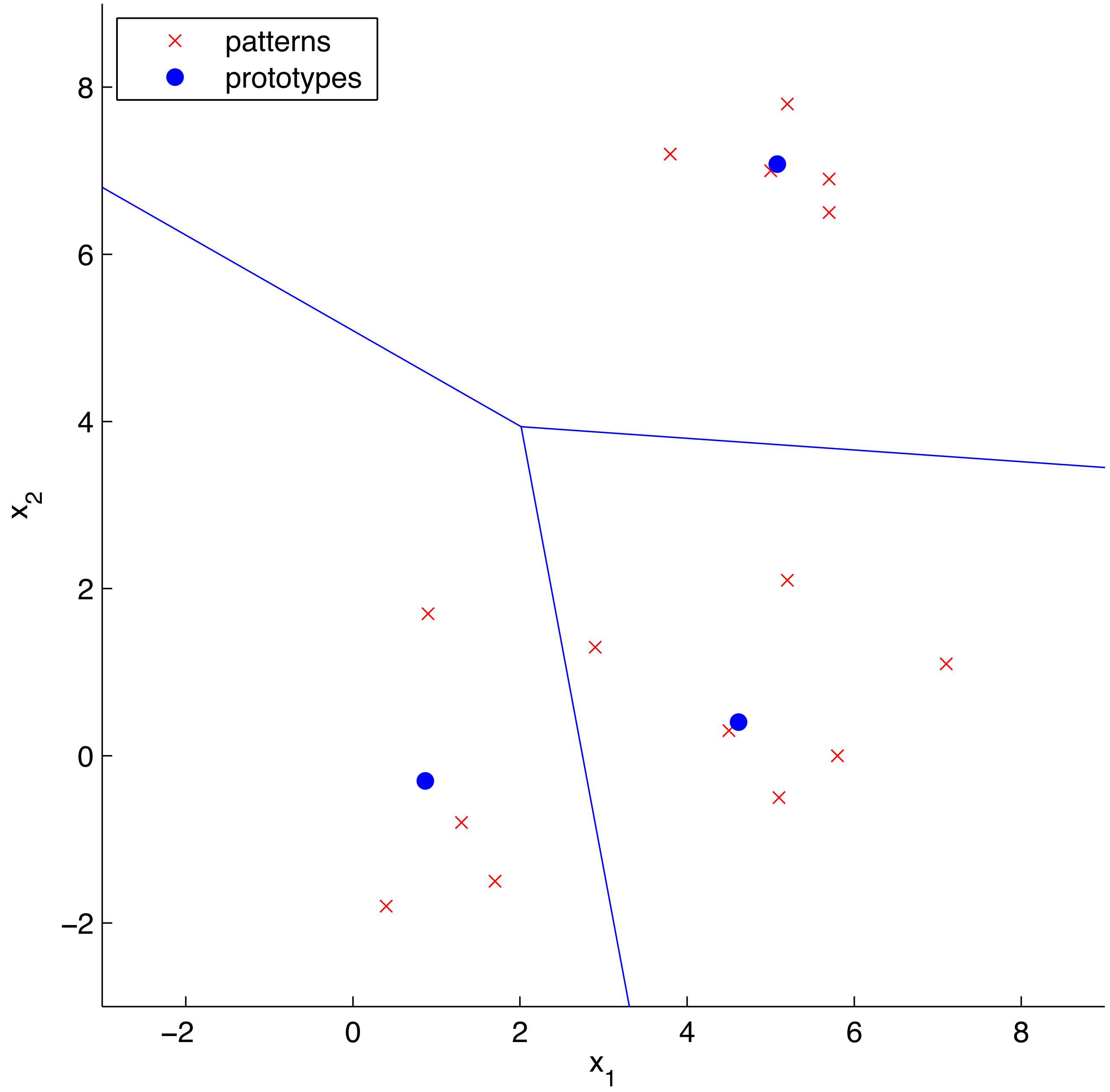
K-means Algorithm Example



Iteration 1:
compute closest centroids
move controids to mean of
assigned points

Iteration 2:
compute closest centroids

K-means Algorithm Example

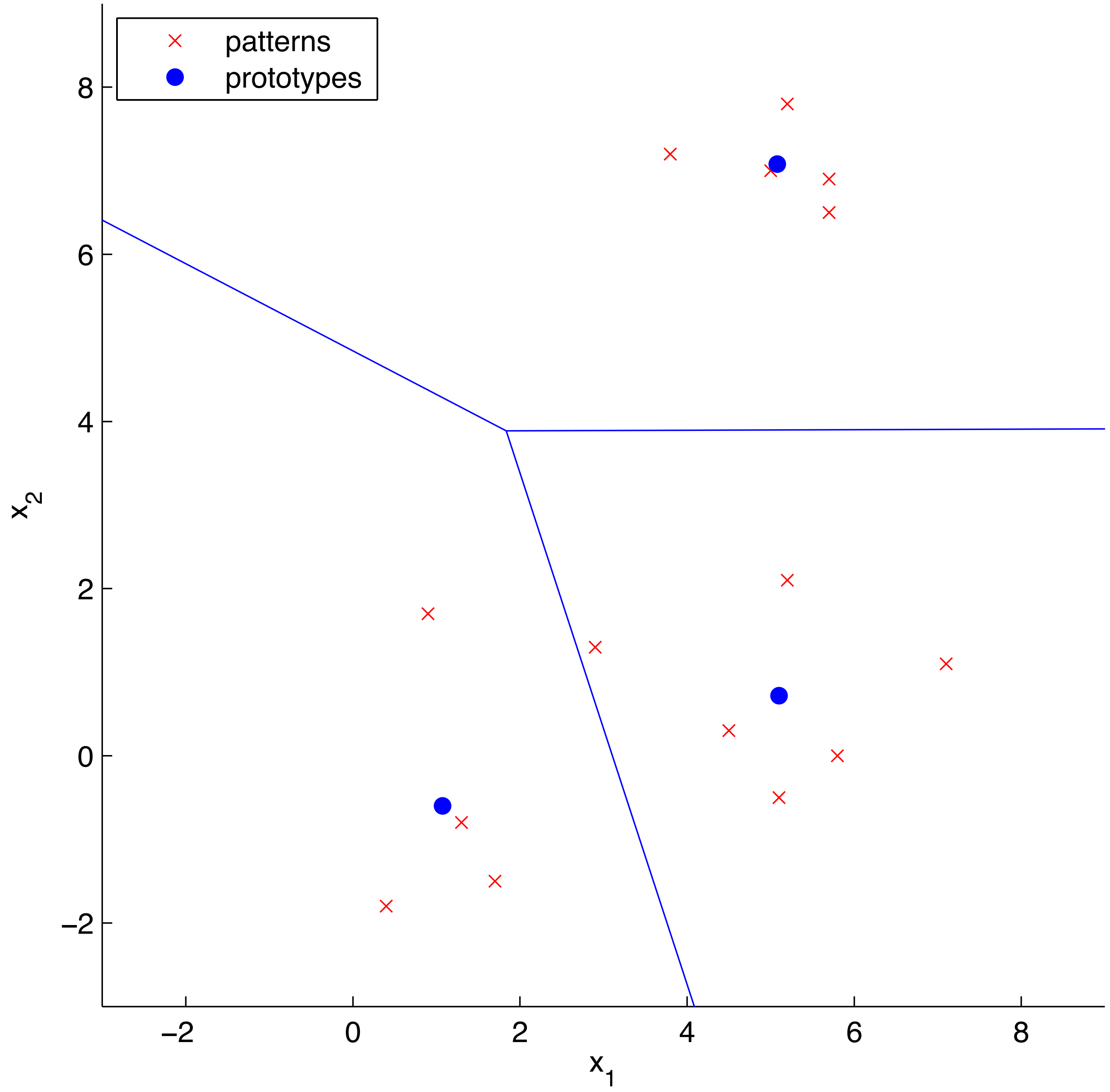


Iteration 1:
compute closest centroids
move controids to mean of assigned points

Iteration 2:
compute closest centroids
move centroids

Iteration 3:
compute closest centroids

K-means Algorithm Example



Iteration 1:
compute closest centroids
move controids to mean of
assigned training patterns

Iteration 2:
compute closest centroids
move centroids

Iteration 3:
compute closest centroids
move centroids

Iteration 4:
algorithm converges

K-means Algorithm

```
input: K, $\{x^{(1)}, x^{(2)}, ..., x^{(m)}\}$, $x^{(i)} \in R^n$
randomly initialize K cluster controids $\mu_1, \mu_2, ..., \mu_K \in R^n$
do
    for i = 1 to m
        $c^{(i)}$ := index of cluster centroid closest to $x^{(i)}$
    for k = 1 to K
        $\mu_k$ := mean of training patterns assigned to cluster k
until convergence
```

K-means Optimization Objective

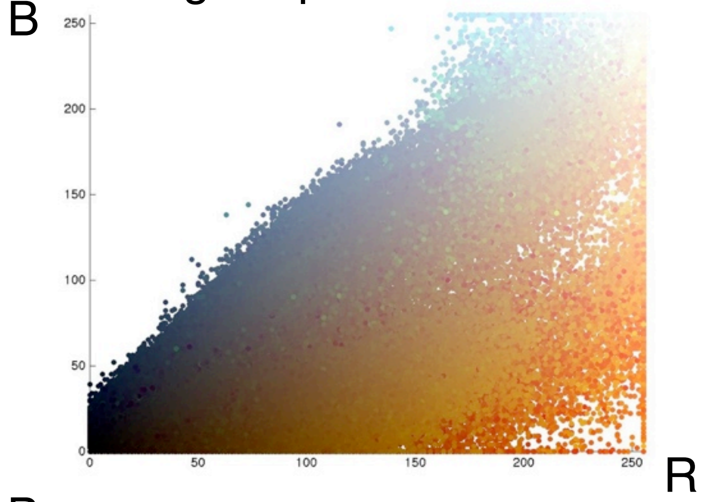
$$J\left(c^{(1)},\ldots,c^{(m)},\mu_1,\ldots,\mu_K\right)=\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)}-\mu_{c^{(i)}}\right\|^2$$

- cluster assignment step minimizes *J* w.r.t. $c^{(1)}$, ..., $c^{(m)}$
- moving the centroids minimizes *J* w.r.t. μ_1,..., μ_K
- the objective function is monotonically decreasing towards a local minimum of *J*
- K-means always converges within finite time

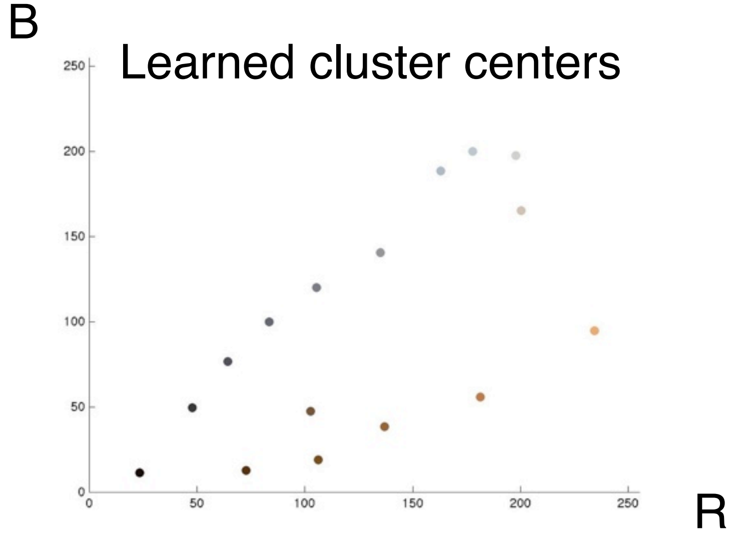
Application: Color Quantization

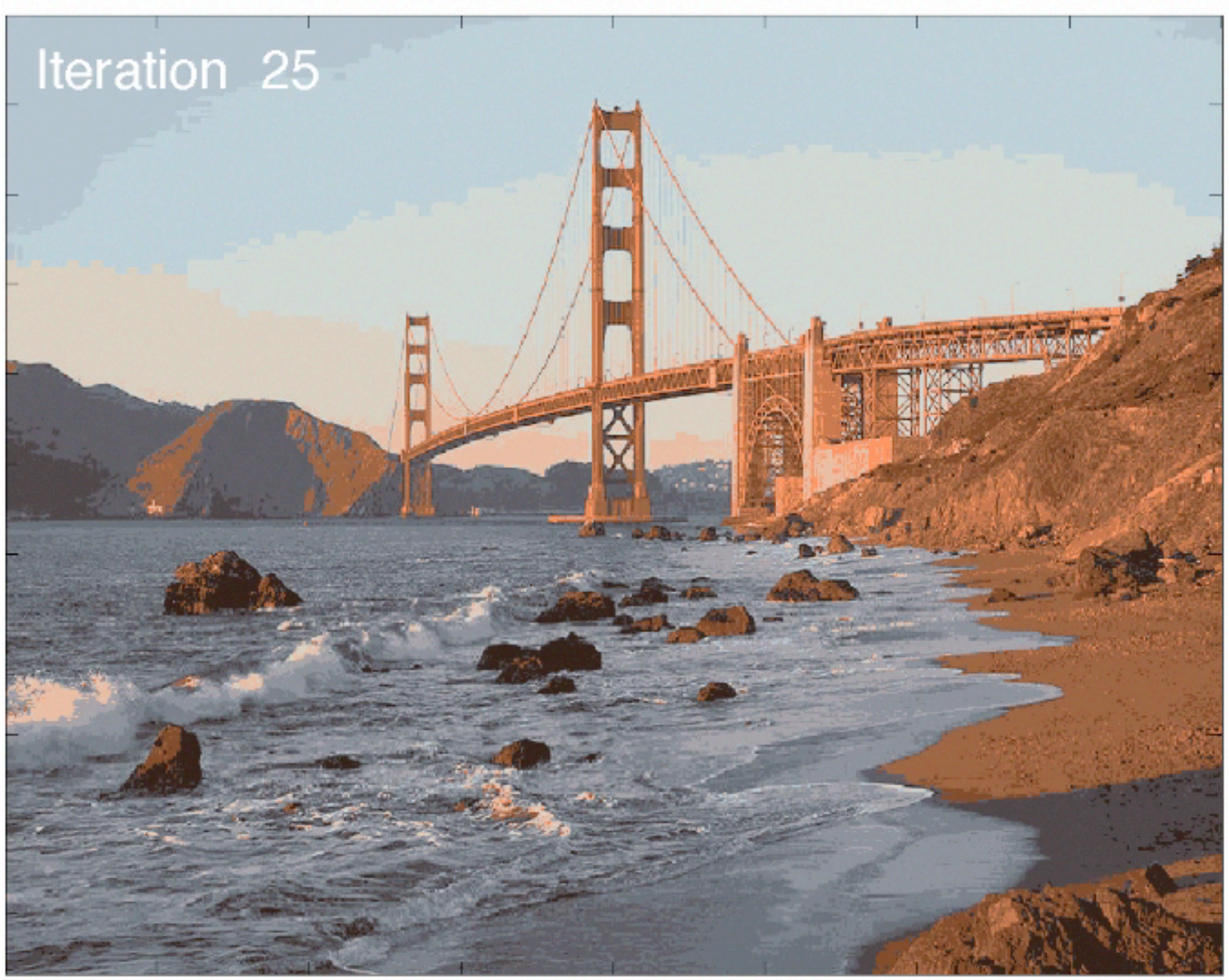
- reduce the number of distinct colors in an image by clustering the pixels
- the pixels of the original image are used as training patterns $x^{(i)}$
- K controls the number of colors in the output image
- K-means will learn the K most typical colors in the image
- the original pixels can be replaced by the closest prototypes after training

Original pixels



Learned cluster centers





Random Initialization

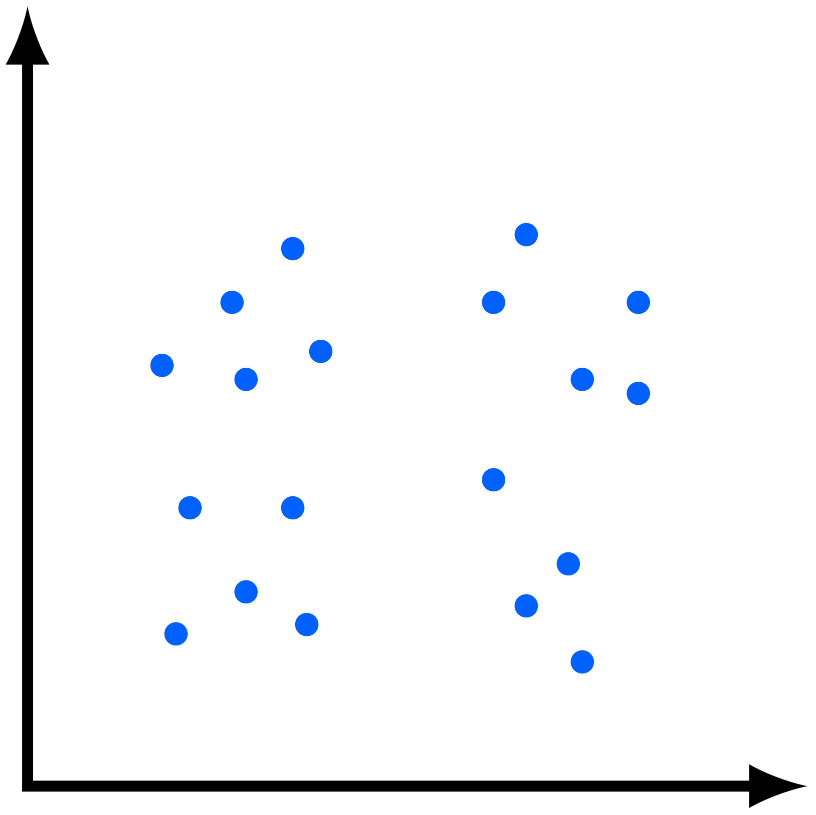
- Forgy‘s method:
  - initialize centroids to K randomly picked training patterns

- the random partition method:
  - randomly assign a cluster to each training pattern
  - move centroids to the means of the randomly assigned points

- *problem:* the performance of K-means heavily depends on the initial cluster centers

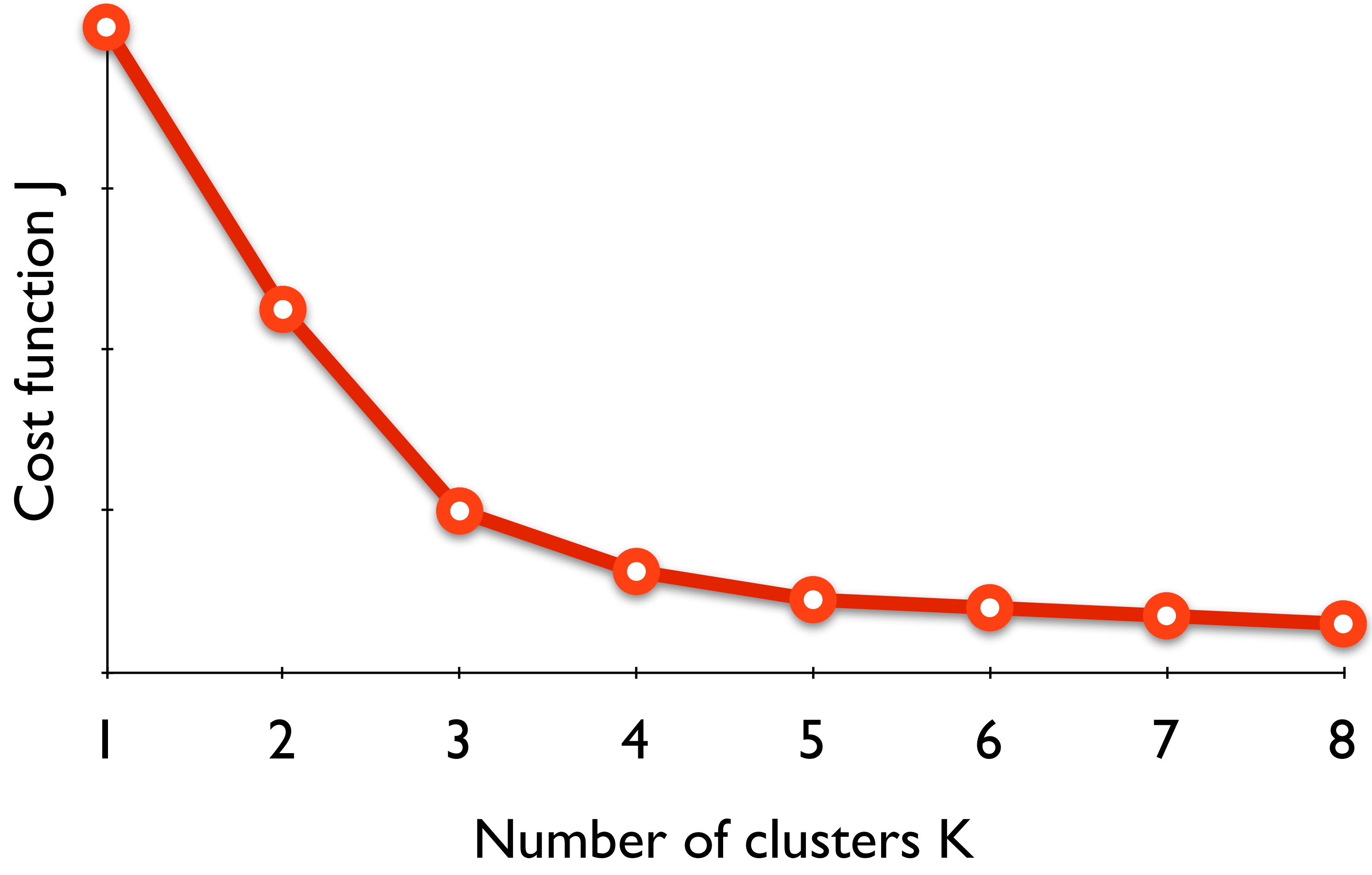
- simple solution:
  - run K-means multiple times using different random initializations
  - choose the clustering that minimizes the cost function *J*

How to choose K



- correct choice of k is often ambiguous
- most often K is specified by hand

- the elbow method:

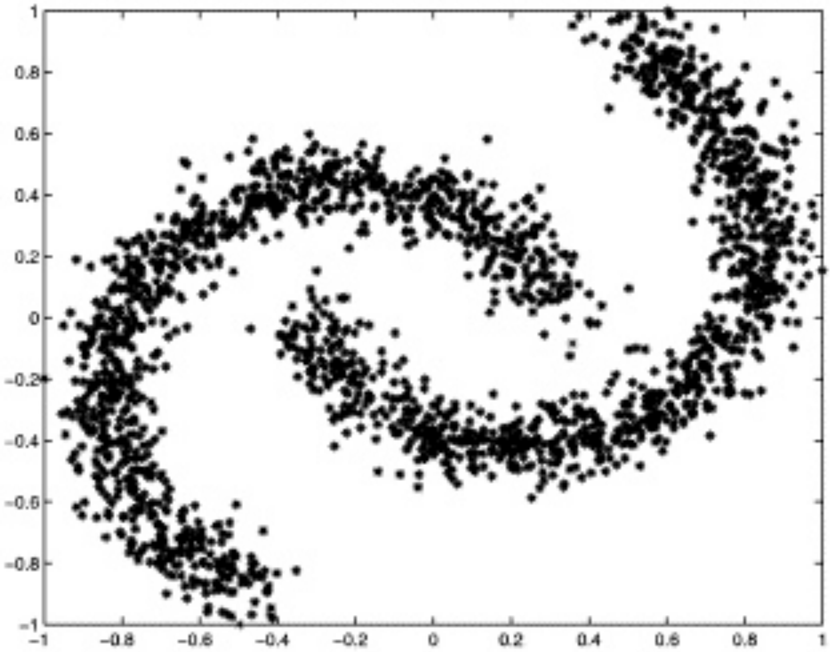


Final Remarks

- results of K-means heavily depend on the scaling of the data

- Euclidean distance must be a meaningful measure of similarity for the dataset

- K-means will rarely work for high dimensional data (d > 20)



- cluster centroids are also called prototypes or codebook vectors

- the set of prototypes is called codebook