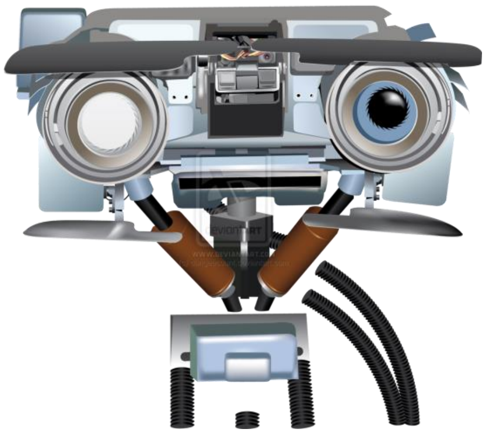
Today

- Unsupervised Learning

Reminders: PS1 is due tonight @11:59pm
Pre-lecture Material for Friday

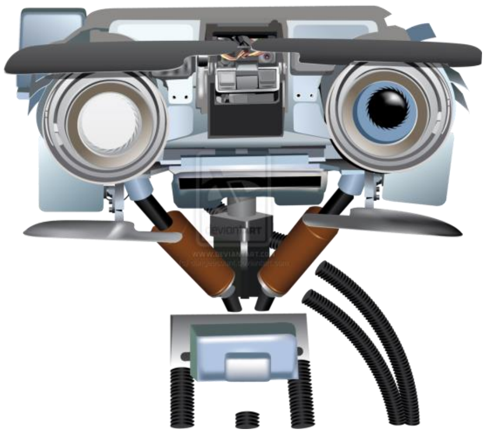
Announcement: Lab 2: Anaconda Setup
PS2 will be posted on Friday
Midterm Room Scheduling Ongoing



Unsupervised Learning I

Today

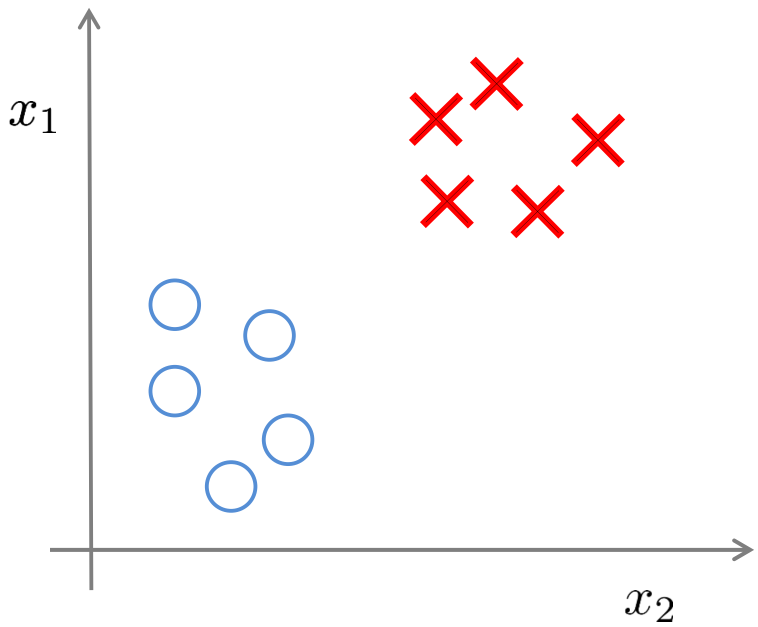
- Unsupervised learning
  – K-Means clustering
  – Dimensionality reduction



Unsupervised Learning I

Clustering

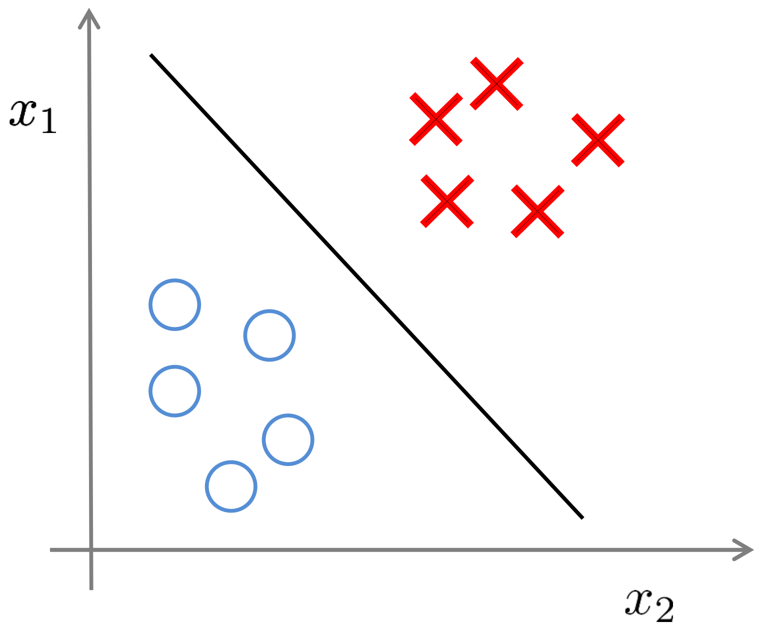
Supervised learning



Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$

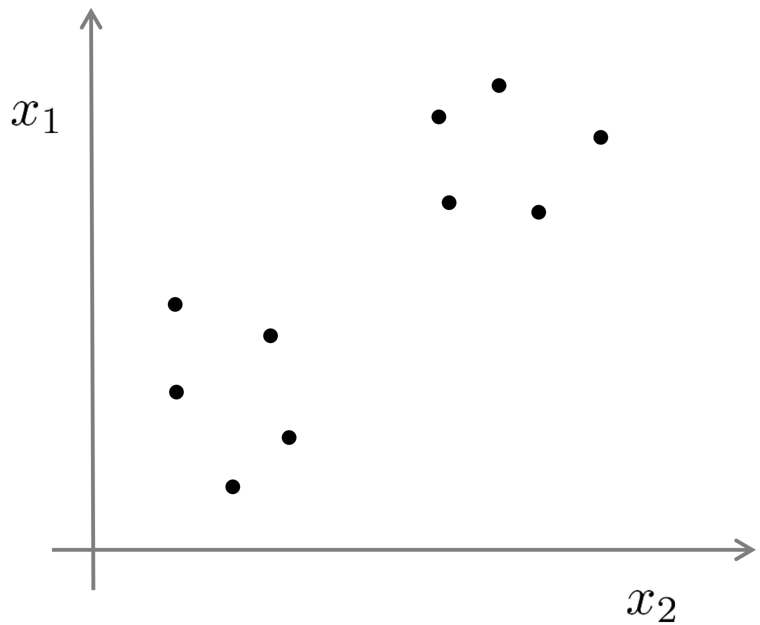
slide credit: Andrew Ng

Goal of Supervised learning



Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$

Unsupervised learning

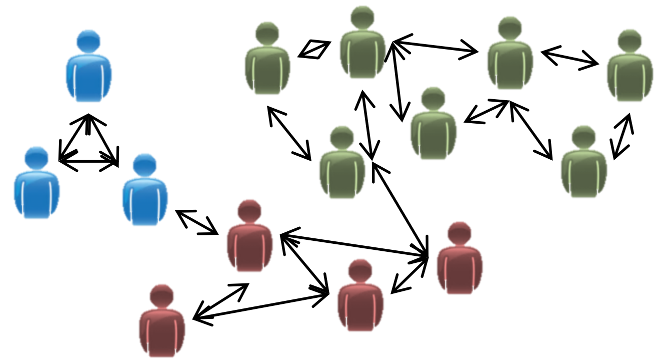


Training set: $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$

slide credit: Andrew Ng

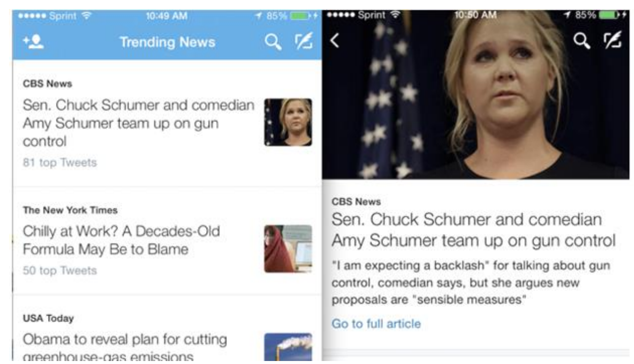
Clustering

Gene analysis

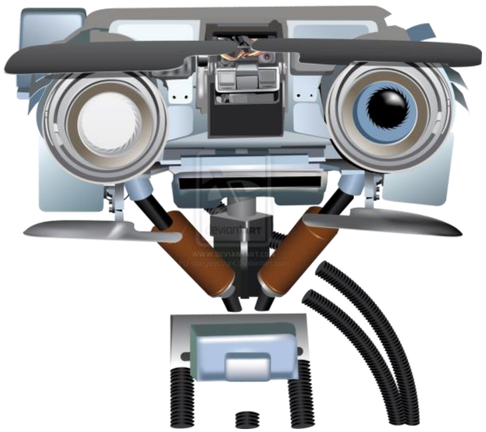


Social network analysis

Types of voters

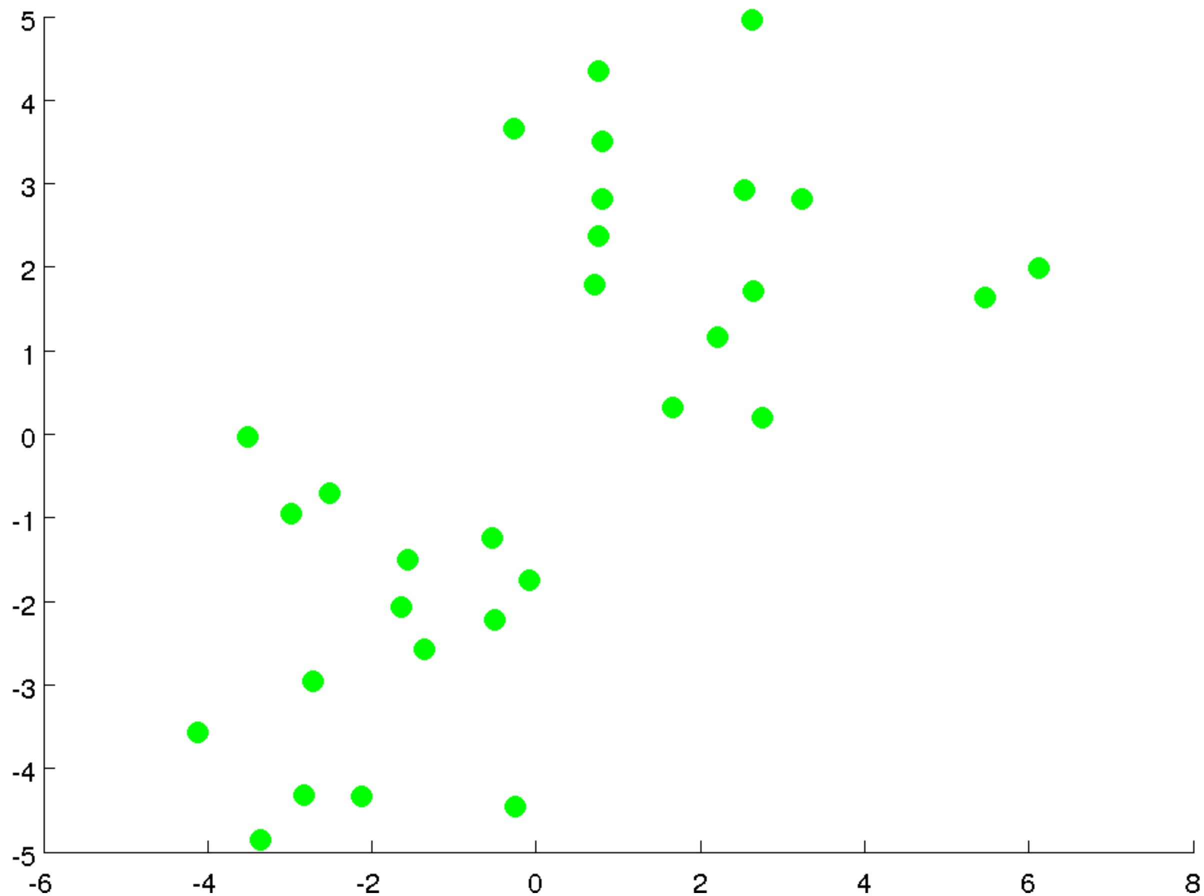


Trending news

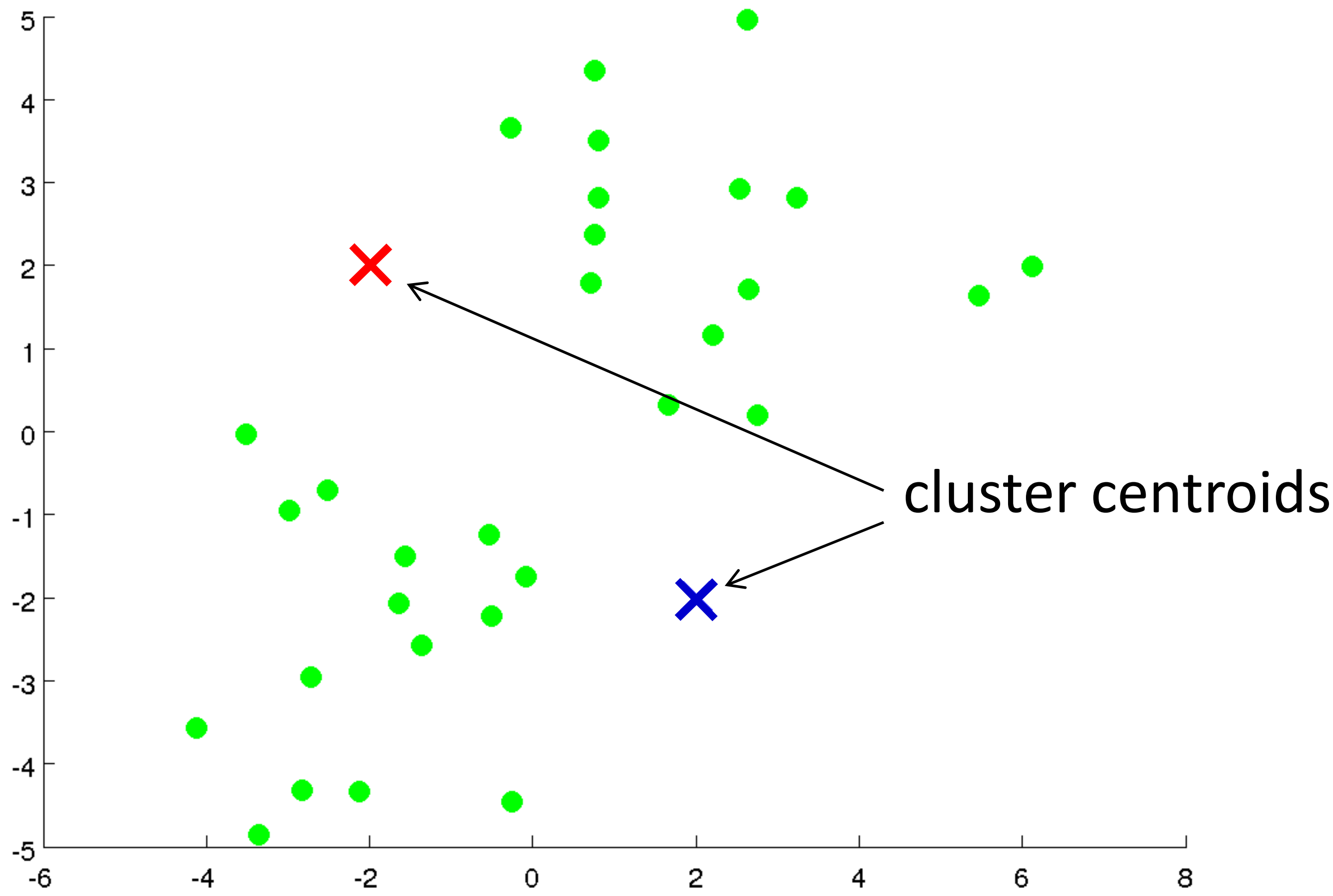


Unsupervised Learning I

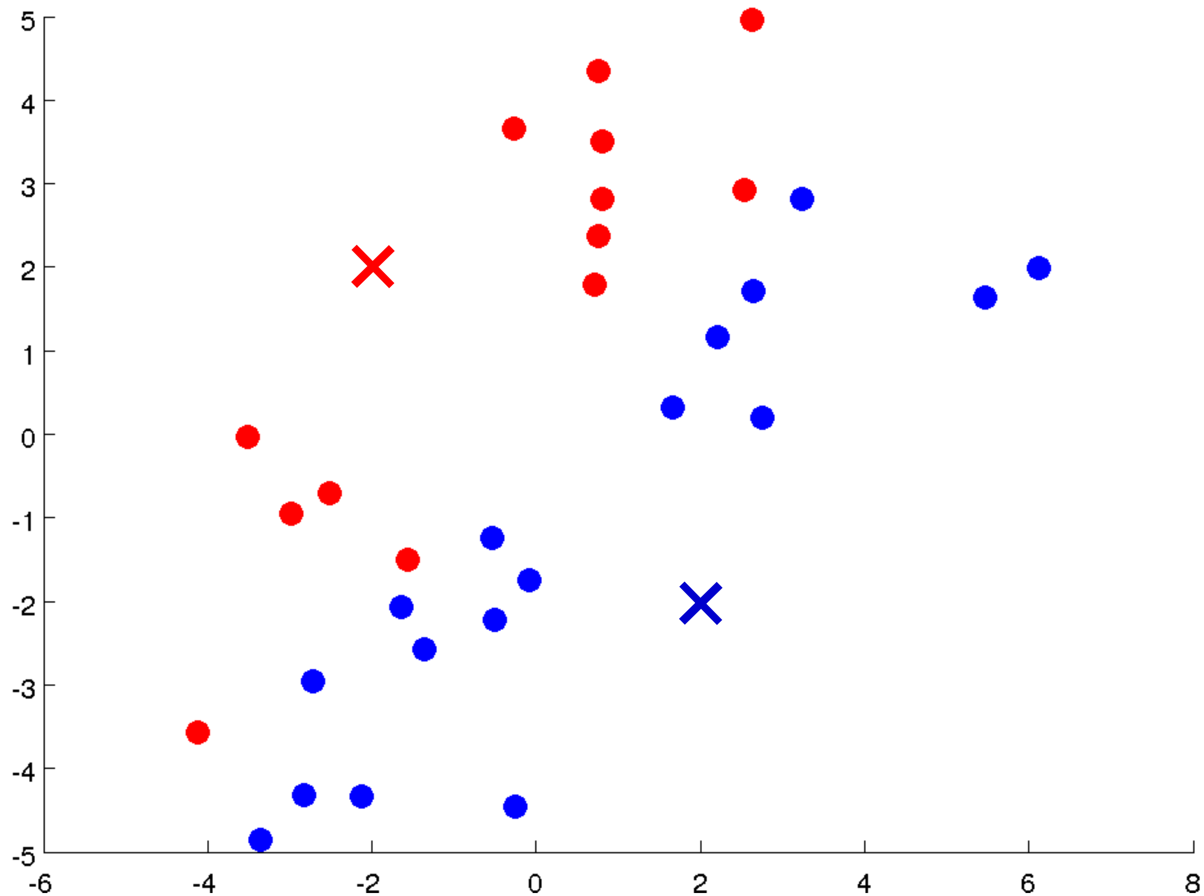
K-means Algorithm



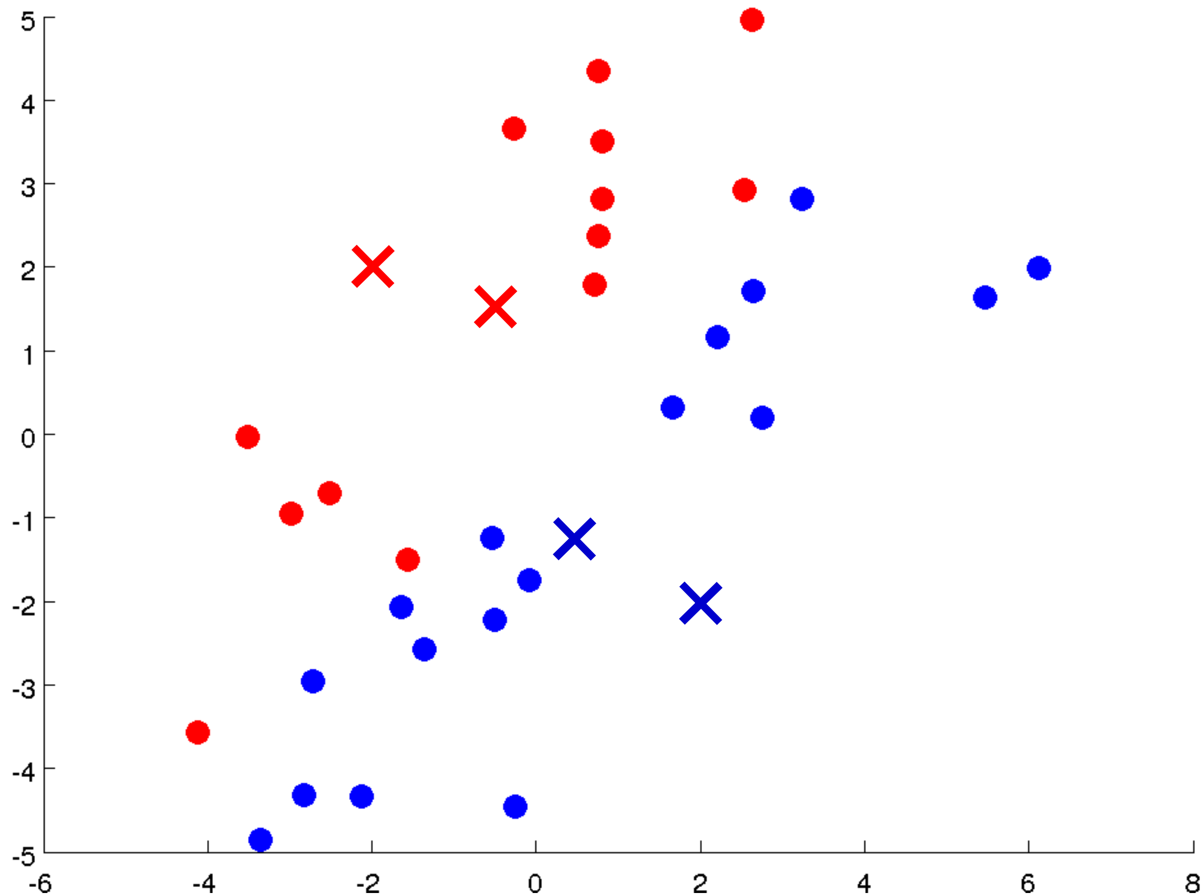
slide credit: Andrew Ng



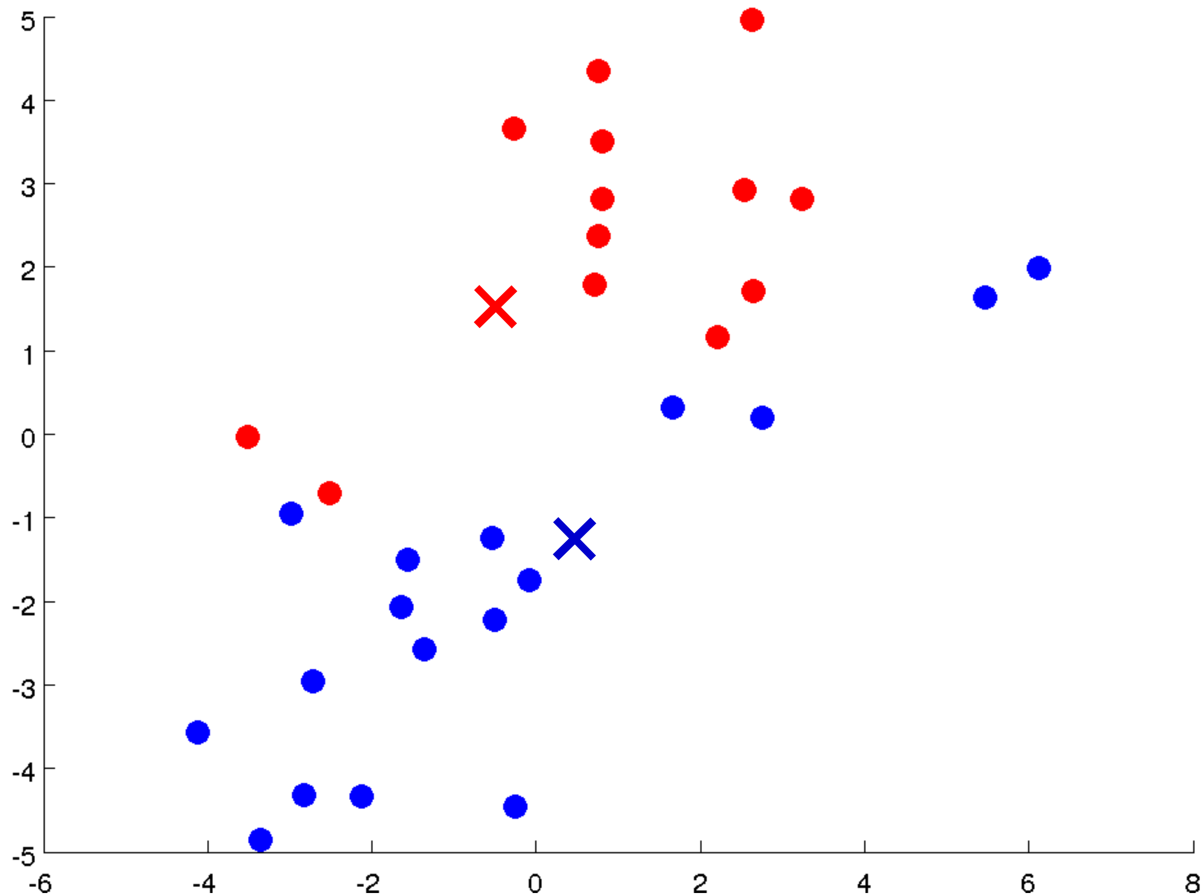
slide credit: Andrew Ng



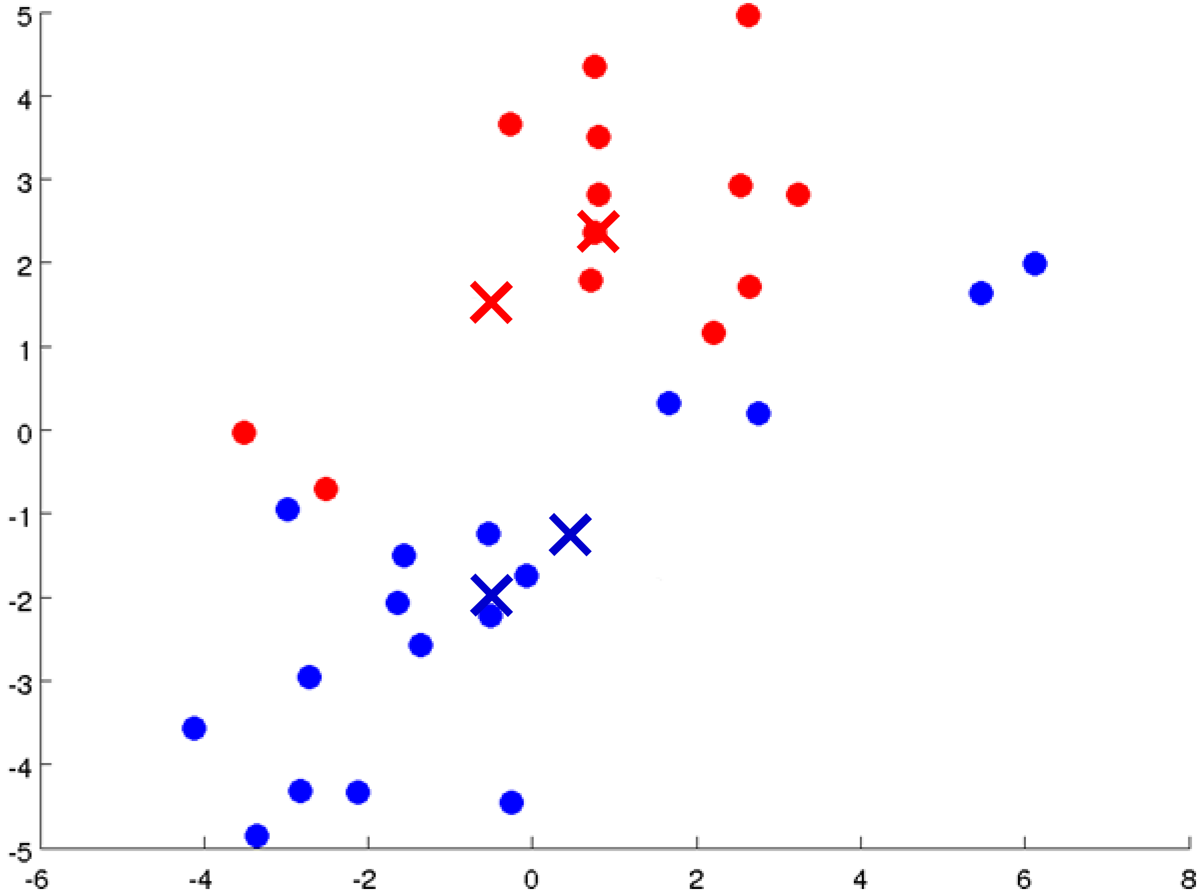
slide credit: Andrew Ng



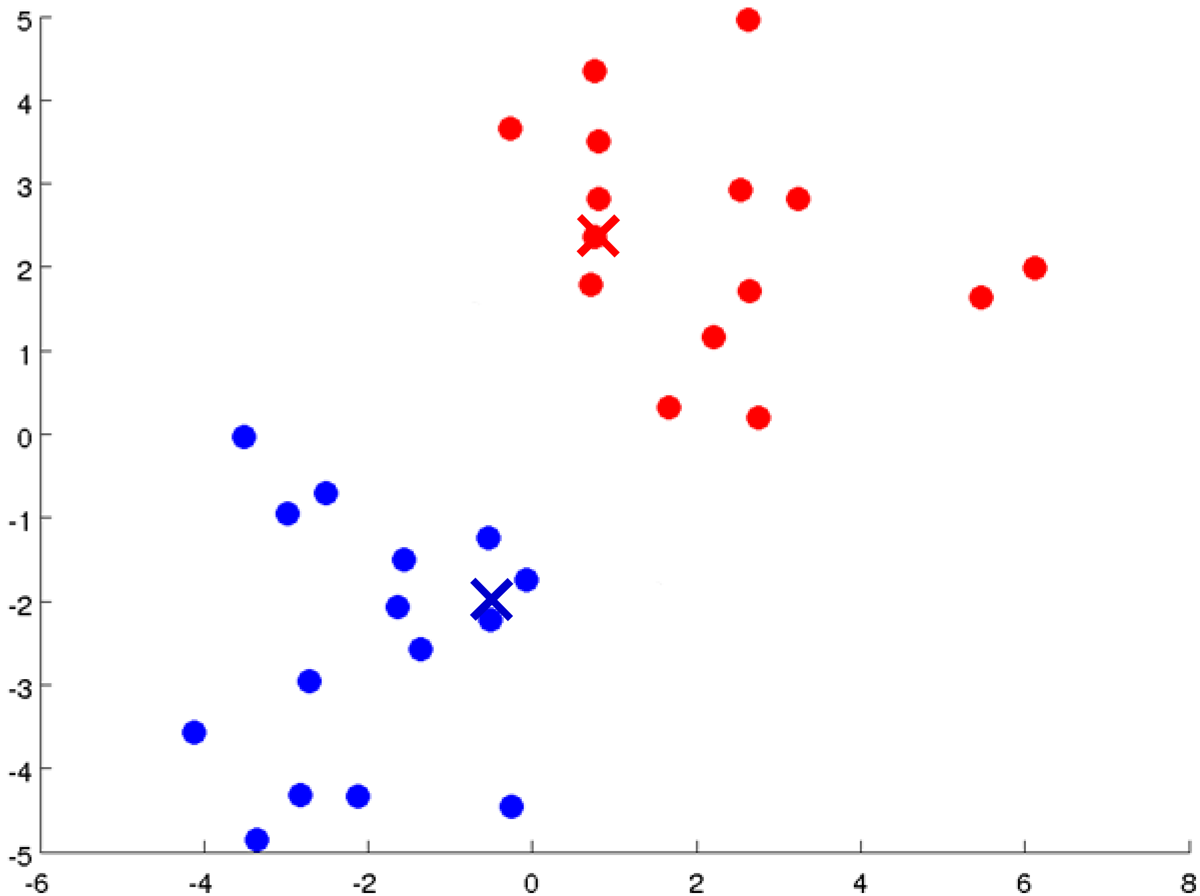
slide credit: Andrew Ng



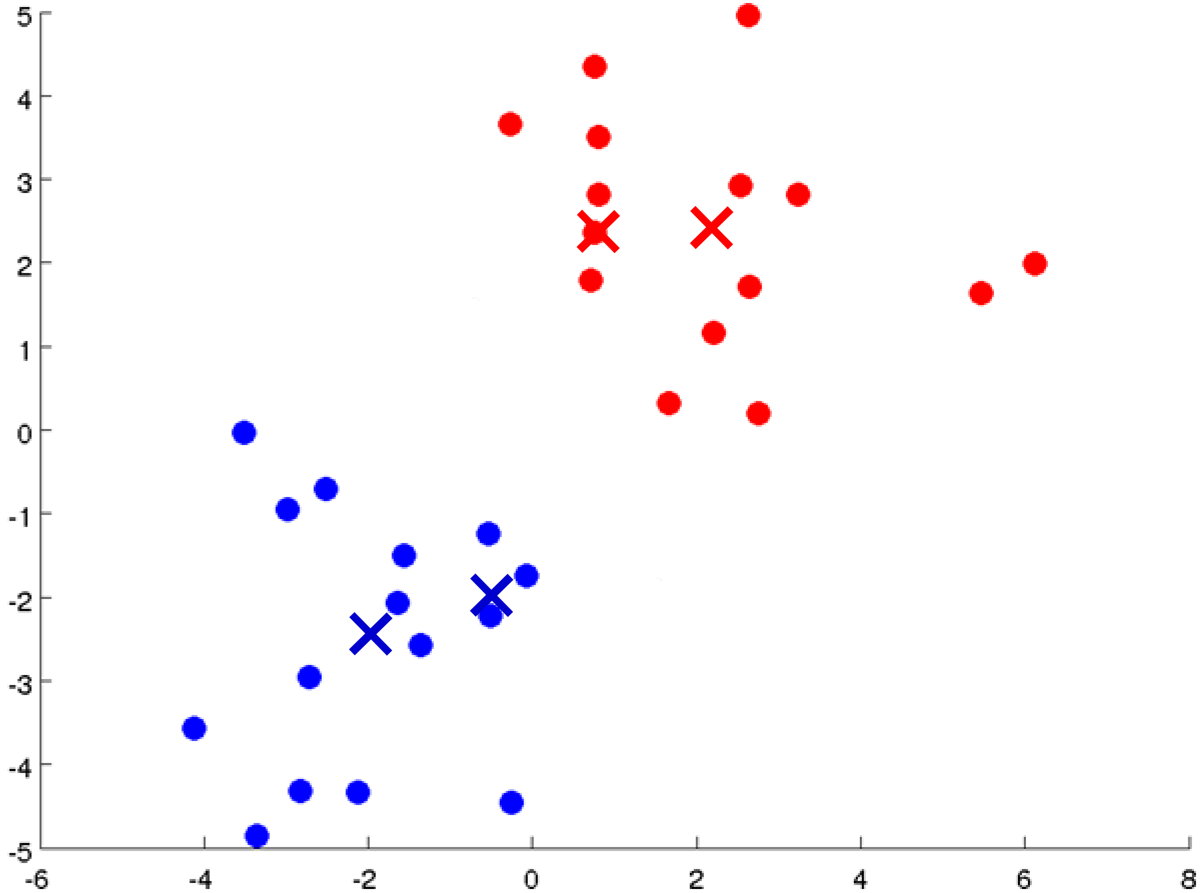
slide credit: Andrew Ng



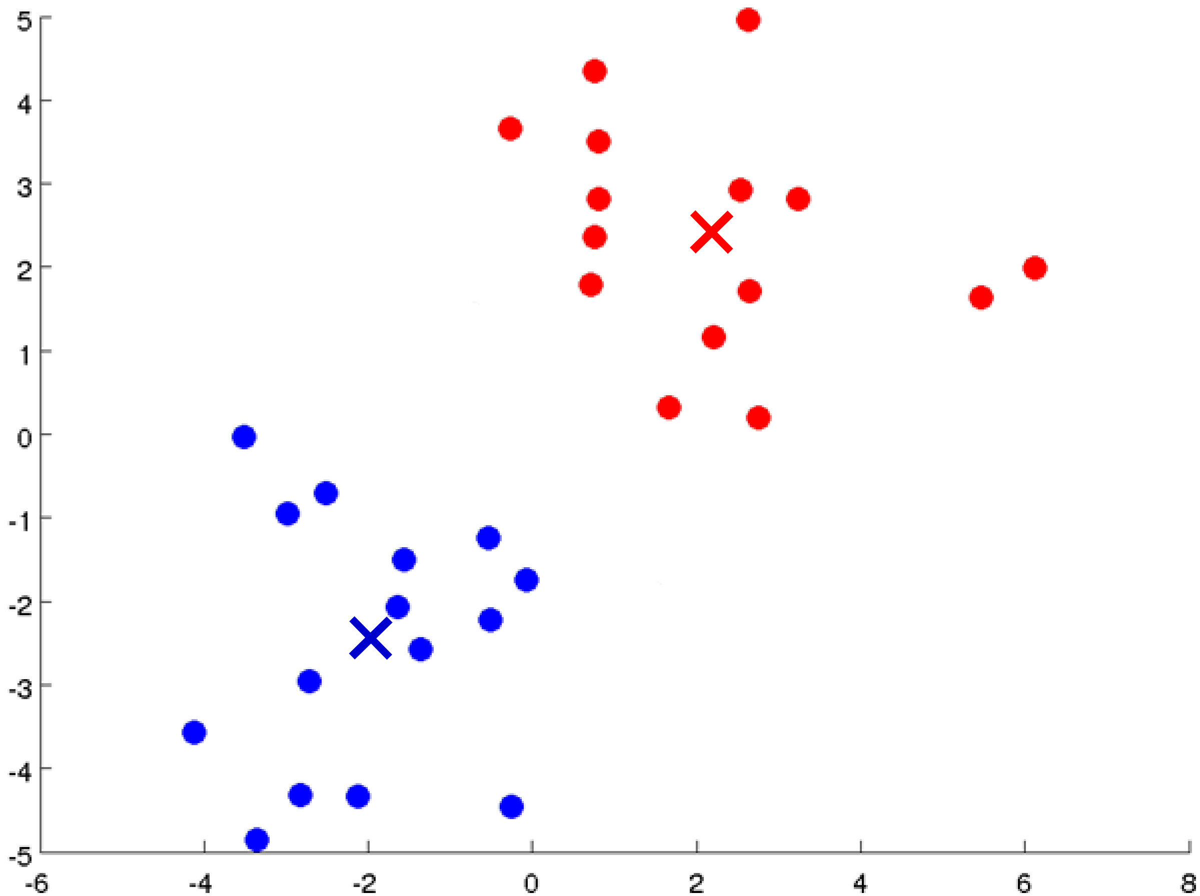
slide credit: Andrew Ng



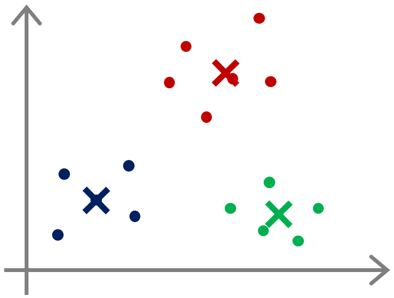
slide credit: Andrew Ng



slide credit: Andrew Ng



slide credit: Andrew Ng



K-means algorithm

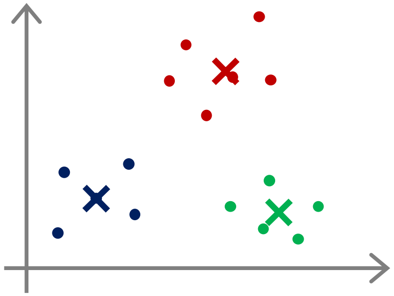
Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

$x^{(i)} \in \mathbb{R}^n$ (drop $x_0 = 1$ convention)

slide credit: Andrew Ng

K-means algorithm



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

Repeat {

for i = 1 to m

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

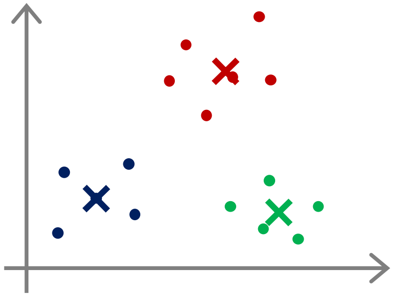
for k = 1 to K

μ_k := average (mean) of points assigned to cluster k

}

slide credit: Andrew Ng

K-means Cost Function



$c^{(i)}$ = index of cluster (1,2,…,K) to which example $x^{(i)}$ is currently assigned

μ_k = cluster centroid k ($\mu_k \in \mathbb{R}^n$)

$\mu_{c^{(i)}}$ = cluster centroid of cluster to which example $x^{(i)}$ has been assigned

Optimization cost: "distortion"

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

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

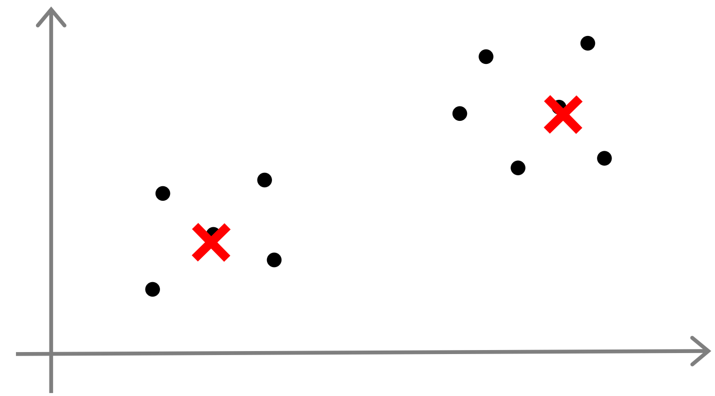
slide credit: Andrew Ng

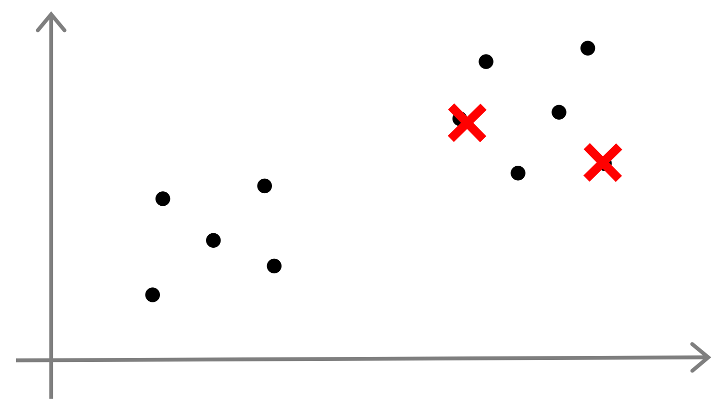
Random initialization

Should have $K < m$

Randomly pick K training examples.

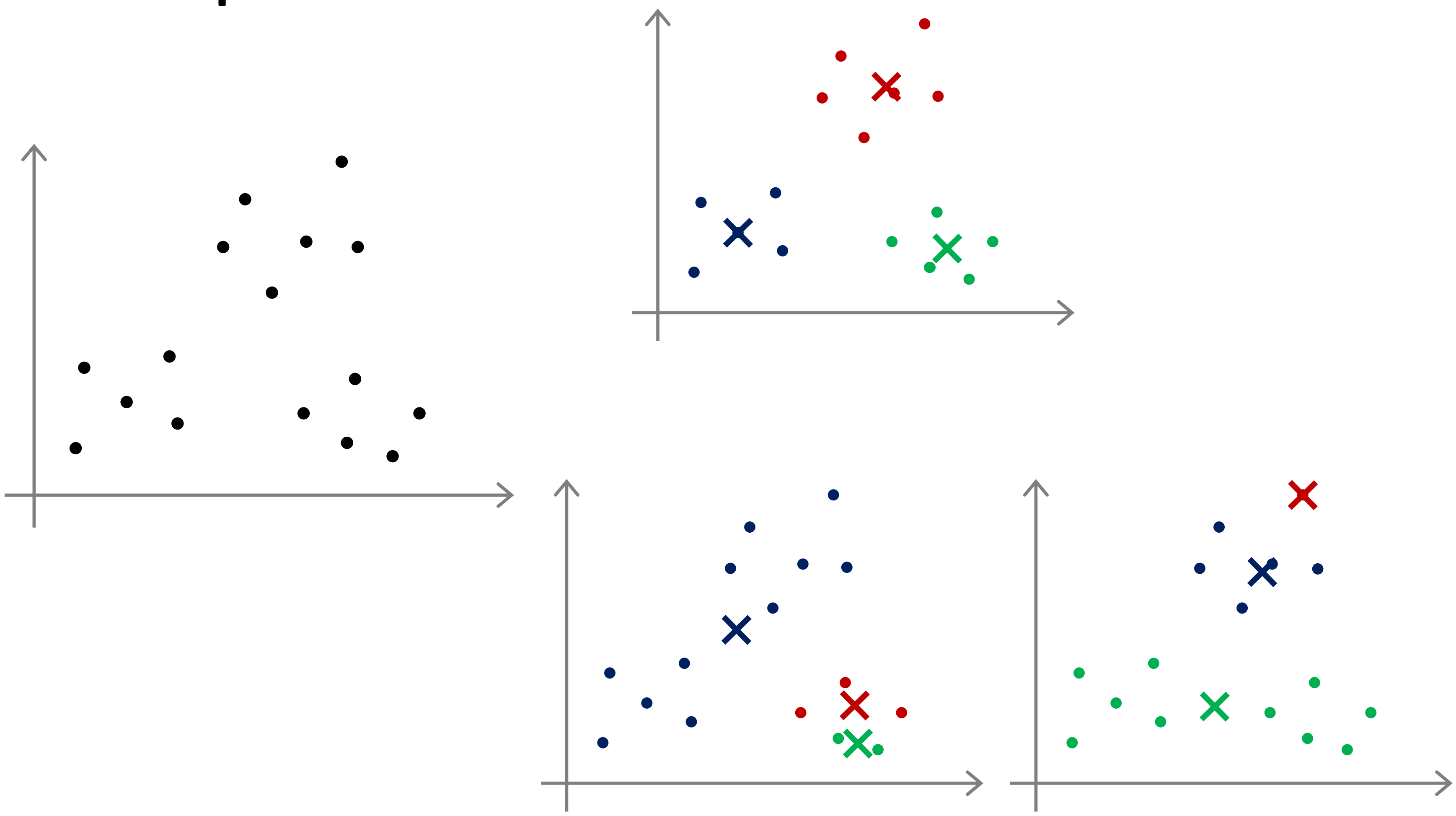
Set $\mu_1, \ldots, \mu_K$ equal to these K examples.





slide credit: Andrew Ng

Local Optima



slide credit: Andrew Ng

Avoiding Local Optima with Random Initialization

For i = 1 to 100 {

Randomly initialize K-means.

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

Compute cost function (distortion)

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

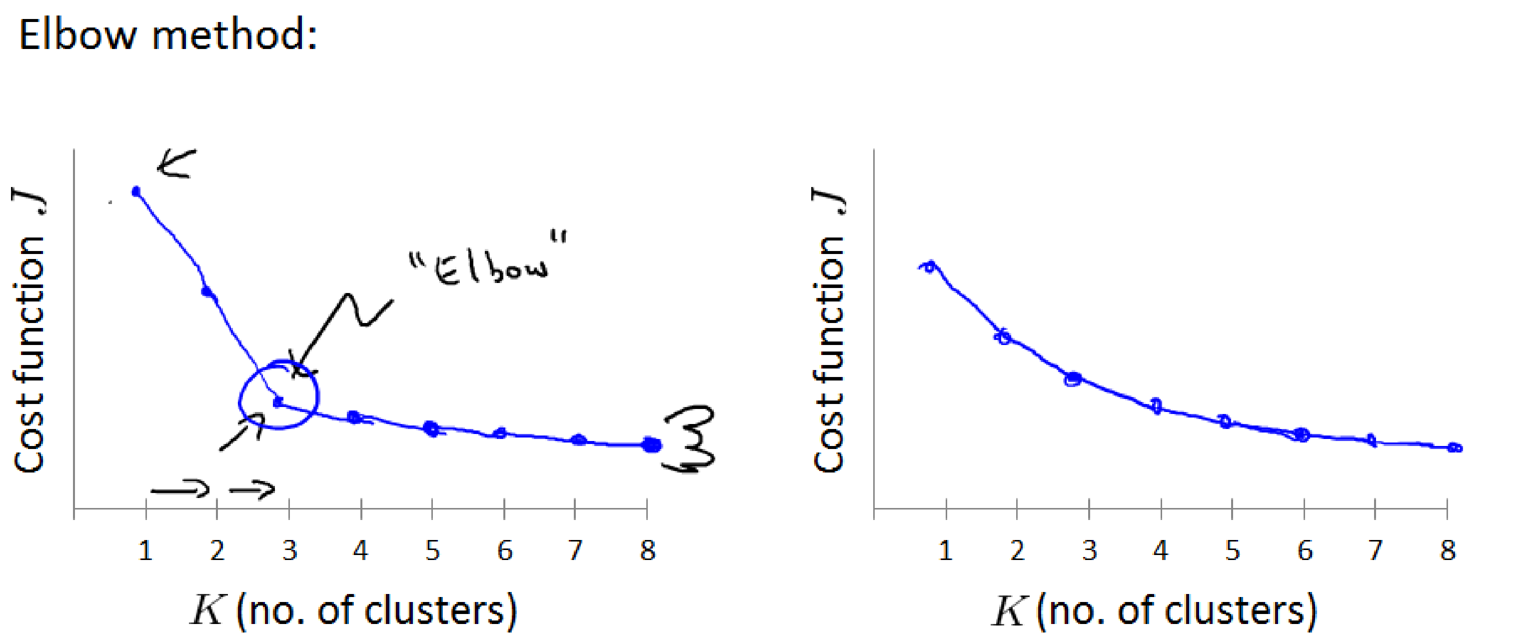
}

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

slide credit: Andrew Ng

How to choose K?

Elbow method:



slide credit: Andrew Ng

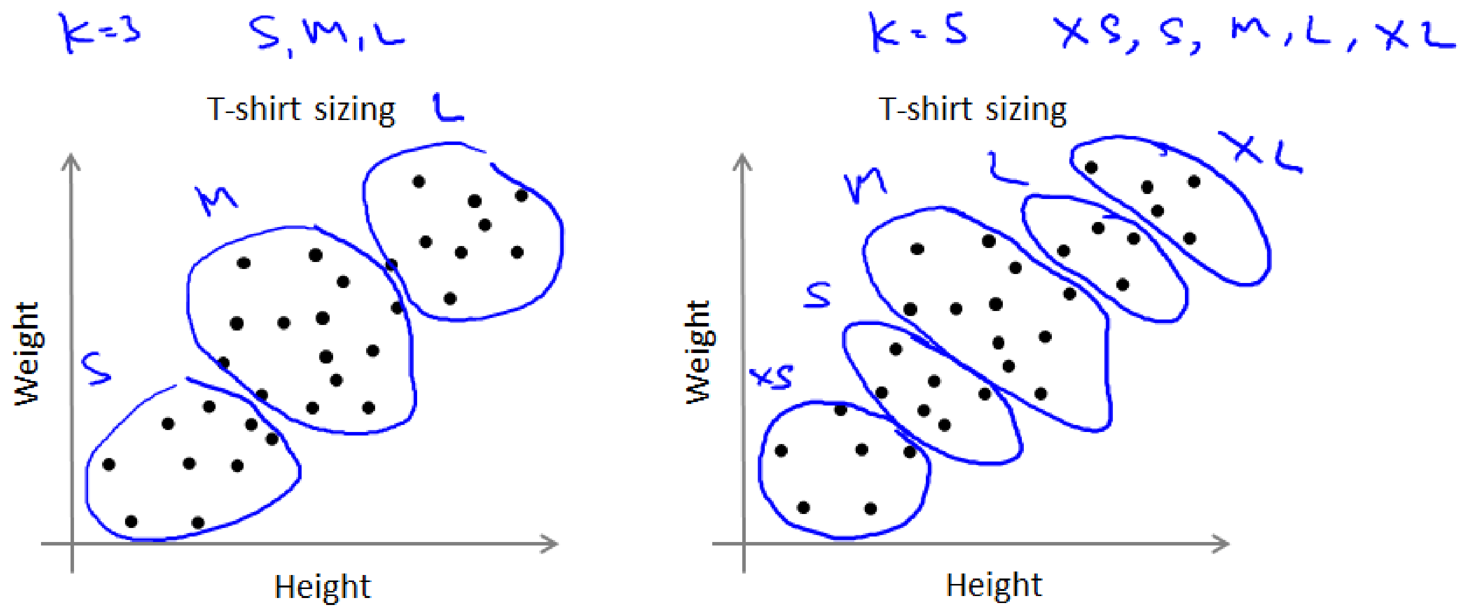
K-means for Non-Separated Clusters

slide credit: Andrew Ng

How to choose K?

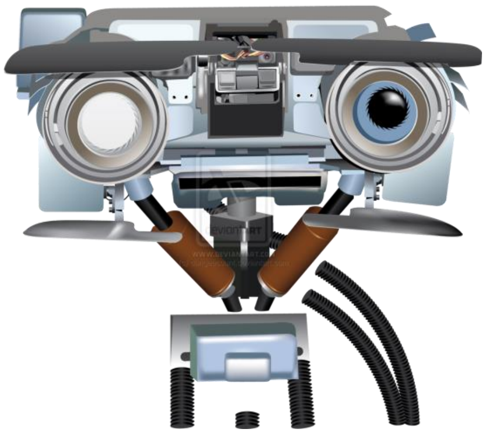
Sometimes, you're running K-means to get clusters to use for some later/downstream purpose. Evaluate K-means based on a metric for how well it performs for that later purpose.

E.g.



Cheaper t-shirts ***vs.*** *Better fitting t-shirts*

slide credit: Andrew Ng



Unsupervised Learning I

Applications of Clustering

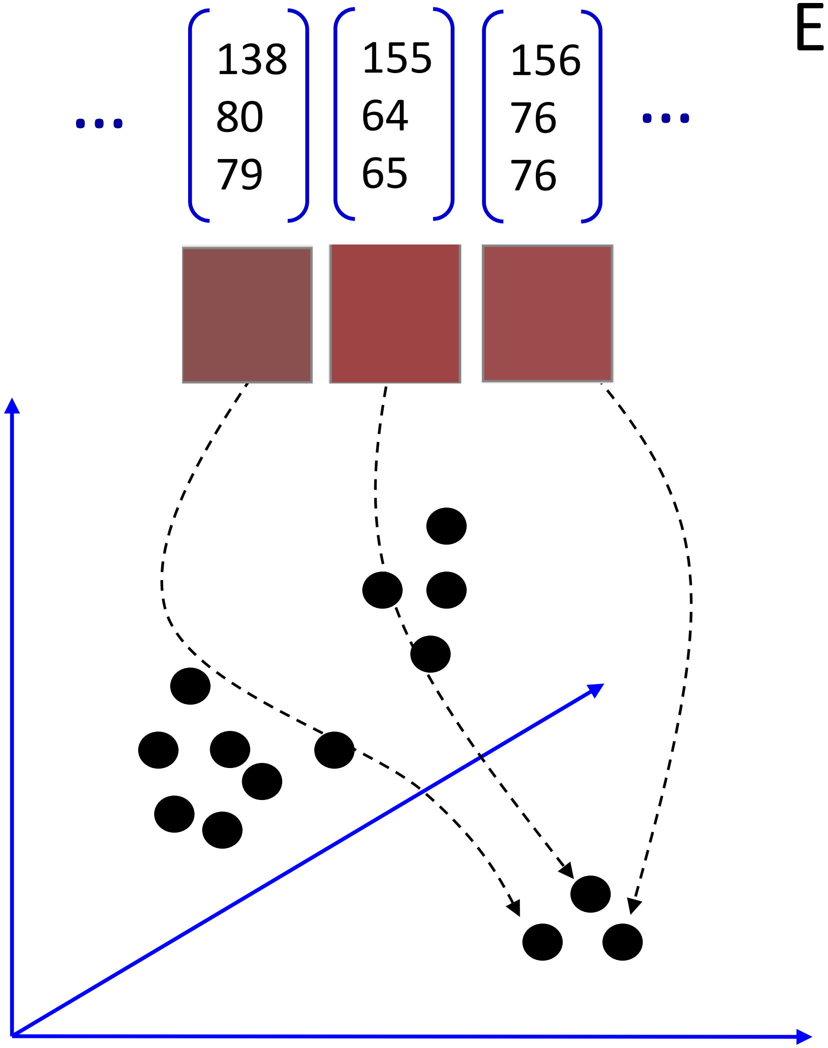
Application of Clustering: Vector Quantization

$$x^{(i)} = \begin{bmatrix} 138 \\ 80 \\ 79 \end{bmatrix} \rightarrow \mu_{c^i}$$

- Compress an image using clustering
- Each {R, G, B} pixel value is an input vector $x^{(i)}$ (255 x 255 x 255 possible values)
- Cluster into K clusters (using k-means)
- Replace each vector by its cluster's index $c^{(i)}$ (K possible values)
- For display, show the mean μ_{c^i}

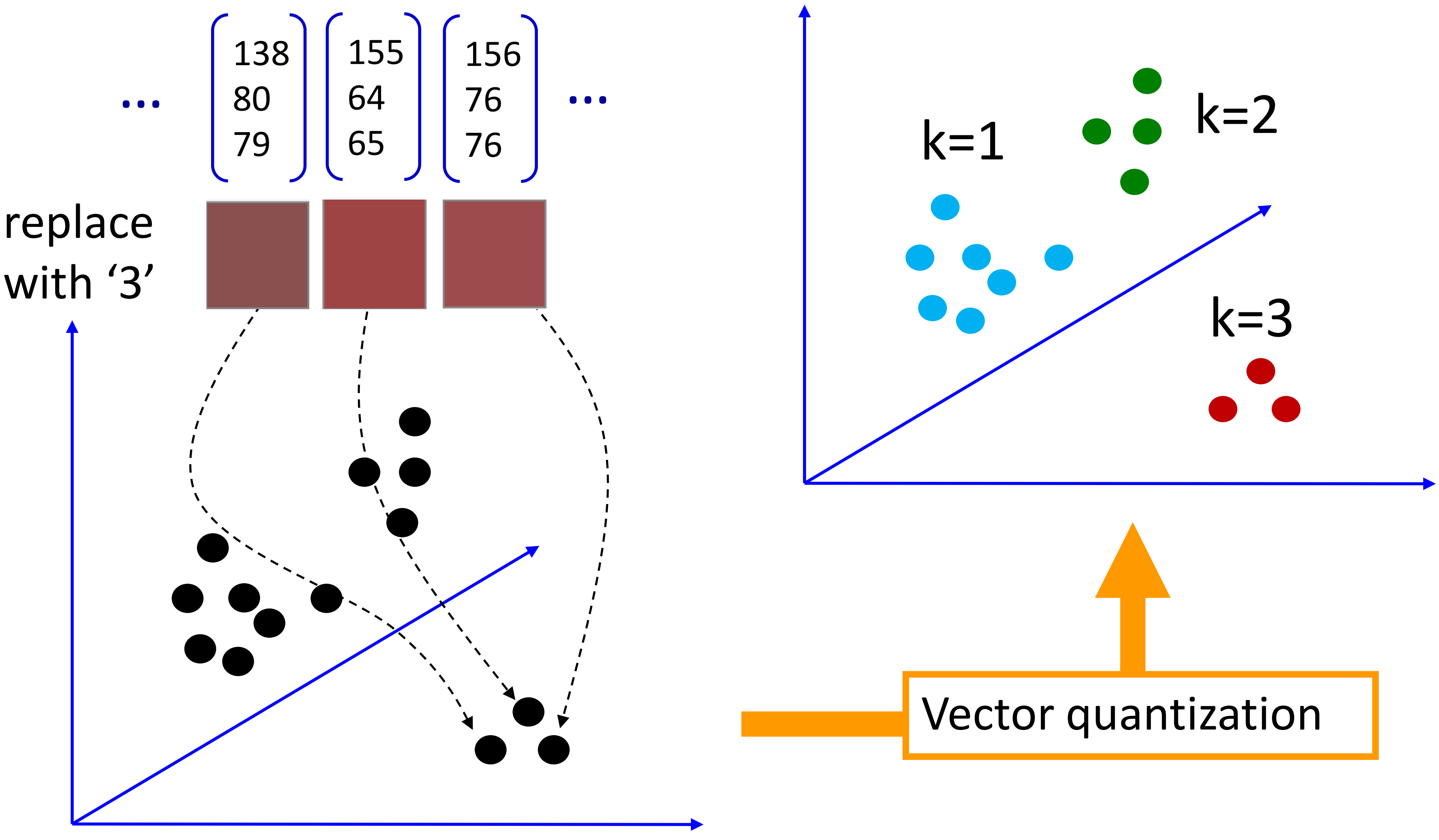
Vector quantization: color values

Example: R, G, B vectors



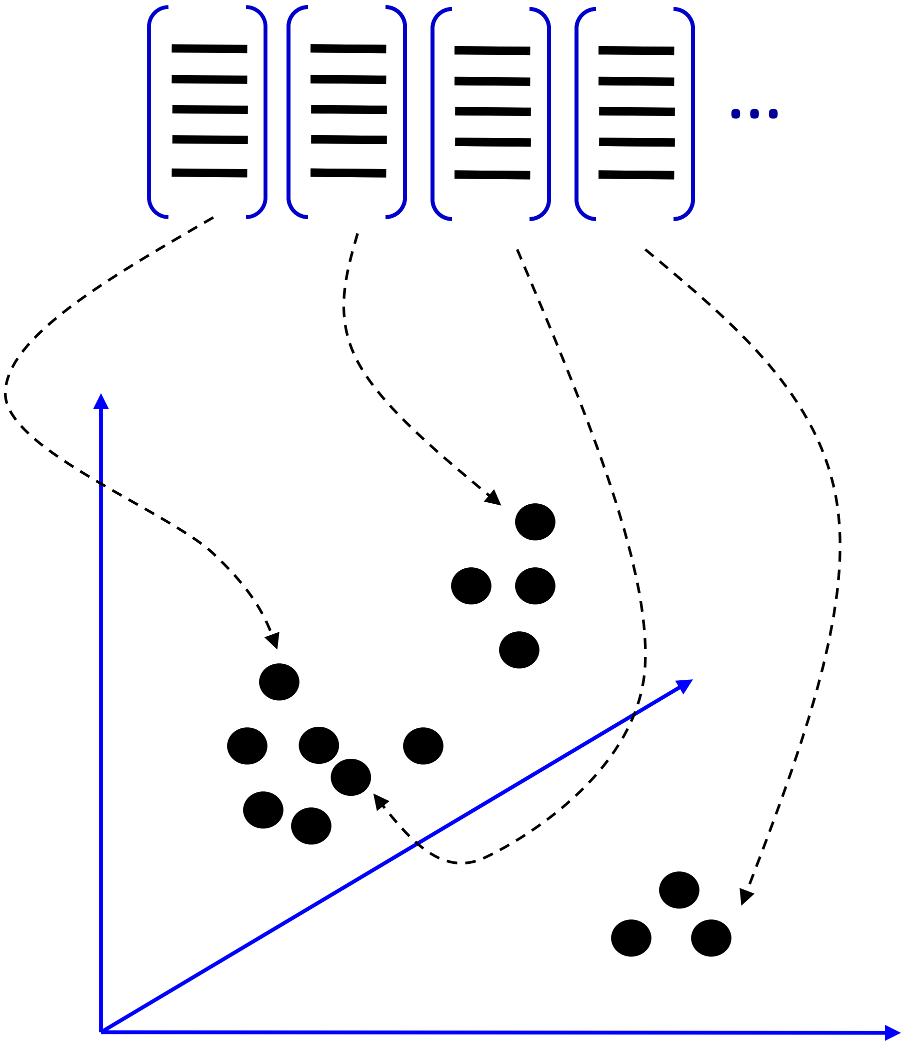
Slide credit: Josef Sivic

Vector quantization: color values



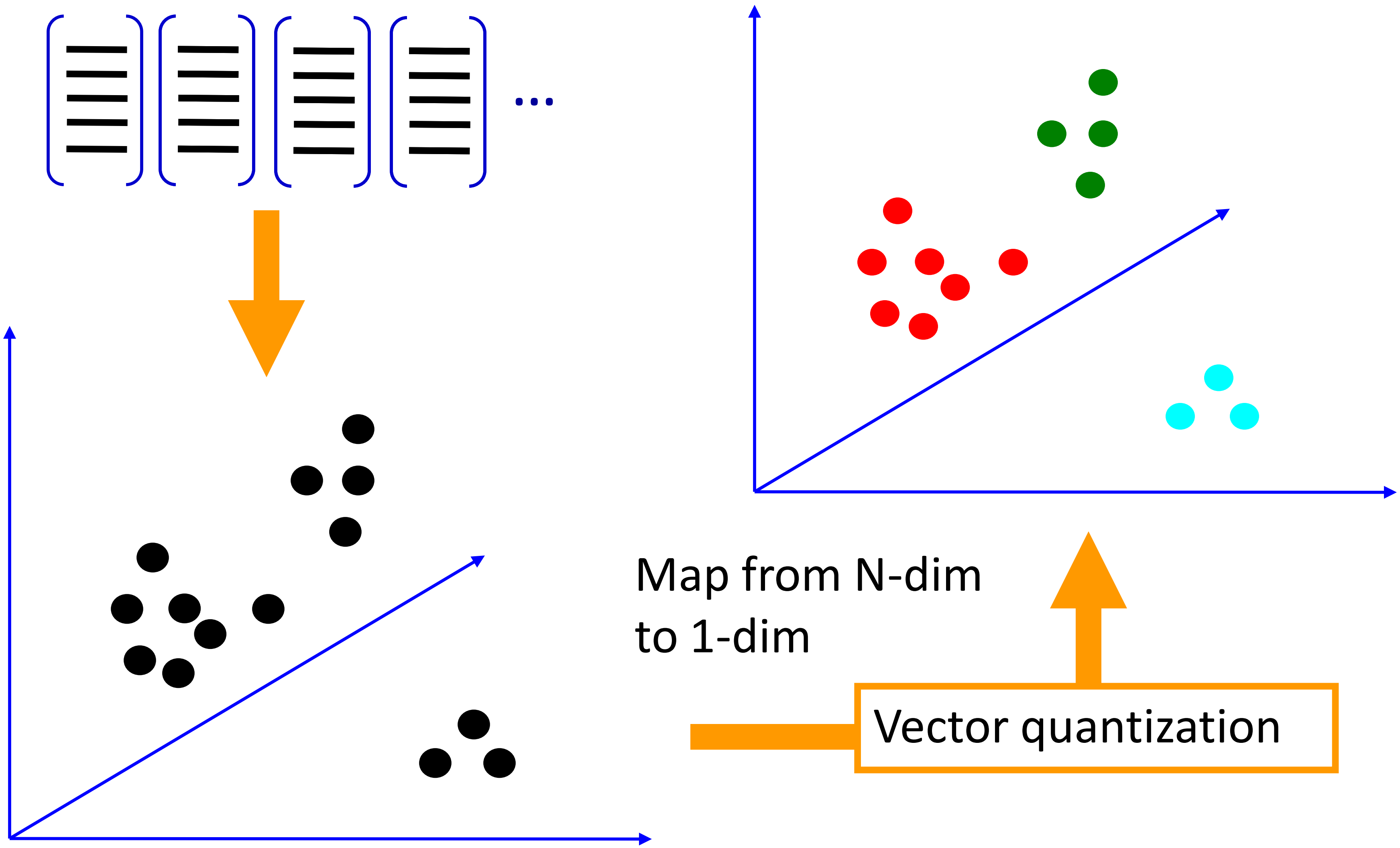
Slide credit: Josef Sivic

Vector quantization: general case



Slide credit: Josef Sivic

Vector quantization: general case



Slide credit: Josef Sivic

K-Means for Image Compression

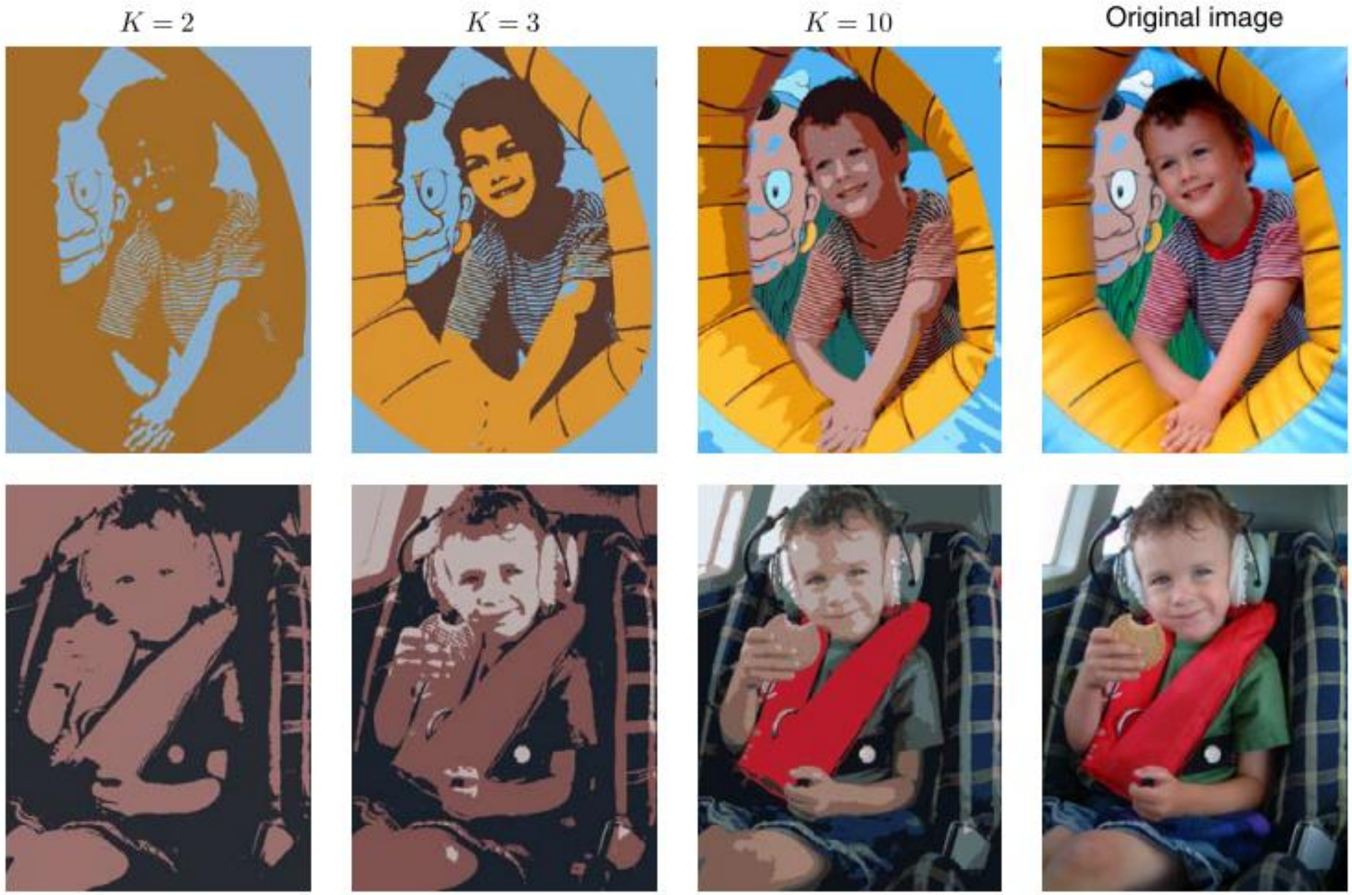
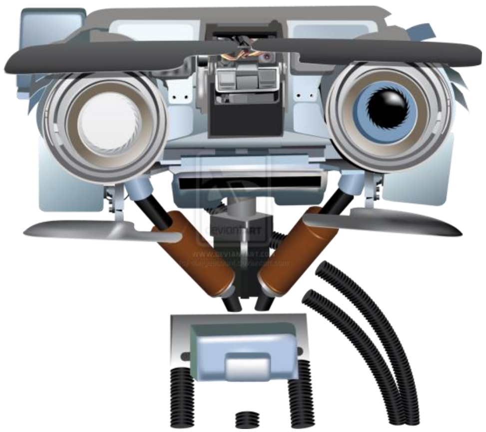


Figure 9.3 Two examples of the application of the K-means clustering algorithm to image segmentation showing the initial images together with their K-means segmentations obtained using various values of K. This also illustrates of the use of vector quantization for data compression, in which smaller values of K give higher compression at the expense of poorer image quality.



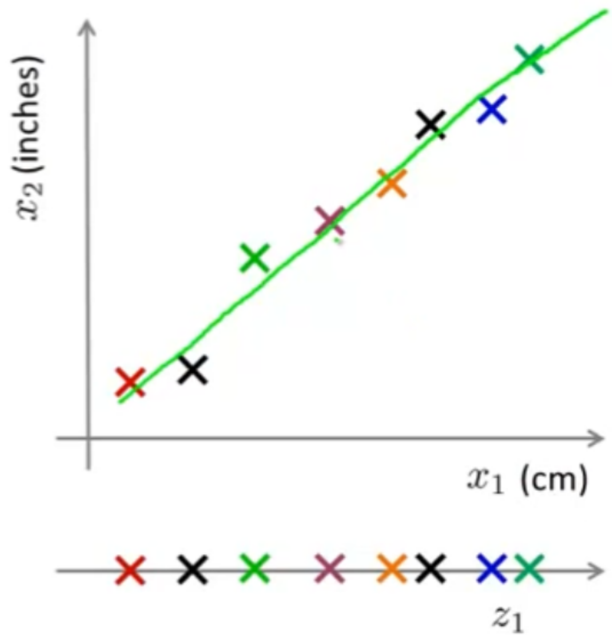
Unsupervised Learning I

Dimensionality Reduction

Data Compression

For each data point, we could reduce the memory requirement for storage and having learning algorithms run faster.

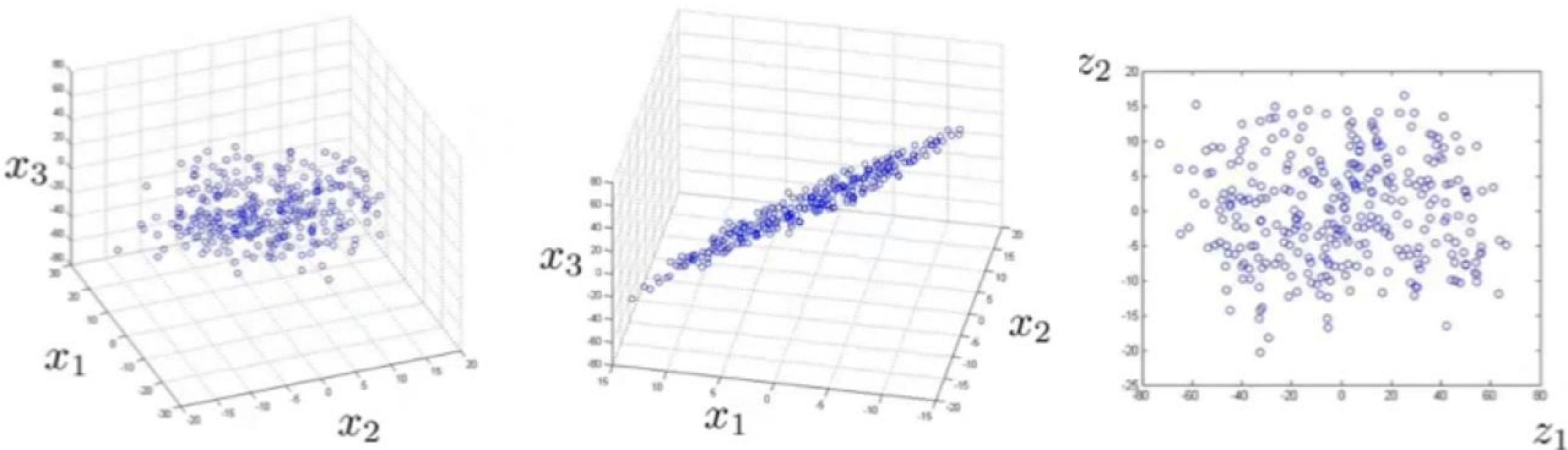
E.g.

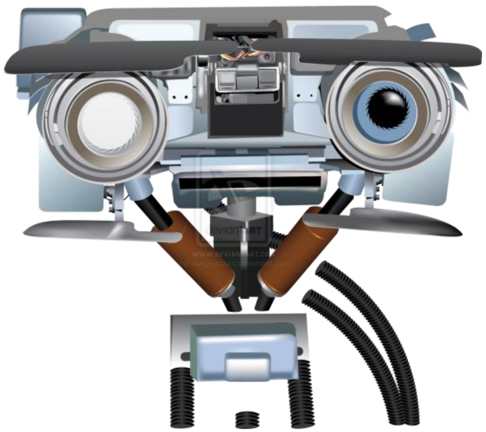


Reduce data from 2D to 1D

Data Compression

Reduce data from 3D to 2D

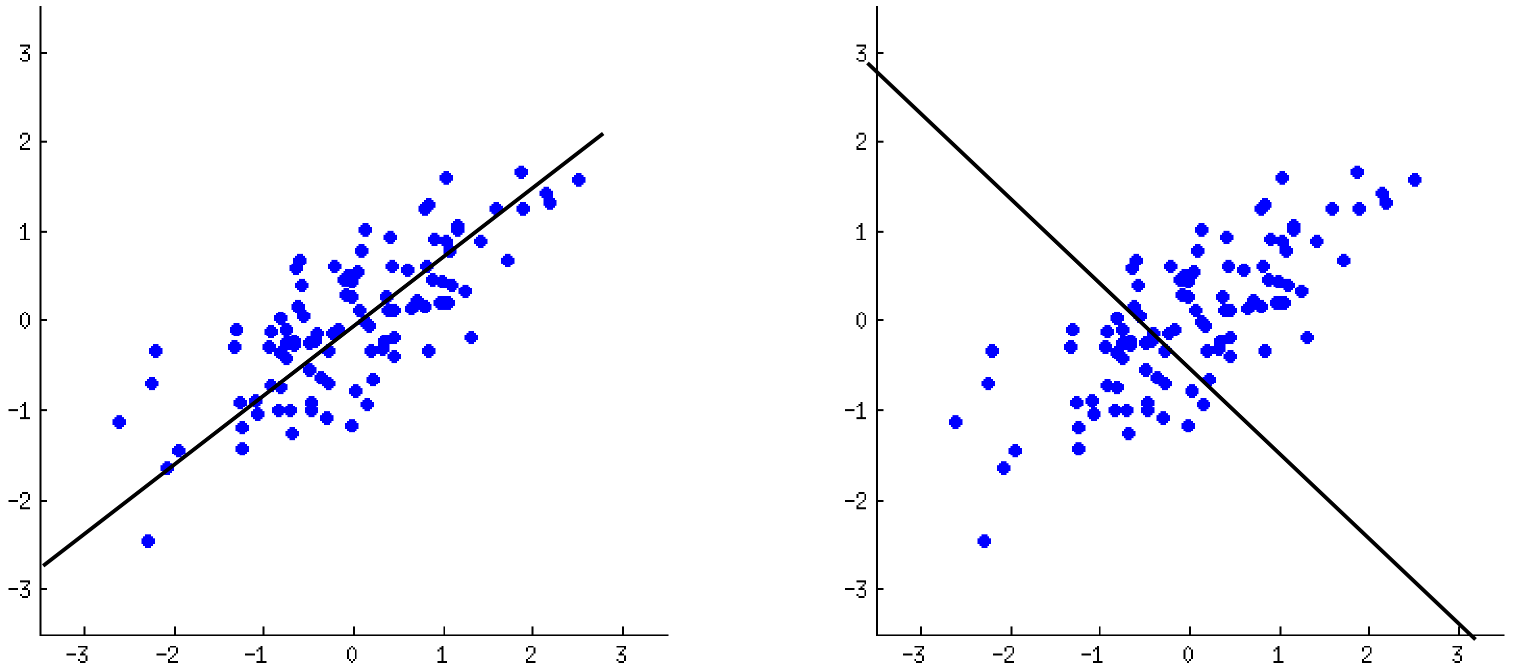




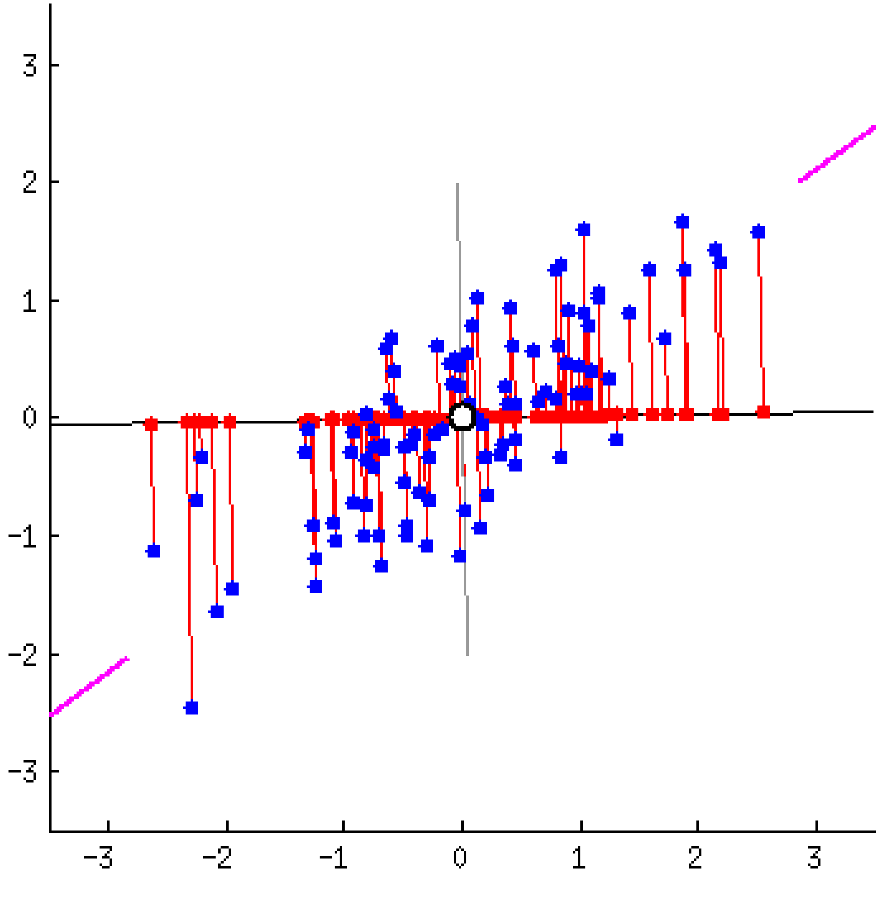
Unsupervised Learning I

Principal Component Analysis

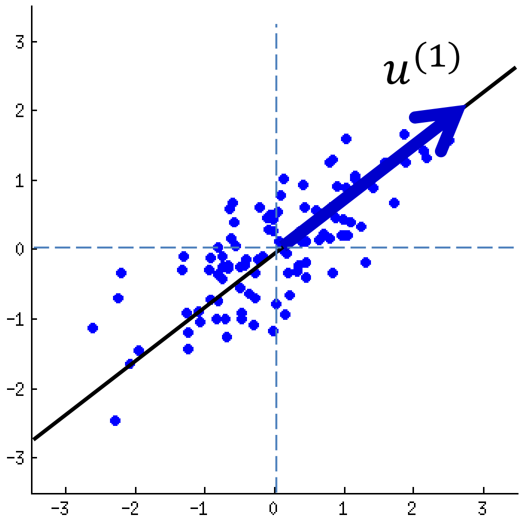
How to choose lower-dim subspace?



Minimize “error”

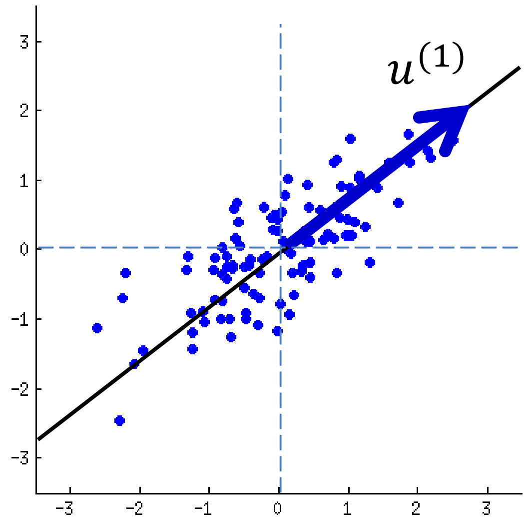


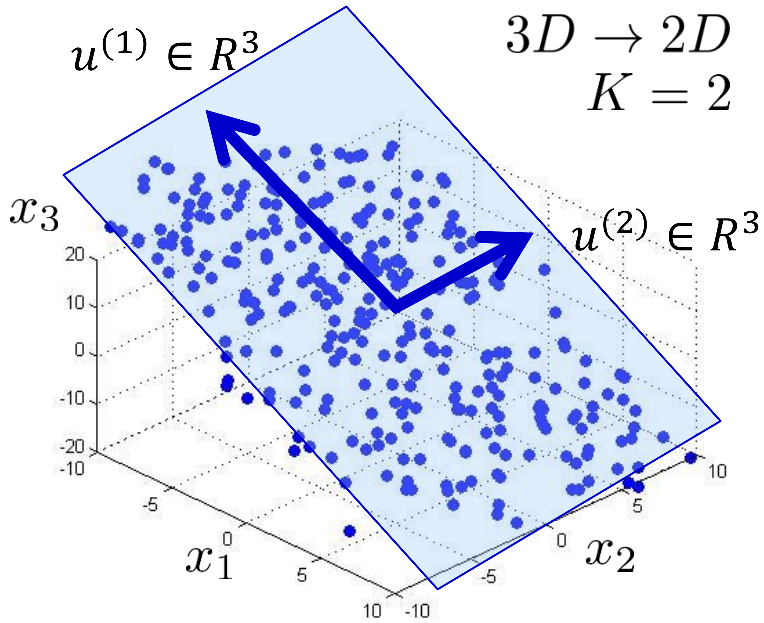
Choose subspace with minimal "information loss"



Reduce from 2-dimension to 1-dimension: Find a direction (a vector $u^{(1)}$) onto which to project the data, so as to minimize the projection error.

Choose subspace with minimal "information loss"



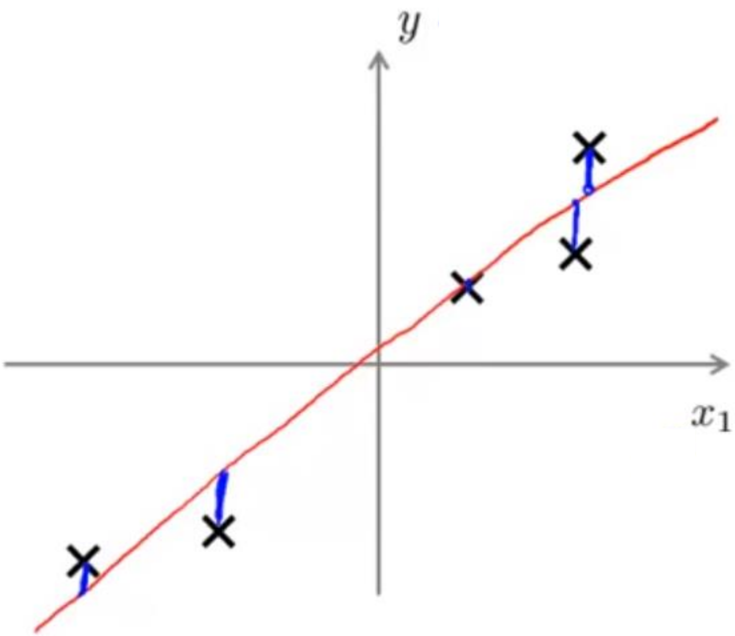


Reduce from 2-dimension to 1-dimension: Find a direction (a vector $u^{(1)}$) onto which to project the data, so as to minimize the projection error.

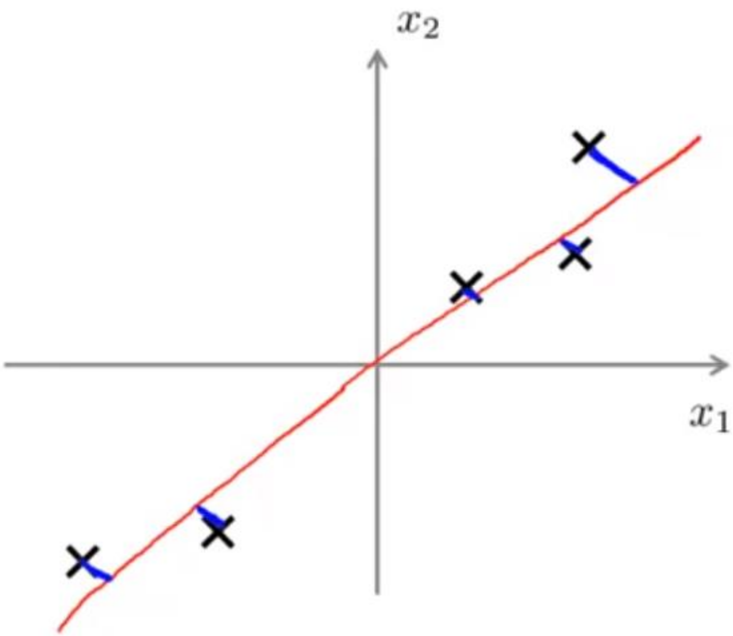
Reduce from n-dimension to K-dimension: Find K vectors $u^{(1)}, u^{(2)}, \ldots, u^{(K)}$ onto which to project the data so as to minimize the projection error.

PCA is not Linear Regression

Linear Regression: predicting y



PCA: no predicted variable



PCA Algorithm

Data preprocessing

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

Preprocessing (feature scaling/mean normalization):

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

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

If different features on different scales (e.g., $x_1 =$ size of house, $x_2 =$ number of bedrooms), scale features to have comparable range of values.

PCA Solution

- The solution turns out to be the first K eigenvectors of the data covariance matrix (see Bishop 12.1 for details)

- Closed-form, use Singular Value Decomposition (SVD) on covariance matrix

PCA Algorithm

Normalize features (ensure every feature has zero mean) and optionally scale feature

Compute "covariance matrix" Σ:

`Sigma =` $$\frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})^T$$

Compute its "eigenvectors":

`[U,S,V] = svd(Sigma);` $$U = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \dots & u^{(n)} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times n}$$

Keep first K eigenvectors and project to get new features **`z`**

```
Ureduce = U(:,1:K);
z = Ureduce'*x;
```

Choosing k (number of principal components)

Average squared projection error: $\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x_{approx}^{(i)}\|^2$
Total variation in the data: $\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2$

Typically, choose k to be smallest value so that

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

“99% of variance is retained”

Good use of PCA

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

- Visualization

Bad use of PCA: To prevent overfitting

Use $z^{(i)}$ instead of $x^{(i)}$ to reduce the number of features to $k < n$.
Thus, fewer features, less likely to overfit.

This might work OK, but isn't a good way to address overfitting. Use regularization instead.

$$\min_{\theta} \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2$$

Summary

- Unsupervised learning
- Discrete latent variables:
  - K-Means clustering
  - Dimensionality Reduction