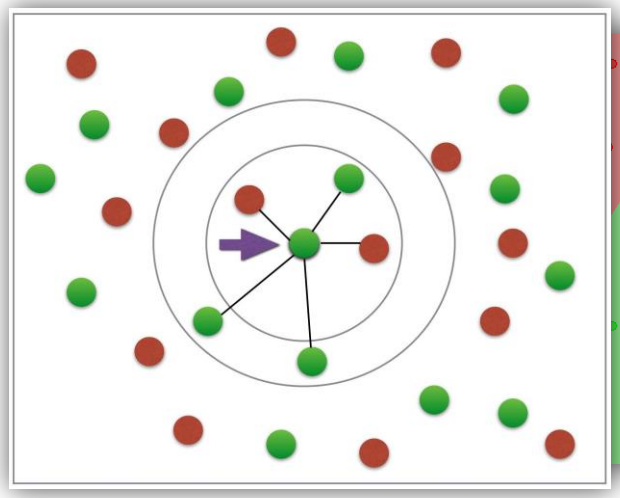
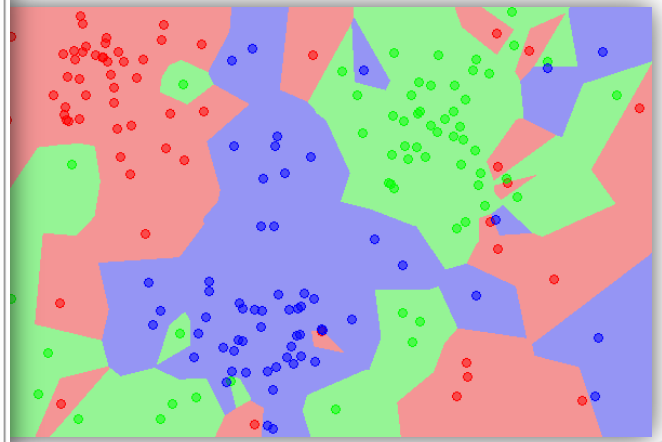
K-NEAREST NEIGHBORS





RECAP

- Residual sum of squares is RSS = $\sum_i^N (y^{(i)} - f(x^{(i)}))^2$, where f is the predictor function trained from the training data. The index i to N of the sum refers to the sum over the data set.
- The MSE is the Residual sum of squares (averaged): MSE = RSS/N.
- The Root Mean Squared Error (RMSE) is the square root of the MSE.
- The MSE of the training set and the MSE of the test set need not be close. If it is far apart, it does not mean that the method is inaccurate necessarily.

INTUITION

- Tell me about your friends(*who your neighbors are*) and *I will tell you who you are*.

OTHER NAMES

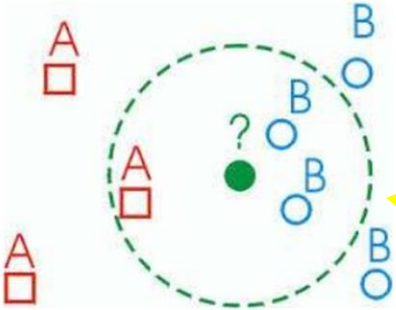
- K-Nearest Neighbors
- Memory-based Reasoning
- Example-based Reasoning
- Instance-based Reasoning
- Lazy Learning

ABOUT KNN

- A powerful classification algorithm used in pattern recognition.
- The algorithm stores all available cases.
- It classifies new cases based on a similarity measure (e.g. distance function)
- It is non-parametric, does not generalize a model.

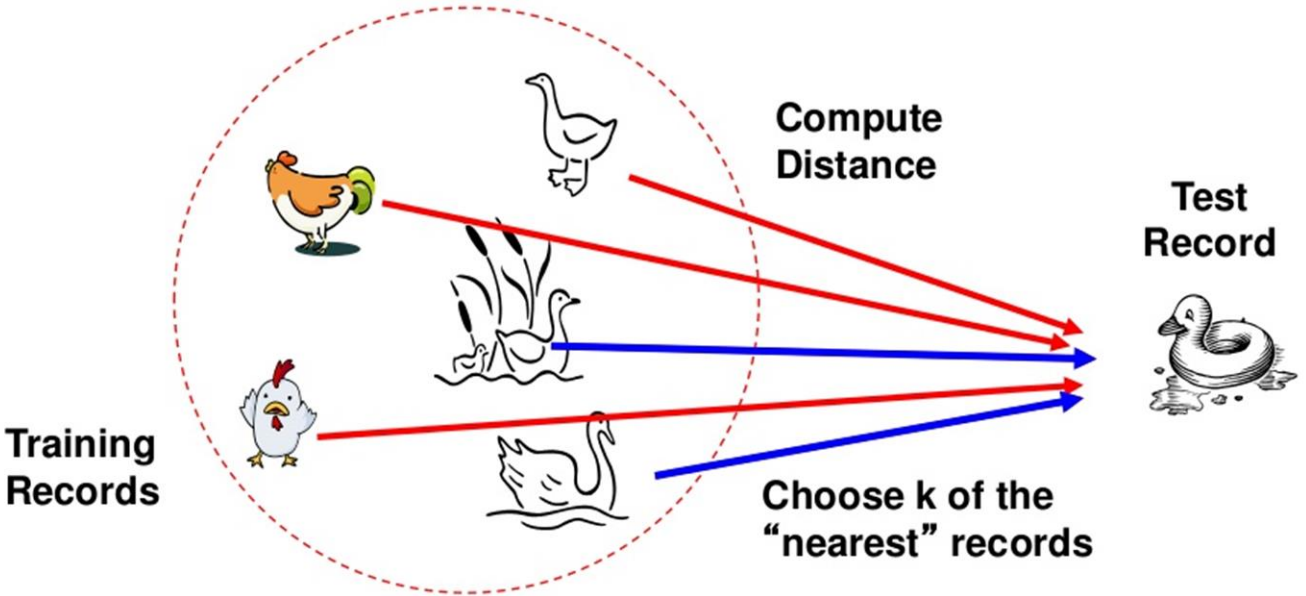
APPROACH

- An object (a new instance) is classified by a majority votes for its neighbor classes.
- The object is assigned to the most common class amongst its K nearest neighbors.(*measured by a distant function*)



The radius matters!

APPROACH


Compute
Distance
Test
Record
Training
Records
Choose k of the
“nearest” records

DISTANCE MEASURE

- Calculate the distance between new example (E) and all examples in the training set.

- *Euclidean* distance between two examples.
  - $X = [x_1, x_2, x_3, .., x_n]$
  - $Y = [y_1, y_2, y_3, \ldots, y_n]$
  - The Euclidean distance between *X* and *Y* is defined as:

$$D(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

ALGORITHM

1. Map all the instances or objects in the training data onto an n-dimensional space (called a feature space).
2. Each instance is represented with a set of numerical attributes.
3. Each instance consists of a vector or a set of vectors, and a class label associated with it.
4. For each test sample, the test sample is classified by comparing the distance between it and the vectors of the K nearest instances in the training set.
5. The test sample is attributed the class label which occur the most often in the set of K nearest instances.

PROS AND CONS

PROS:

- Very simple and intuitive
- Can be applied to data from any distribution
- No model needs to be assumed
- Good classification if training data is large enough
- Does not require the more modern notion of training

CONS:

- Hard to choose K
- Need large training data set to be accurate
- Memory storage required is high since all instances must be recorded, as opposed to a model-based approach where once the model is learnt, the training data is no longer required.

NON PARAMETRIC METHODS

NON-PARAMETRIC METHODS

The main method that we use for probability density estimation is Maximum Likelihood Estimation (MLE).

MLE is known as a parametric method, because we are estimating the parameters in a model. To do this, we have to assume a model first, such as Gaussian or Binomial or Poisson. This model supposedly gives rise to the samples that we see.

If the model chosen is poor, the resulting estimation may be poor.

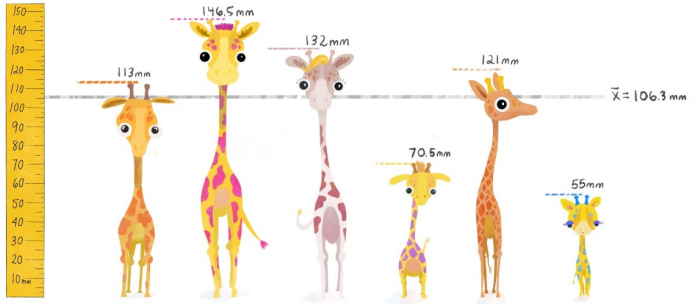
What if we do not have access to a model assumption?

HEIGHT EXAMPLE

The height of different people is usually assumed to be a normal distribution.

In actual fact, this is not true. The heights of a group of people are affected how the people are chosen, what is their background, whether they eat spinach, etc.

Yet height comes from an almost continuous distribution. How do we then estimate the probability density function for the heights of a group of people?



HISTROGRAM METHOD

This method creates a discrete probability function that estimates the sample set $\{x_1, \ldots, x_N\}$.

In the one-dimensional setting (state space is scalar), split the state space (usually the entire real set) into intervals of size $\triangle_i$.

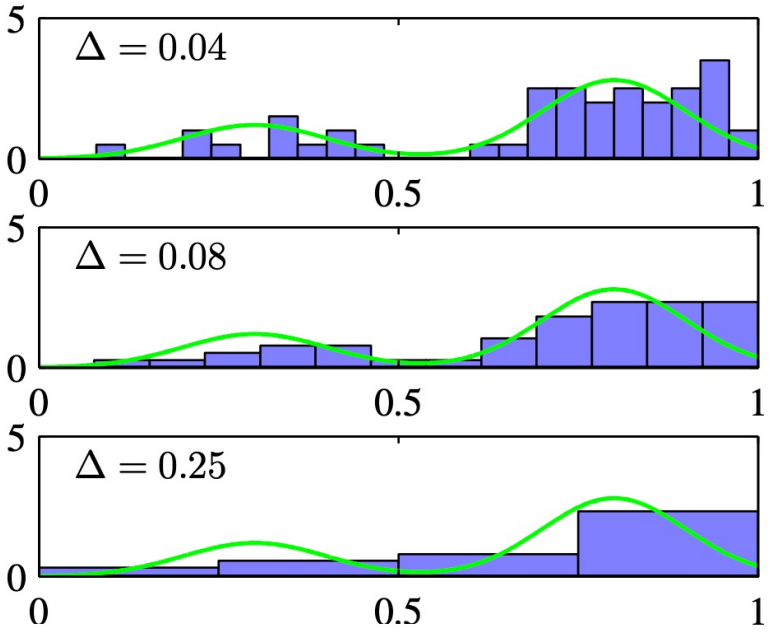
Count the number n_i of sample points in each interval.

We define:

$$p_i = \frac{n_i}{N\Delta_i}$$

HISTROGRAM METHOD

An illustration of the histogram approach to density estimation, in which a data set of 50 data points is generated from the distribution shown by the green curve. Histogram density estimates, based on (2.241), with a common bin width Δ are shown for various values of Δ.



HISTROGRAM METHOD

Pros:

- Doesn’t require a model assumption.
- Straightforward to compute.

Cons:

- Hard to scale with respect to dimensions (multivariate setting).
- Difficult to convert to a continuous probability function.
- This distribution doesn’t produce samples of the same state space
- Hard to pick interval size depending on how spaced out the samples are.

KERNEL DENSITY ESTIMATORS

Sometimes, we might need a continuous probability function as an estimate for the underlying distribution. To do this, we will use kernel density estimators.

We will properly define kernels in the next lecture. For now we will just describe the function that is commonly used for kernel density estimators.

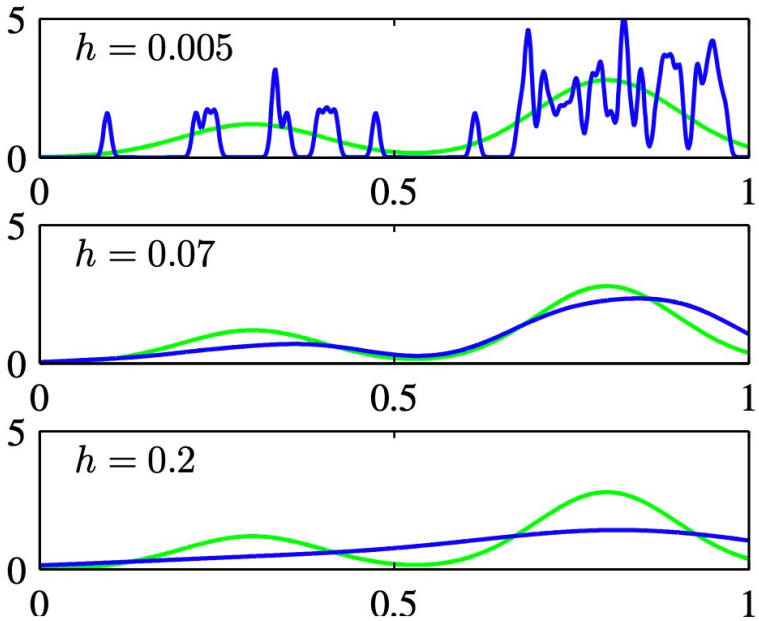
KERNEL DENSITY ESTIMATORS

Kernel density estimators assume that there for some fixed h, there is a Gaussian distribution of standard deviation h over each data point. The Gaussian distribution function here is the kernel chosen. The proposed estimator is the sum of these Gaussians.

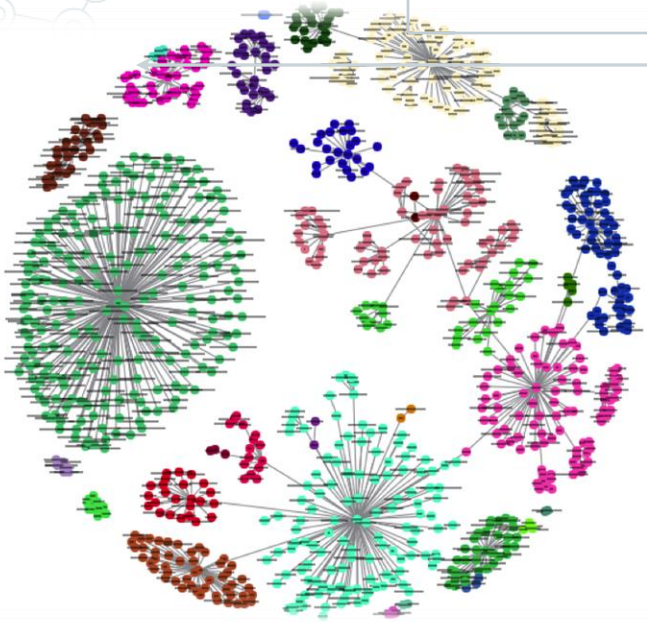
$$p(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{(2\pi h^2)^{1/2}}\exp\left\{-\frac{\|\mathbf{x}-\mathbf{x}_n\|^2}{2h^2}\right\}$$

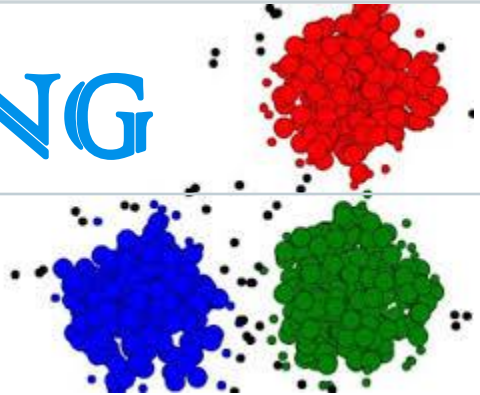
KERNEL DENSITY ESTIMATORS

Illustration of the kernel density model (2.250) applied to the same data set used to demonstrate the histogram approach in Figure 2.24. We see that h acts as a smoothing parameter and that if it is set too small (top panel), the result is a very noisy density model, whereas if it is set too large (bottom panel), then the bimodal nature of the underlying distribution from which the data is generated (shown by the green curve) is washed out. The best density model is obtained for some intermediate value of h (middle panel).



CLUSTERING



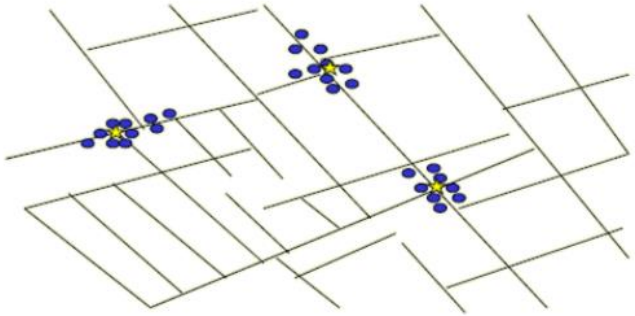


HISTORY APPLICATION

London physician John Snow plotted the locations of cholera deaths during an outbreak in the 1850s, in order to better understand the root causes of the outbreak.

It turned out that many of these cases were clustered around certain road intersections where there were polluted wells.

Through clustering, everything was well again.



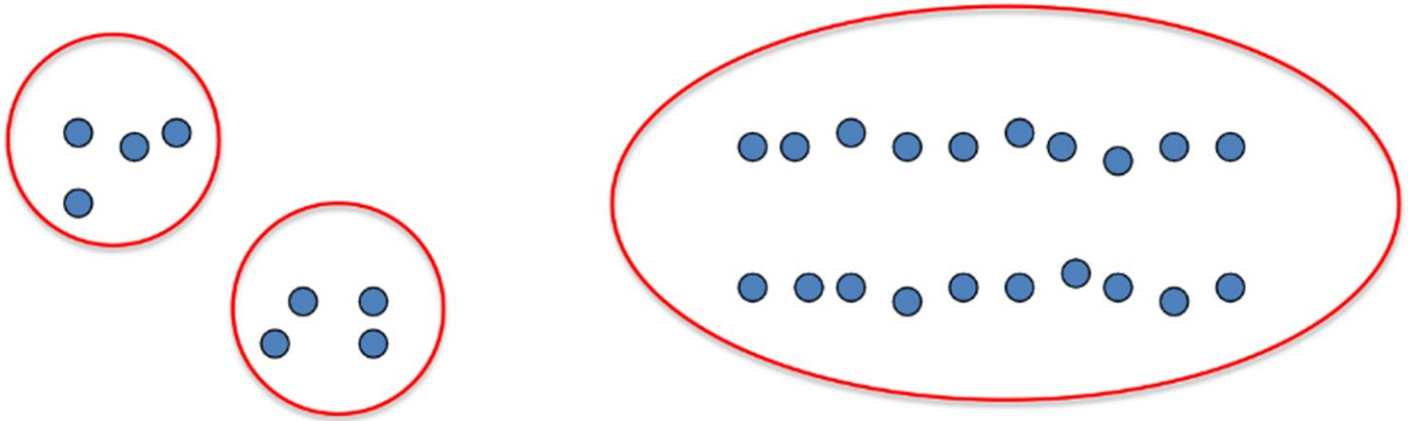
From: Nina Mishra HP Labs

UNSUPERVISED LEARNING

- Unsupervised learning
- Requires data, but no labels
- Detect patterns e.g. in
  - Group emails or search results
  - Customer shopping patterns
  - Regions of images
- Useful when don't know what you're looking for
- But: can get gibberish

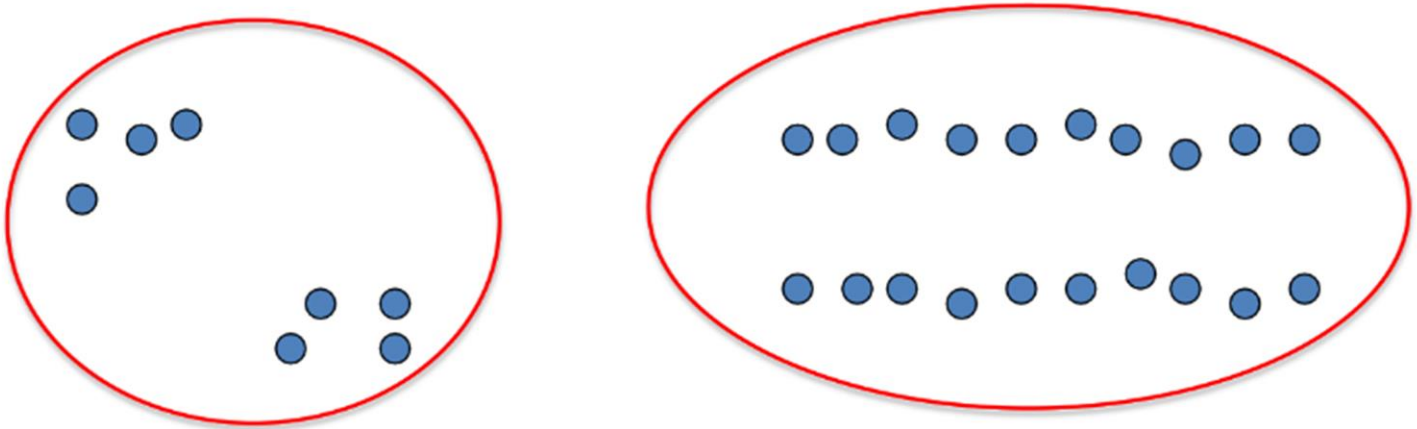
INTUITION

- Basic idea: group together similar instances
- Example: 2D point patterns



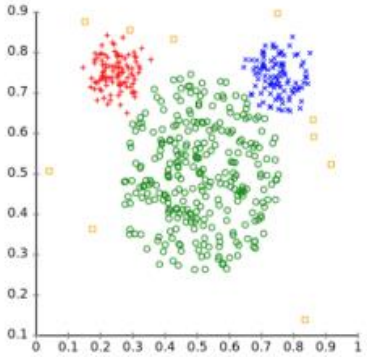
INTUITION

- Basic idea: group together similar instances
- Example: 2D point patterns



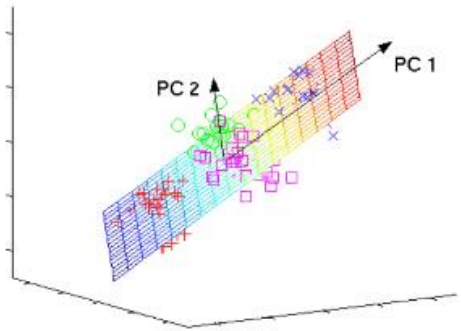
CLUSTERING

- No labels/responses. Finding structure in data.
- Dimensionality Reduction.



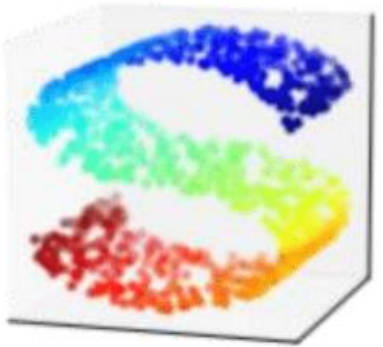
Clustering

$T: \mathbb{R}^d \rightarrow \{1, 2, \ldots, k\}$



Subspace Learning

$T: \mathbb{R}^d \rightarrow \mathbb{R}^m$



Manifold Learning

CLUSTERING APPLICATION

- Improve classification/regression (semi-supervised learning)

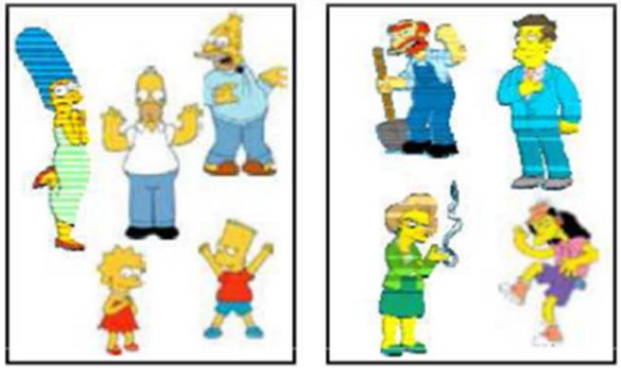
1. From *unlabeled data*, learn a good features $T: \mathbb{R}^d \rightarrow \mathbb{R}^m$.
2. To *labeled data*, apply transformation $T: \mathbb{R}^d \rightarrow \mathbb{R}^m$.

$$\left(T\left(x^{(1)}\right), y^{(1)}\right), \ldots, \left(T\left(x^{(n)}\right), y^{(n)}\right)$$

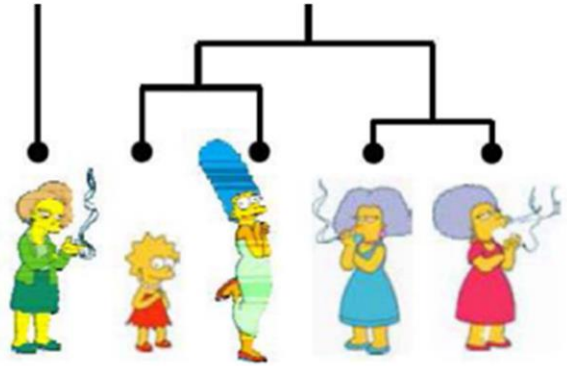
3. Perform classification/regression on transformed data.

TYPES OF CLUSTERING

- Partition algorithms (Flat)
  - K-means
  - Mixture of Gaussian
  - Spectral Clustering



- Hierarchical algorithms
  - Bottom up – agglomerative
  - Top down – divisive



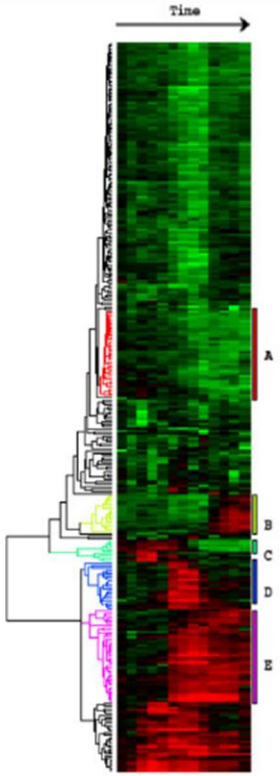
EXAMPLES

Image segmentation

Goal: Break up the image into meaningful or perceptually similar regions

EXAMPLES

Clustering gene expression data

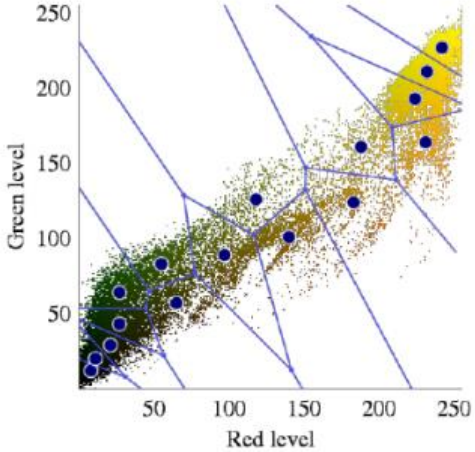


Eisen et al, PNAS 1998

EXAMPLES

- Data compression

RGB
(12,165,73)
(11,167,79)
:

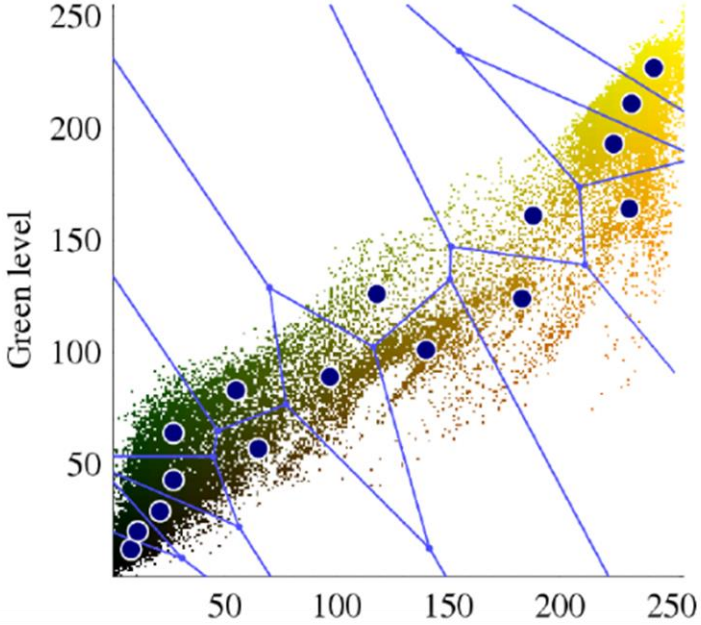


Labels
3
43
:

Dictionary
1 ~ (10, 160, 70)
2 ~ (40, 240, 20)
:

VORONOI DIAGRAM

We can partition all the points in the space into regions, according to their closest representative.

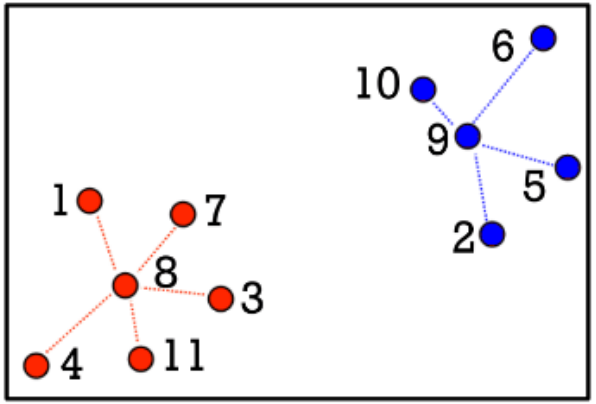


DEFINING A CLUSTER

- By listing all its elements

$$\mathcal{C}_1 = \{1,3,4,7,8,11\}$$

$$\mathcal{C}_2 = \{2,5,6,9,10\}$$

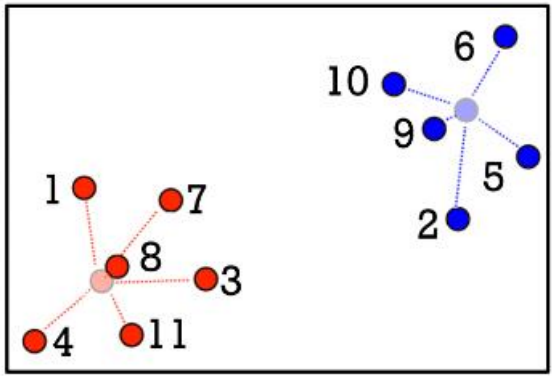


DEFINING A CLUSTER

Each point $x^{(i)}$ will be assigned the closest representative.

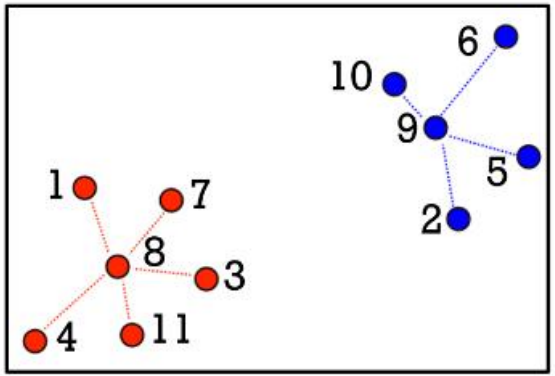
- Using a representative
  a. A point in center of cluster (centroid)
  b. A point in the training data (exemplar)

$$z^{(1)} = \begin{pmatrix}1\\1\end{pmatrix}, z^{(2)} = \begin{pmatrix}5\\4\end{pmatrix}$$



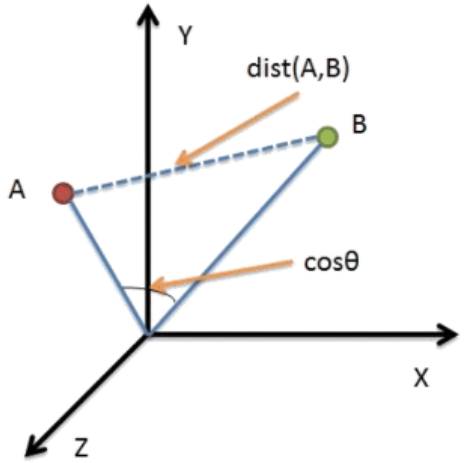
centroid

$$z^{(1)} = 8, z^{(2)} = 9$$



exemplar

DISTANCE METRICS

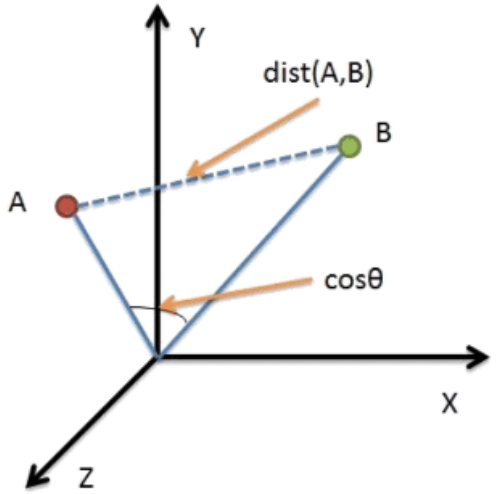


(sometimes called *loss functions*)

A measure of how close two data points are. Nearby points (i.e. distance is *small*) are more likely they belong to the same cluster.

- Euclidean Distance $\text{dist}(x, y) = \|x - y\|^2$

SIMILARITY METRICS



(sometimes called *kernels, correlation*)

A measure of how alike two data points are. Similar points (i.e. similarity is *large*) are more likely they belong to the same cluster.

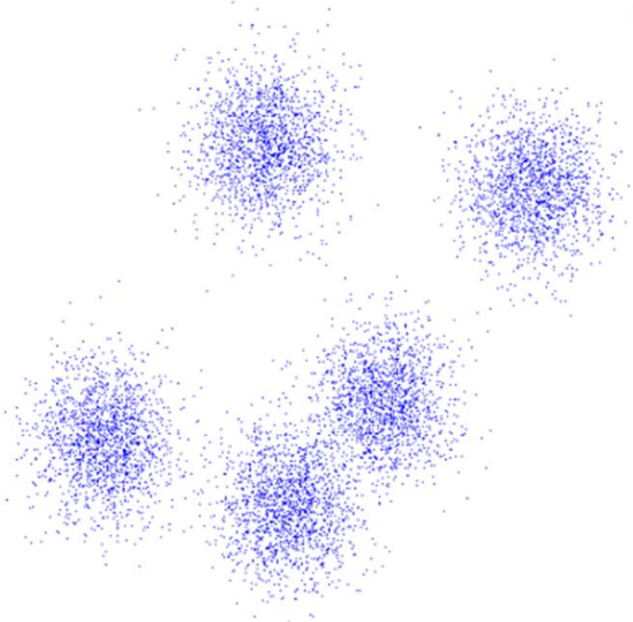
- Cosine Similarity $\cos(x, y) = \frac{x^T y}{\|x\| \, \|y\|}$

K-MEANS

- An iterative clustering algorithm
  - Initialize: Pick *K* random points as cluster centers
  - Alternate:
    1. Assign data points to closest cluster center
    2. Change the cluster center to the average of its assigned points
  - Stop when no points' assignments change

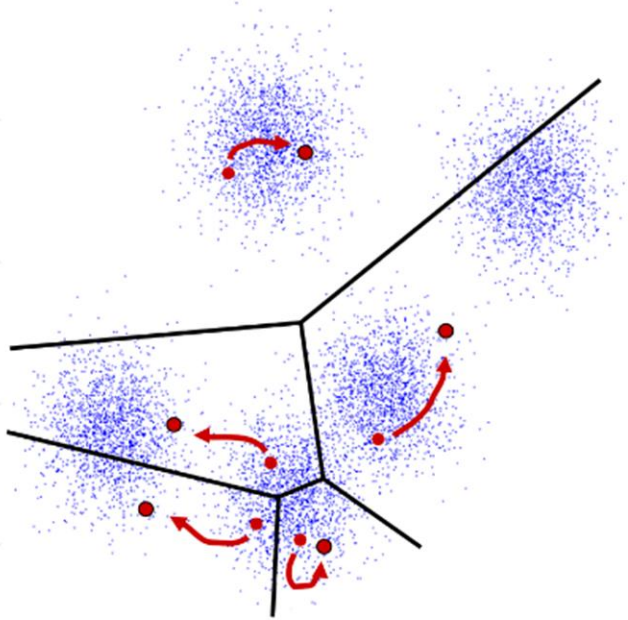
Voronoi!

Stability reached.

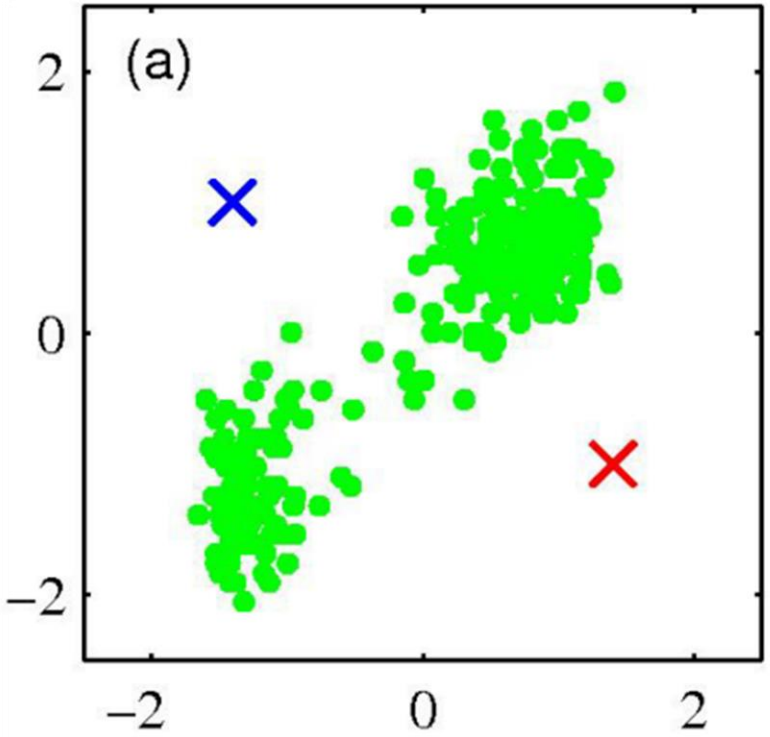


K-MEANS

- An iterative clustering algorithm
  - Initialize: Pick *K* random points as cluster centers
  - Alternate:
    1. Assign data points to closest cluster center
    2. Change the cluster center to the average of its assigned points
  - Stop when no points' assignments change



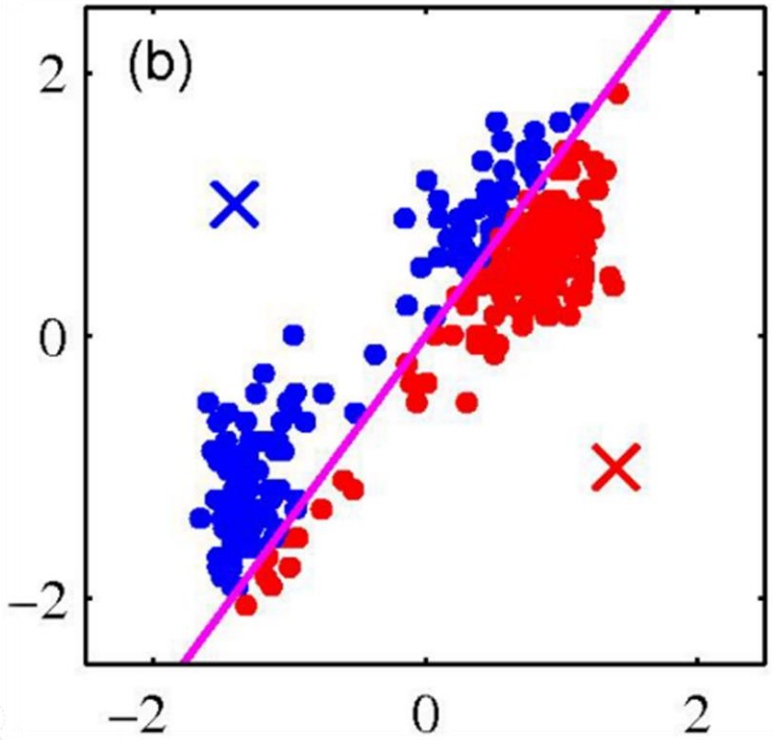
EXAMPLE



- Pick *K* random points as cluster centers (means)

Shown here for *K*=2

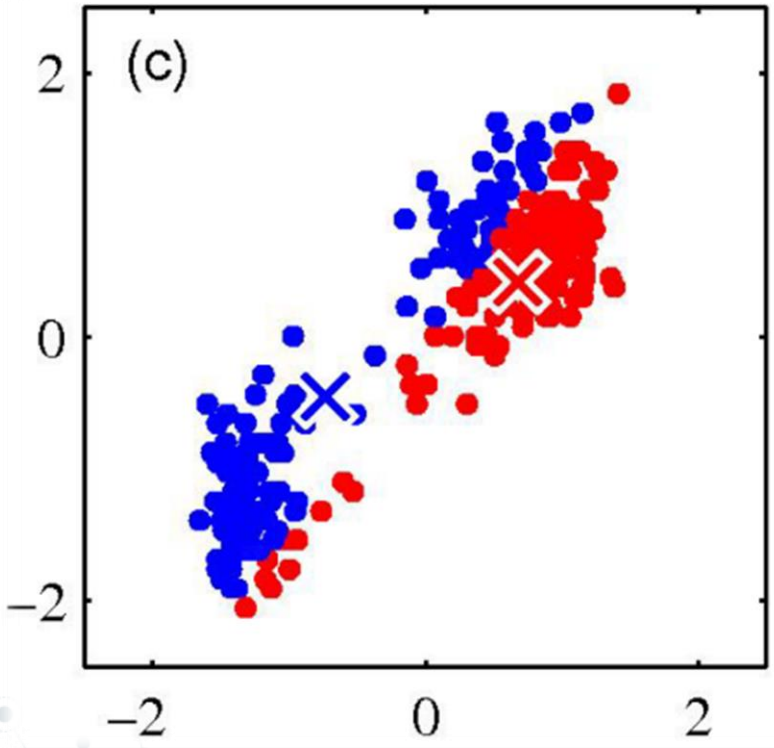
EXAMPLE



Iterative Step 1

- Assign data points to closest cluster center

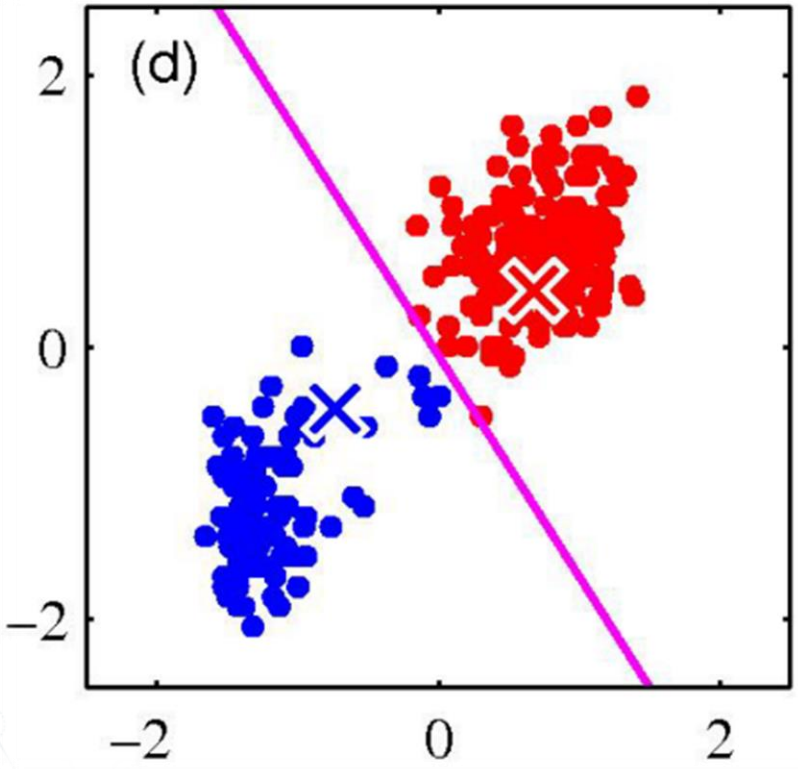
EXAMPLE



Iterative Step 2

- Change the cluster center to the average of the assigned points

EXAMPLE



- Repeat until convergence

LOSS FUNCTION

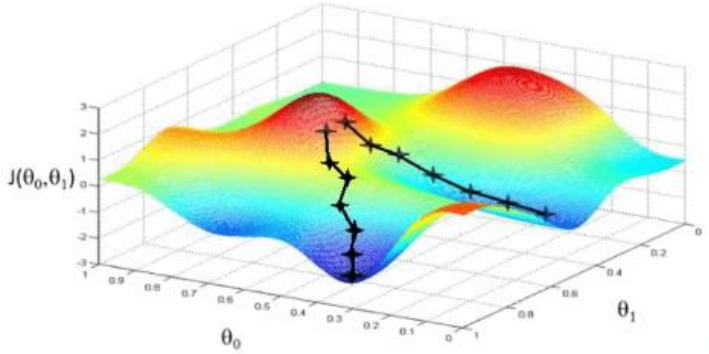
Optimizing over clusters.

$$\mathcal{L}_{n,k}(\mathcal{C}_1, \ldots, \mathcal{C}_n; \mathcal{S}_n) = \sum_{j=1}^{n} \sum_{i \in \mathcal{C}_j} \left\| x^{(i)} - \frac{1}{|\mathcal{C}_j|} \sum_{i' \in \mathcal{C}_j} x^{(i')} \right\|^2 .$$

The norm ||x|| is the euclidean distance.
C_1, …, C_n are the clusters, S_n is the data set, n is the number of clusters here.
Here centroids are used, that is the average of all the data points in the clusters form the centroid.

CONVERGENCE

- Training loss always decreases in each step (coordinate descent).
- Converges to local minimum, not necessarily global minimum.



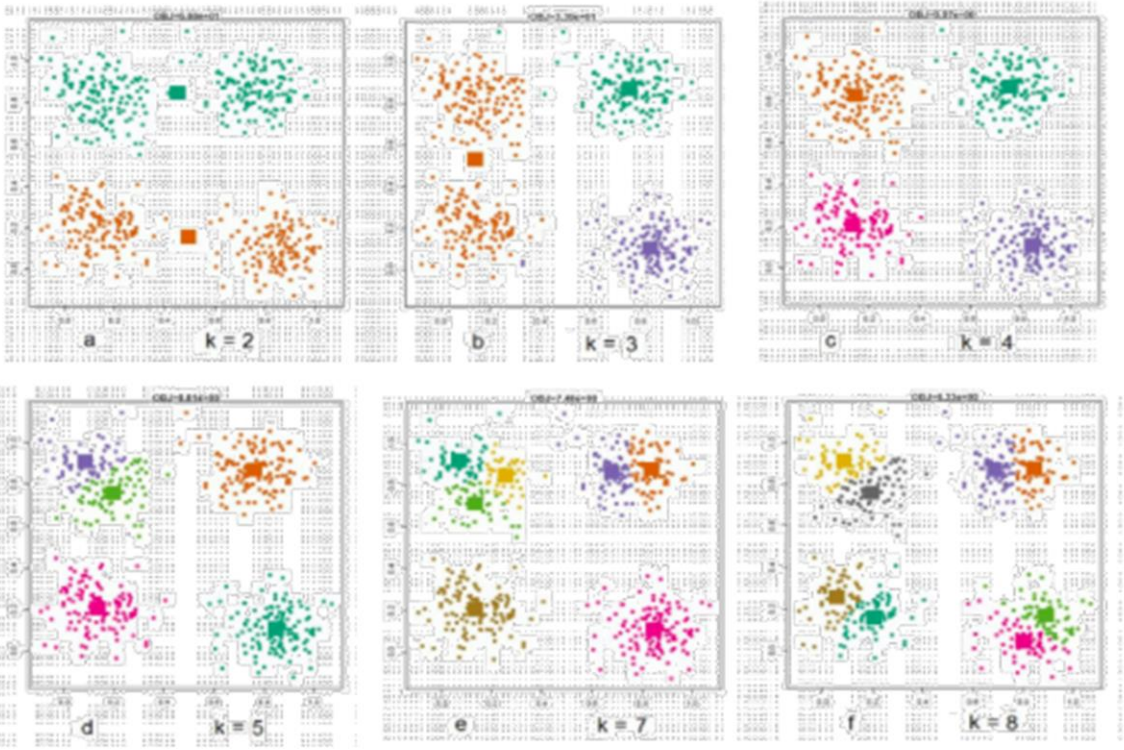
Challenge.
Why does the algorithm terminate in a finite number of steps?
* not in syllabus

Repeat algorithm over many initial points, and pick the configuration with the smallest training loss.

INITIALIZATION

Problem. How to choose good starting positions?
Solution. Place them far apart with high probability.

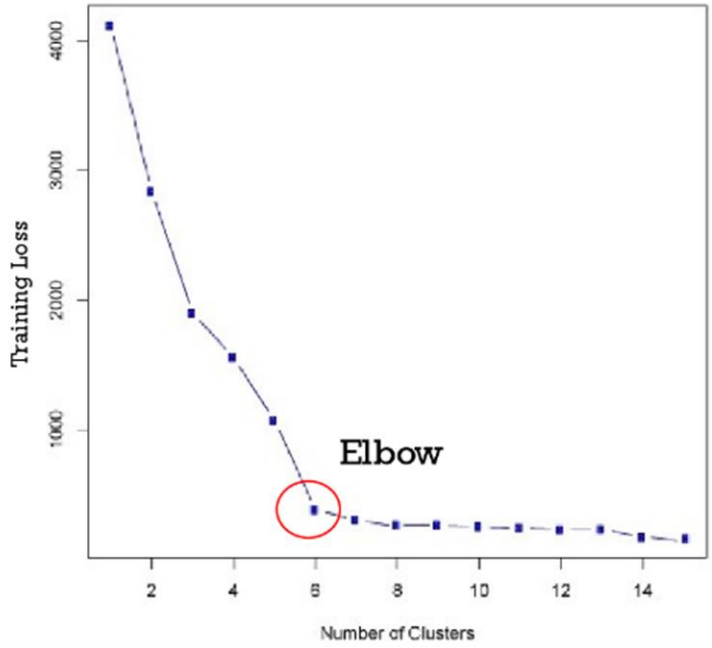
NUMBER OF CLUSTERS



NUMBER OF CLUSTERS

How do we choose k, the optimal number of clusters?

- Elbow method
  - Training Loss



K-MEDOIDS

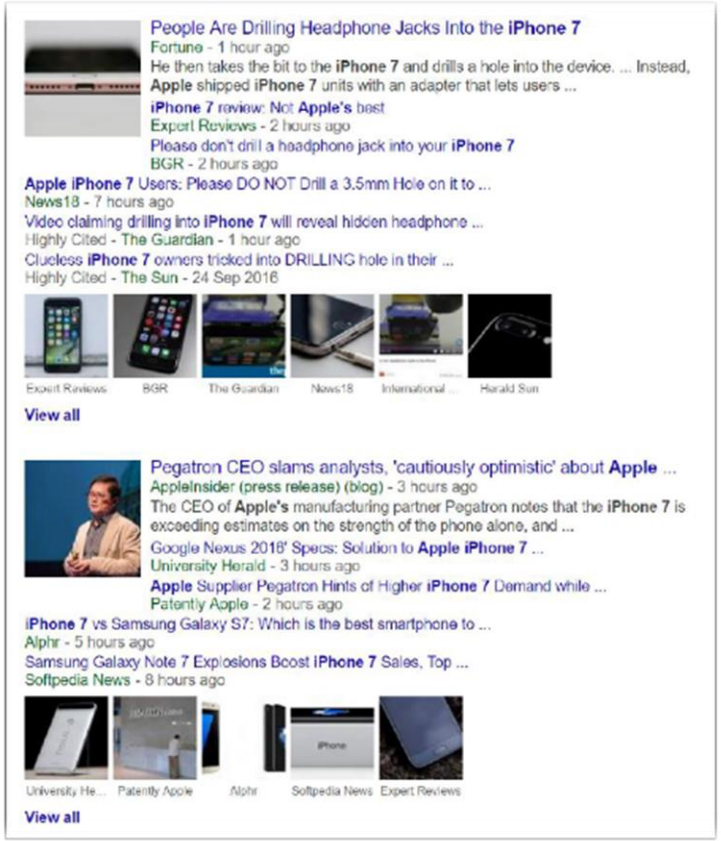
K-MEDOIDS

Use exemplars
instead of centroids.

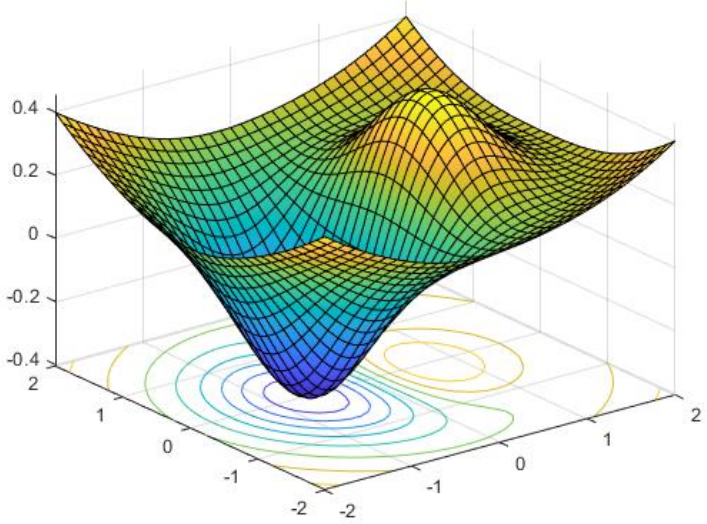
e.g. Google News.

Repeat until convergence:

- Find best clusters given exemplars
- Find best exemplars given clusters



OPTIMIZATION



GOAL

Want to minimize some function $f(x)$, but
there are some *constraints* on the values of x.

Method 1 (Dual Problem)

Solve a *dual optimization problem* where the constraints are nicer,
and where it is easier to implement gradient descent.

Method 2 (Exact Solution)

Solve the *Lagrangian* system of equations.

EQUALITY CONSTRAINTS

Problem.

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & h_1(x) = 0, \ldots, h_l(x) = 0 \end{aligned}$$

Lagrangian.

$$L(x, \lambda) = f(x) + \lambda_1 h_1(x) + \cdots + \lambda_l h_l(x)$$

Example.

$$\begin{aligned} \text{minimize} \quad & f(x) = n_1 \log x_1 + \cdots + n_d \log x_d \\ \text{subject to} \quad & h(x) = x_1 + \cdots + x_d - 1 = 0 \end{aligned}$$

$$L(x, \lambda) = n_1 \log x_1 + \cdots + n_d \log x_d + \lambda(x_1 + \cdots + x_d - 1)$$

TWO PLAYER GAME

$$L(x, \lambda) = f(x) + \lambda_1 h_1(x) + \cdots + \lambda_l h_l(x)$$

Rules.

- You get to choose the value of x.
  Your goal is to minimize $L(x, \lambda)$.
- Your adversary gets to choose the value of λ.
  His goal is to maximize $L(x, \lambda)$.

PRIMAL GAME

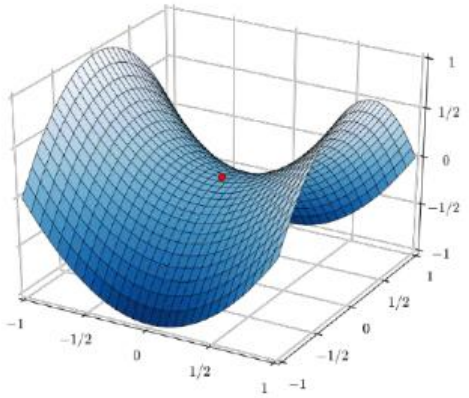
$$L(x, \lambda) = f(x) + \lambda_1 h_1(x) + \cdots + \lambda_l h_l(x)$$

Primal Game. You go first.

Your Strategy.

- Ensure that $h_1(x) = 0, \ldots, h_l(x) = 0$.
- Find x that minimizes $f(x)$.

Final Score. $p^* = \min_x \max_\lambda L(x, \lambda)$



DUAL GAME

$$L(x, \lambda) = f(x) + \lambda_1 h_1(x) + \cdots + \lambda_l h_l(x)$$

Dual Game. You go second.

Adversary's Strategy.

- For each λ, compute $\ell(\lambda) = \min_x L(x, \lambda)$
- Find λ that maximizes $\ell(\lambda)$.

Final Score. $d^* = \max_\lambda \min_x L(x, \lambda)$

MAX-MIN INEQUALITY

Primal. $p^* = \min_x \max_\lambda L(x, \lambda)$

Dual. $d^* = \max_\lambda \min_x L(x, \lambda)$

$$p^* = \min_x \max_\lambda L(x, \lambda) \geq \max_\lambda \min_x L(x, \lambda) = d^*$$

If $p^* = d^*$, we can solve the primal by solving the dual.

MAX-MIN INEQUALITY

Example.

	$x = 1$	$x = 2$
$\lambda = 1$	1	4
$\lambda = 2$	3	2

Primal. $p^* = \min_x \max_\lambda L(x, \lambda) = 3$

Dual. $d^* = \max_\lambda \min_x L(x, \lambda) = 2$

EXACT SOLUTION (EQUALITY)

Problem.

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && h_1(x) = 0, \ldots, h_l(x) = 0 \end{aligned}$$

Lagrange multipliers.

1. Write down the Lagrangian.

$$L(x, \lambda) = f(x) + \lambda_1 h_1(x) + \cdots + \lambda_l h_l(x)$$

2. Solve for critical points x, λ.

$$\nabla_x L(x, \lambda) = 0, \quad h_1(x) = 0, \ldots, h_l(x) = 0$$

3. Pick critical point which gives global minimum.

EXACT SOLUTION (EQUALITY)

$$\begin{aligned} \text{minimize} \quad & f(x) = n_1 \log x_1 + \cdots + n_d \log x_d \\ \text{subject to} \quad & h(x) = x_1 + \cdots + x_d - 1 = 0 \end{aligned}$$

Lagrangian

$$L(x, \lambda) = n_1 \log x_1 + \cdots + n_d \log x_d + \lambda(x_1 + \cdots + x_d - 1)$$

Critical points

$$\begin{aligned} 0 &= x_1 + \cdots + x_d - 1 \\ 0 &= n_i / x_i + \lambda \end{aligned} \quad \Rightarrow \quad \begin{aligned} (-\lambda) &= n_1 + \cdots + n_d \\ x_i &= n_i / (-\lambda) \end{aligned}$$

INEQUALITY CONTRAINTS

Primal Problem.

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && g_1(x) \le 0, \ldots, g_m(x) \le 0 \end{aligned}$$

Lagrangian.

$$L(x, \alpha) = f(x) + \alpha_1 g_1(x) + \cdots + \alpha_m g_m(x)$$

Dual Problem.

$$\begin{aligned} &\text{maximize} && \ell(\alpha) \\ &\text{subject to} && \alpha_1 \ge 0, \ldots, \alpha_m \ge 0 \end{aligned} \qquad \text{where } \ell(\alpha) = \min_{x \in \mathbb{R}^d} L(x, \alpha)$$

Box constraints are easier to work with!

INEQUALITY CONTRAINTS

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && g_1(x) \leq 0, \dots, g_m(x) \leq 0 \end{aligned}$$

Lagrangian.

$$L(x, \alpha) = f(x) + \alpha_1 g_1(x) + \cdots + \alpha_m g_m(x)$$

Solve for x, α satisfying

1. $\nabla_x L(x, \alpha) = 0$
2. $g_1(x) \leq 0, \dots, g_m(x) \leq 0$
3. $\alpha_1 \geq 0, \dots, \alpha_m \geq 0$
4. $\alpha_1 g_1(x) = 0, \dots, \alpha_m g_m(x) = 0$

Karush-Kuhn-Tucker (KKT) Conditions

Complementary Slackness