FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS UNIVERSIDAD DE CHILE

CMM Center for Mathematical Modeling

Introduction to Machine Learning

Guillermo Cabrera
gcabrera@dim.uchile.cl
CMM, Universidad de Chile

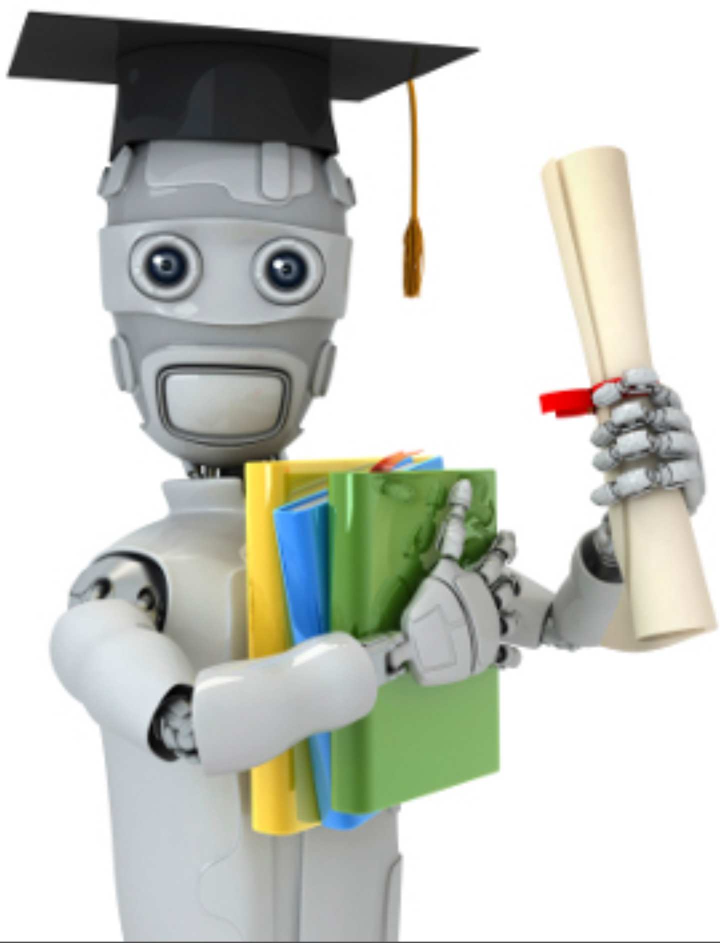
Machine Learning

"Can machines think?"

Turing, Alan (October 1950), "Computing Machinery and Intelligence",

"Can machines do what we (as thinking entities) can do?"

Mitchell, T. (1997), "Machine Learning", McGraw Hill



Machine Learning

"Can machines think?"

Turing, Alan (October 1950), "Computing Machinery and Intelligence",

"Can machines do what we (as thinking entities) can do?"

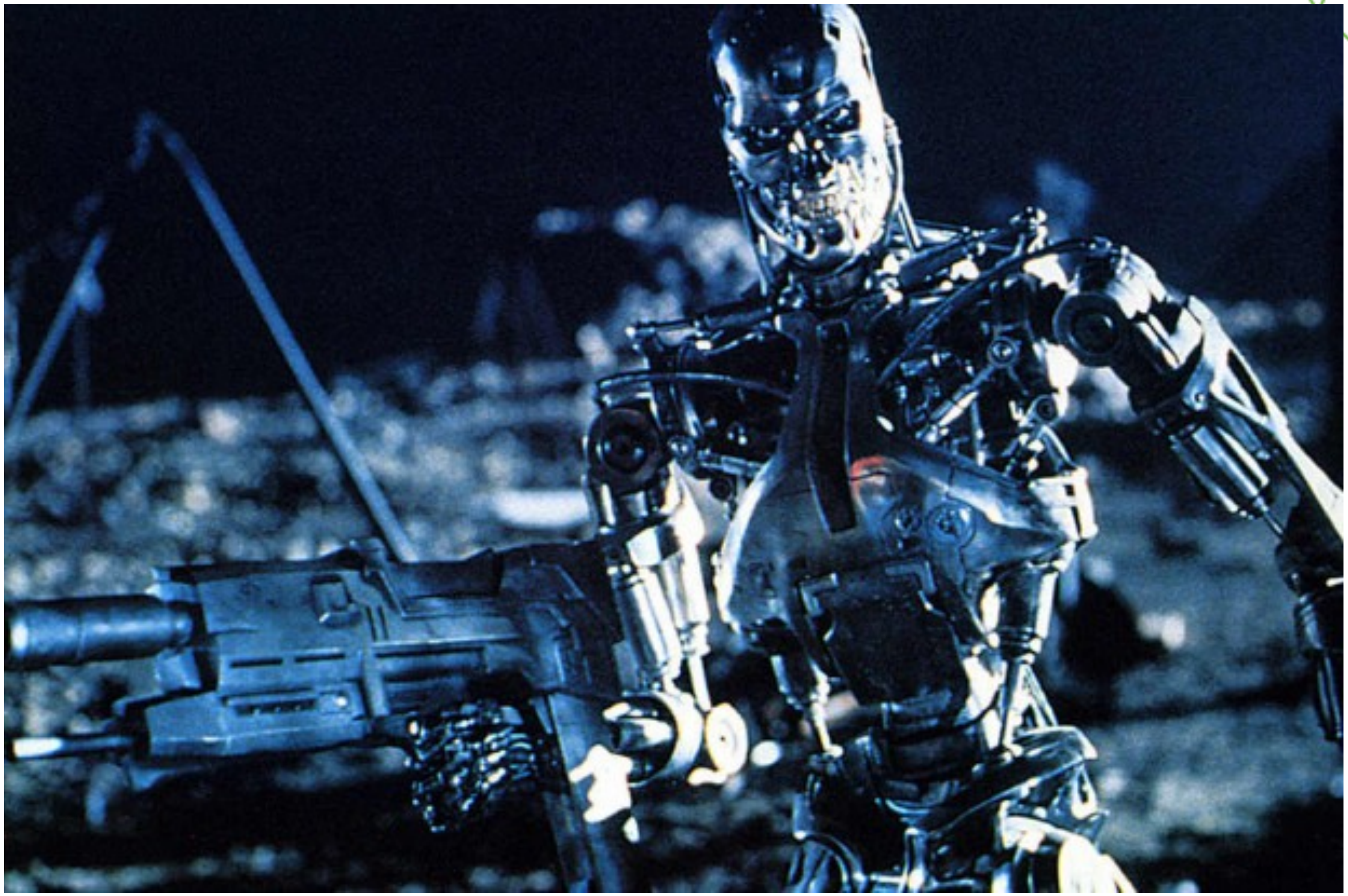
Mitchell, T. (1997), "Machine Learning", McGraw Hill

"Three laws of robotics"

1. A machine may not injure a human being or, through inaction, allow a human being to come to harm.
2. A machine must obey the orders given to it by human beings, except where such orders would conflict with the First Law.
3. A machine must protect its own existence as long as such protection does not conflict with the First or Second Law

Isaac Asimov

Machine Learning



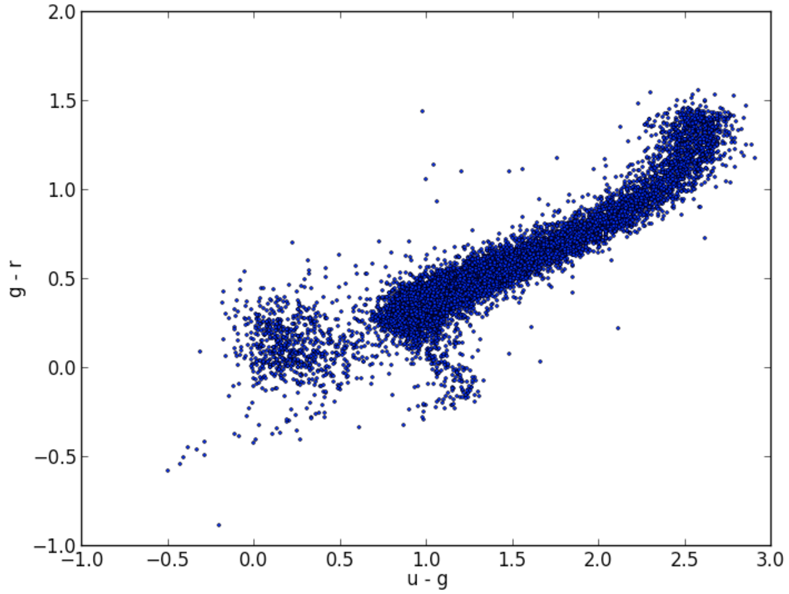
Machine Learning

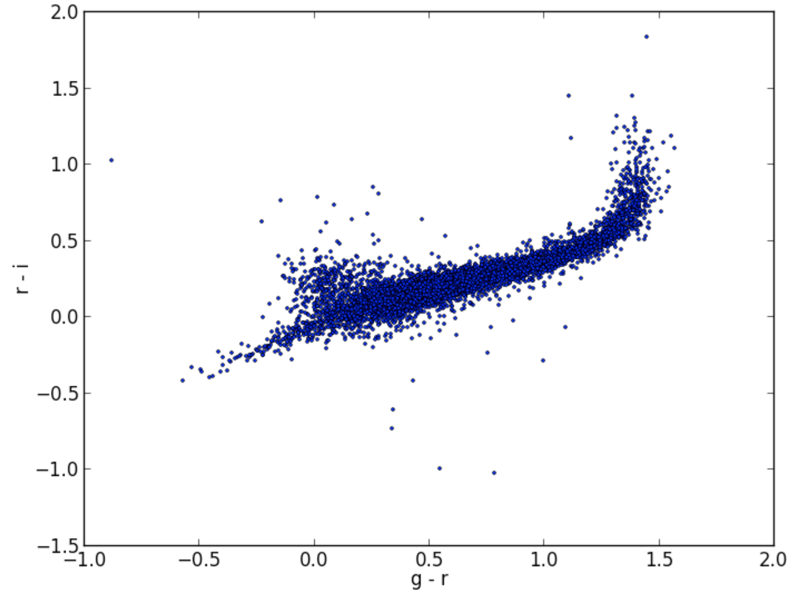
- Study of algorithms that
  - improve their performance P
  - at some task T
  - with experience E
- well-defined learning task: <P, T, E>

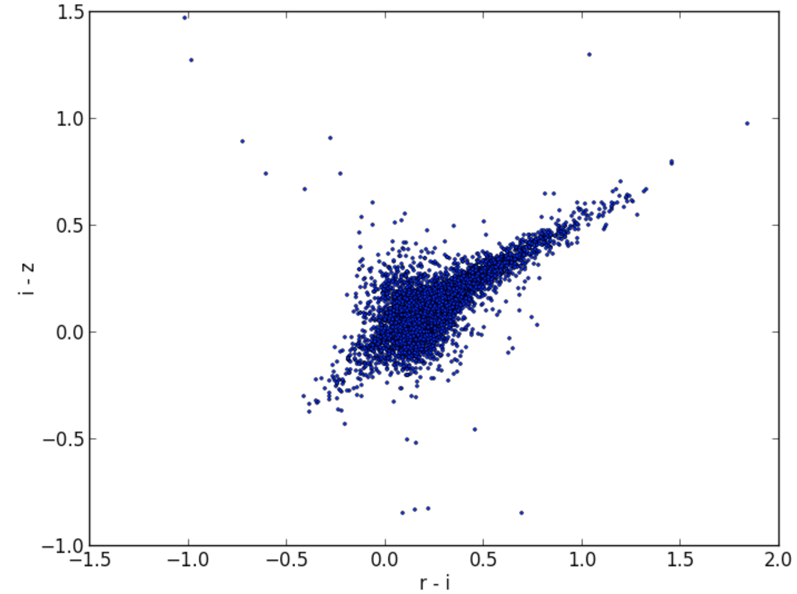
A Machine Learning Problem

- Point sources ... what are they?
- Selection of SDSS point sources.

  http://astrostatistics.psu.edu/MSMA/datasets/
- Get their colors.
- How many different kind of objects can we distinguish?







Formal Definition

- Problem Setting:
  - Set of possible instances X
  - Unknown target function f : $X \rightarrow Y$
  - Set of function hypotheses $H = \{ h \mid h : X \rightarrow Y \}$
- Input:
  - Training examples $\{\langle x_i, y_i \rangle\}$ of unknown target function f
- Output:
  - Hypothesis $h \in H$ that best approximates target function f

Two approaches

- Do we have some already labeled data?
- Yes: Supervised Learning
  - ANN, SVM, Decision Trees, Bayesian Classificators, Nearest Neighbours, etc...
- No: Unsupervised Learning
  - Clustering: K-Means, Hierarchical Clustering, DBSCAN, etc...

Supervised Learning (Classification)

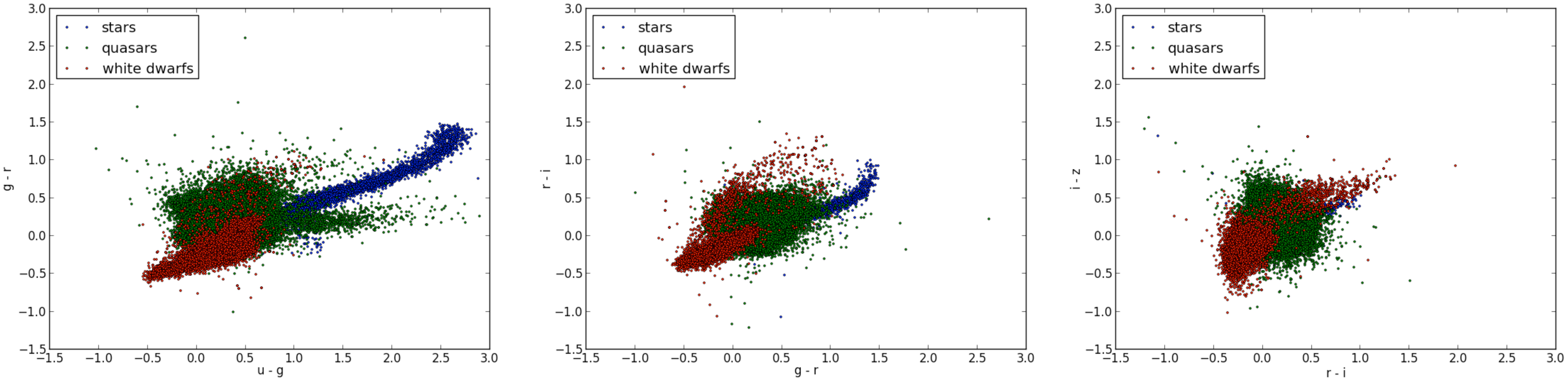
- Given a **training set** {<x_i,y_i>}
  - x_i: **attributes,** y_i: **classes**
- Determine a learning function f : X→Y
- Goal: predict class of a given set of attributes
  - y = f(x)
- Very important: a separate **testing set** is used to validate our classificator.

Supervised Learning (Classification)

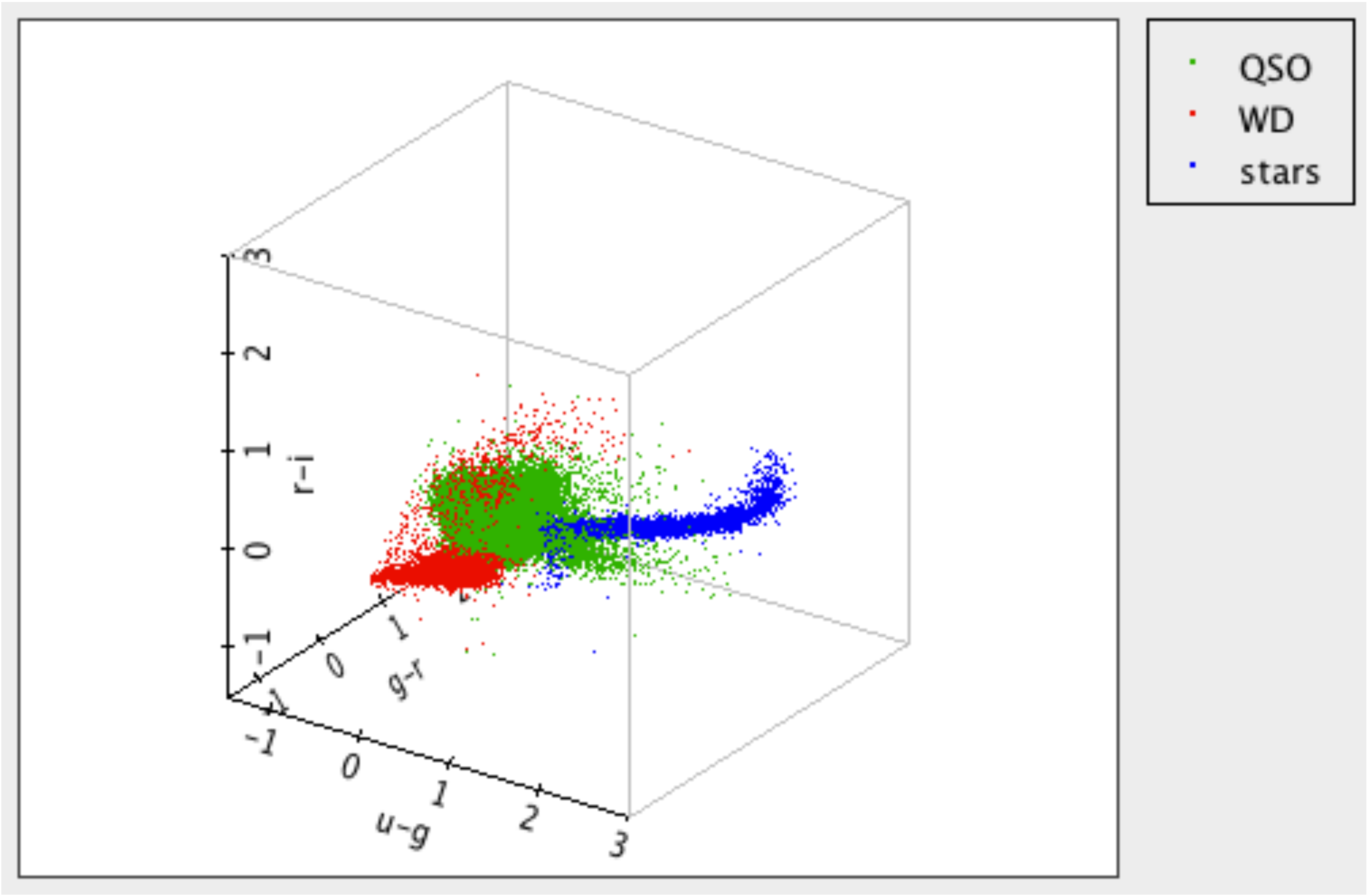
- Given a **training set** $\{<x_i, y_i>\}$
  - x_i: **attributes,** y_i: **classes**
- Determine a learning function $f : X \to Y$
- Goal: predict class of a given set of attributes
  - $y = f(x)$
- Very important: a separate **testing set** is used to validate our classificator.

A Supervised Learning Problem

- Point sources ... what are they?
- A selection of SDSS point sources, along with training sets for three spectroscopically confirmed classes:
  1. main-sequence plus red-giant stars
  2. quasars
  3. white dwarfs



A Supervised Learning Problem

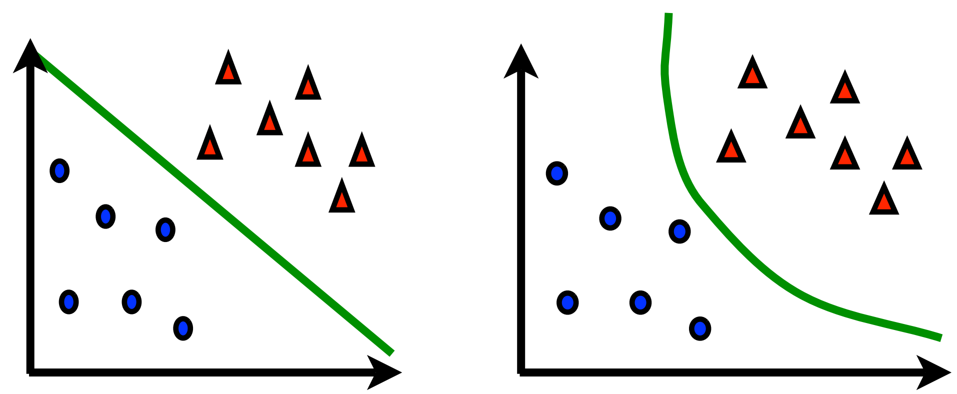


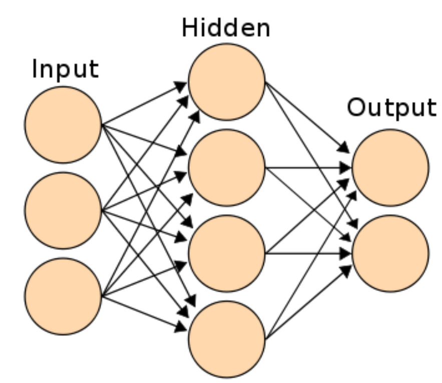
Supervised Learning (Classification)

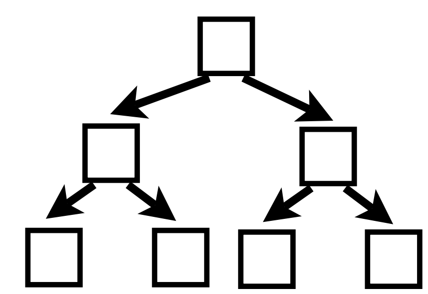
- Given a **training set** $\{<x_i, y_i>\}$
  - x_i: **attributes,** y_i: **classes**
- Determine a learning function $f : X \rightarrow Y$
  - Goal: predict class of a new set of attributes
  - $y = f(x)$
- Very important: a separate **testing set** is used to validate our classificator.

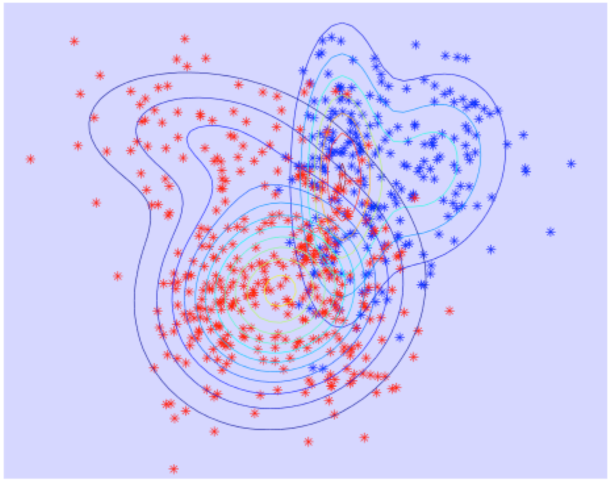
Some Classification Algorithms

- Support Vector Machines
- Artificial Neural Networks
- Decision Trees
- Gaussian Mixture Models
- Ensambles



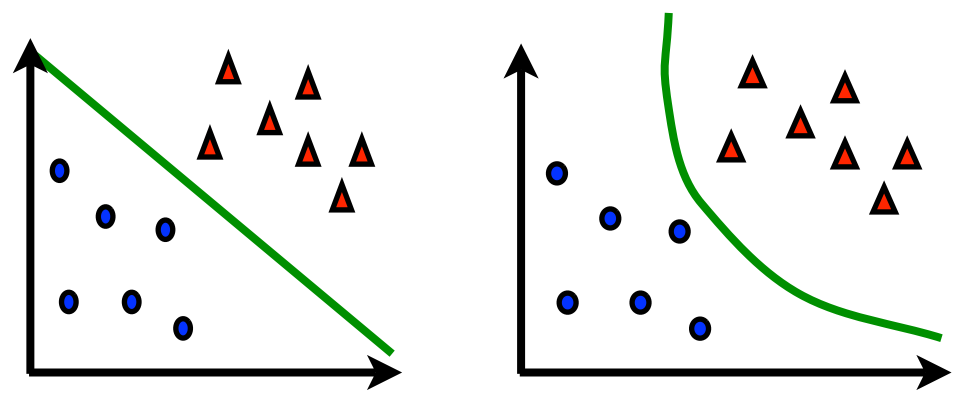


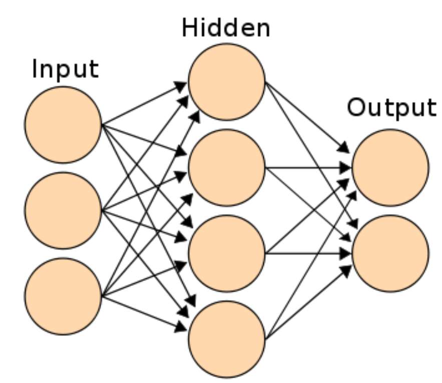


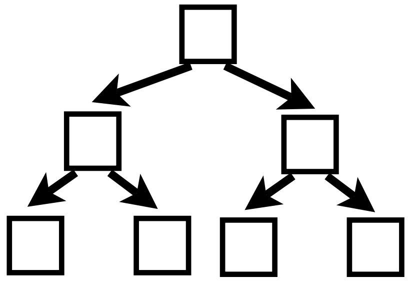


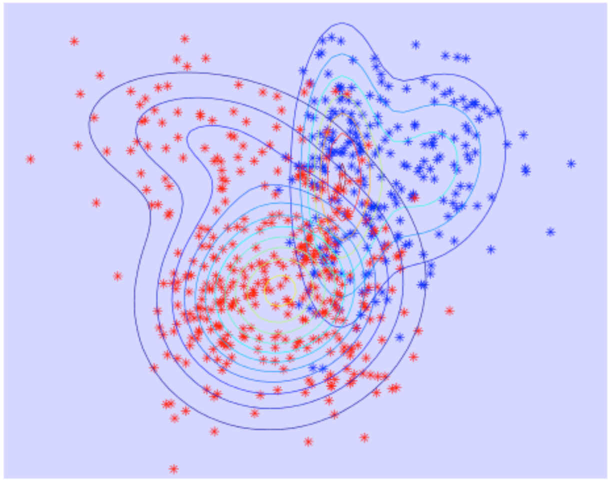
Some Classification Algorithms

- Support Vector Machines
- Artificial Neural Networks
- Decision Trees
- Gaussian Mixture Models
- Ensambles

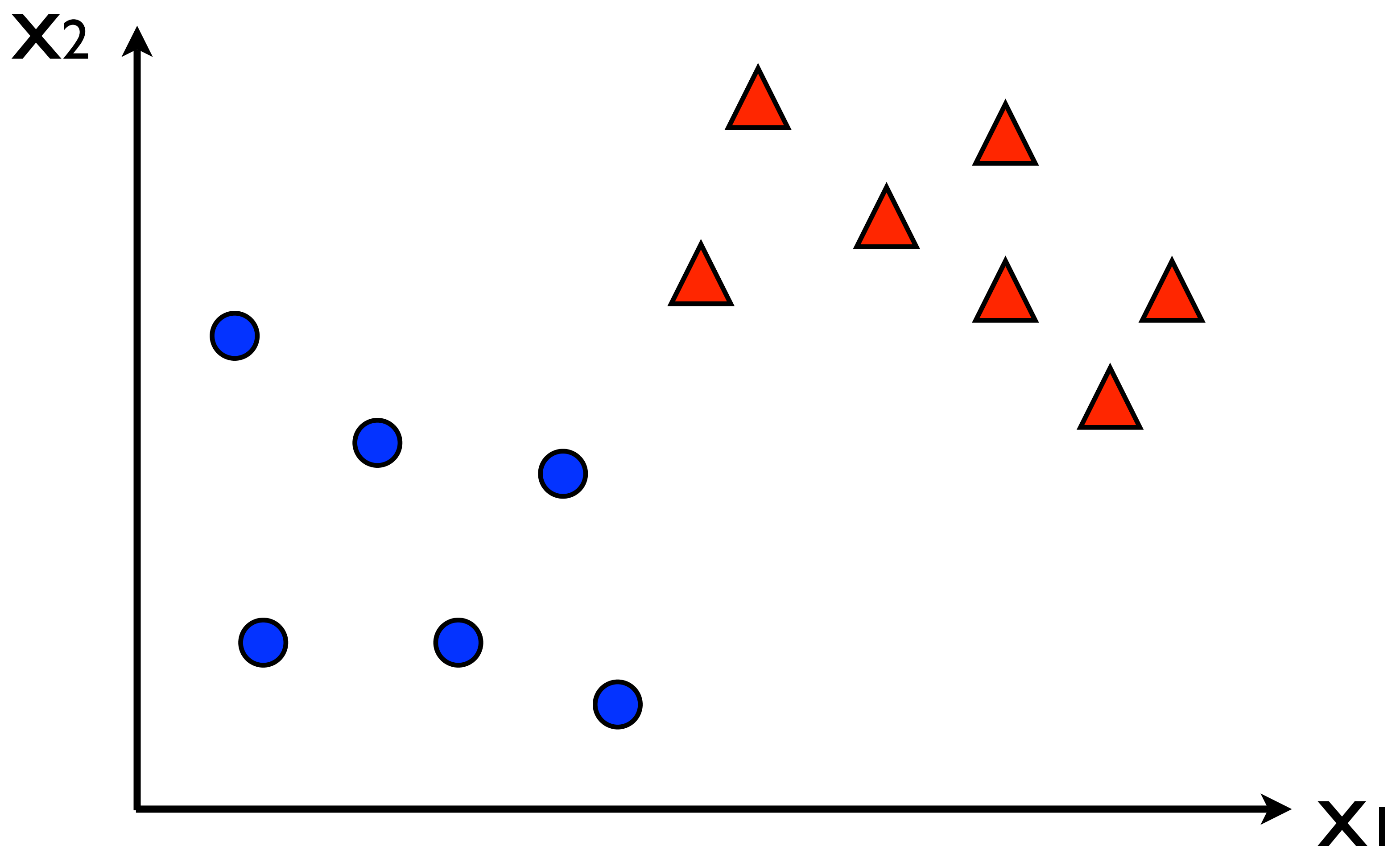






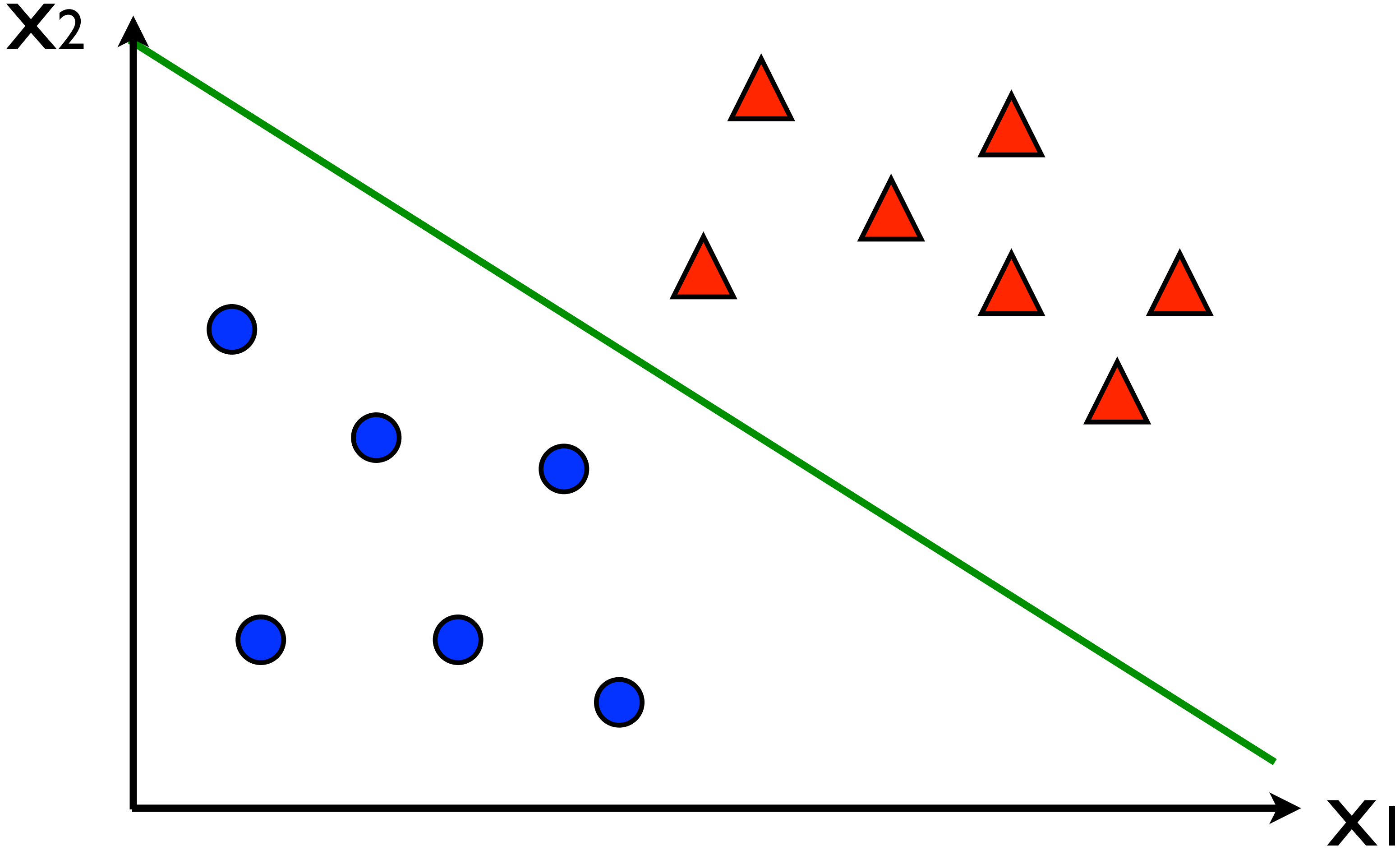


Support Vector Machines

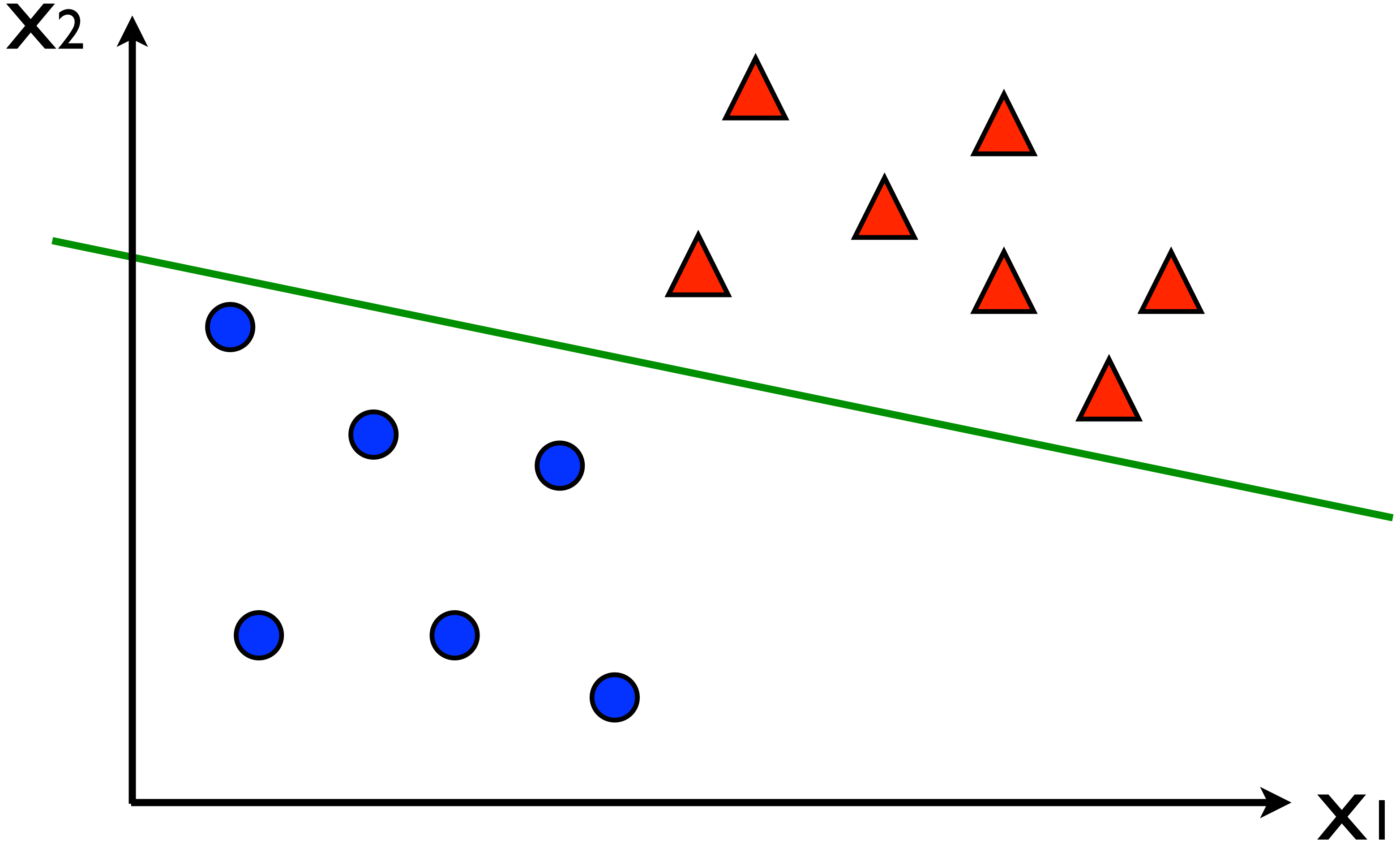


$$\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbb{R}^p,\, y_i \in \{-1, 1\}\}_{i=1}^{n}$$

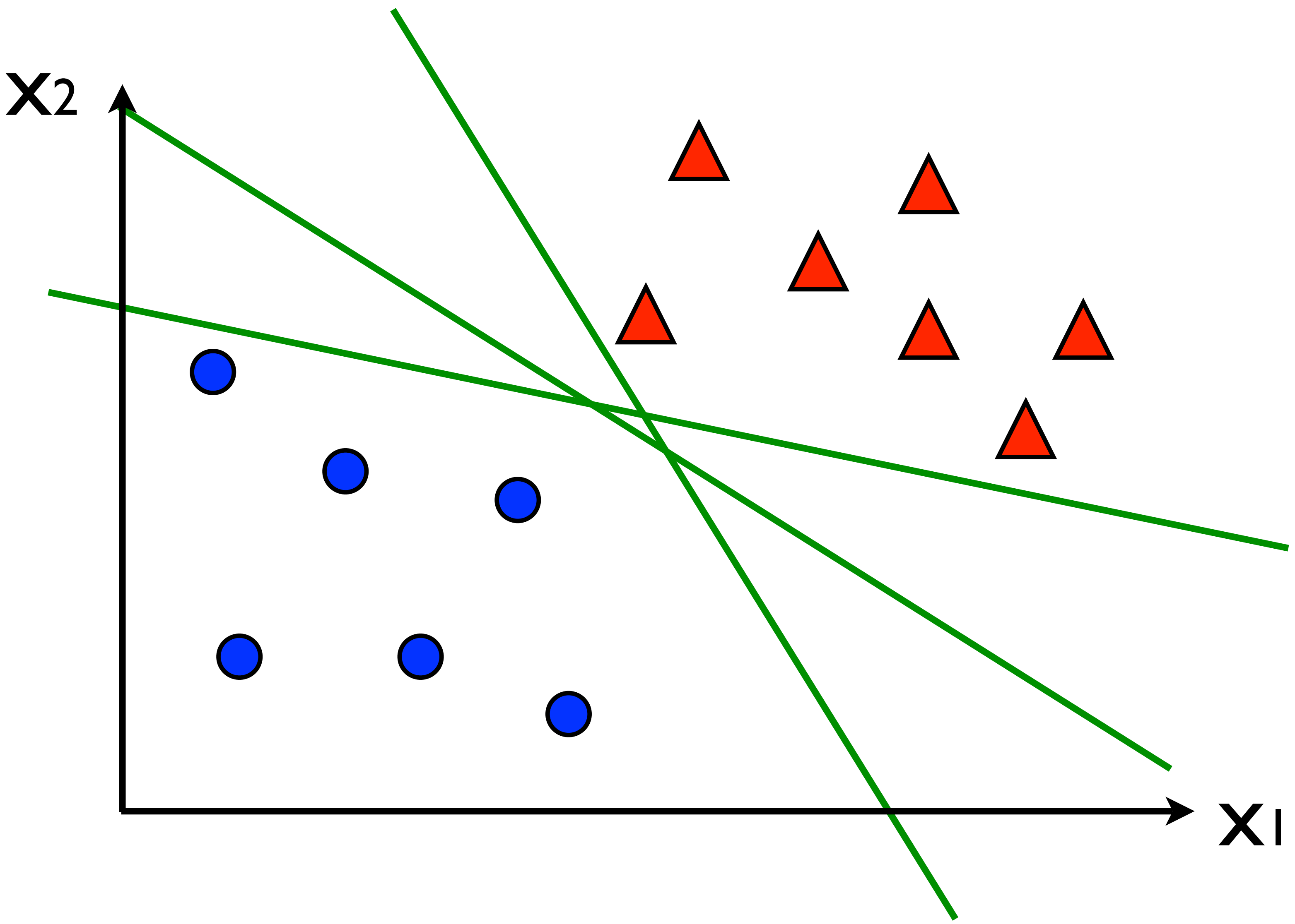
Support Vector Machines



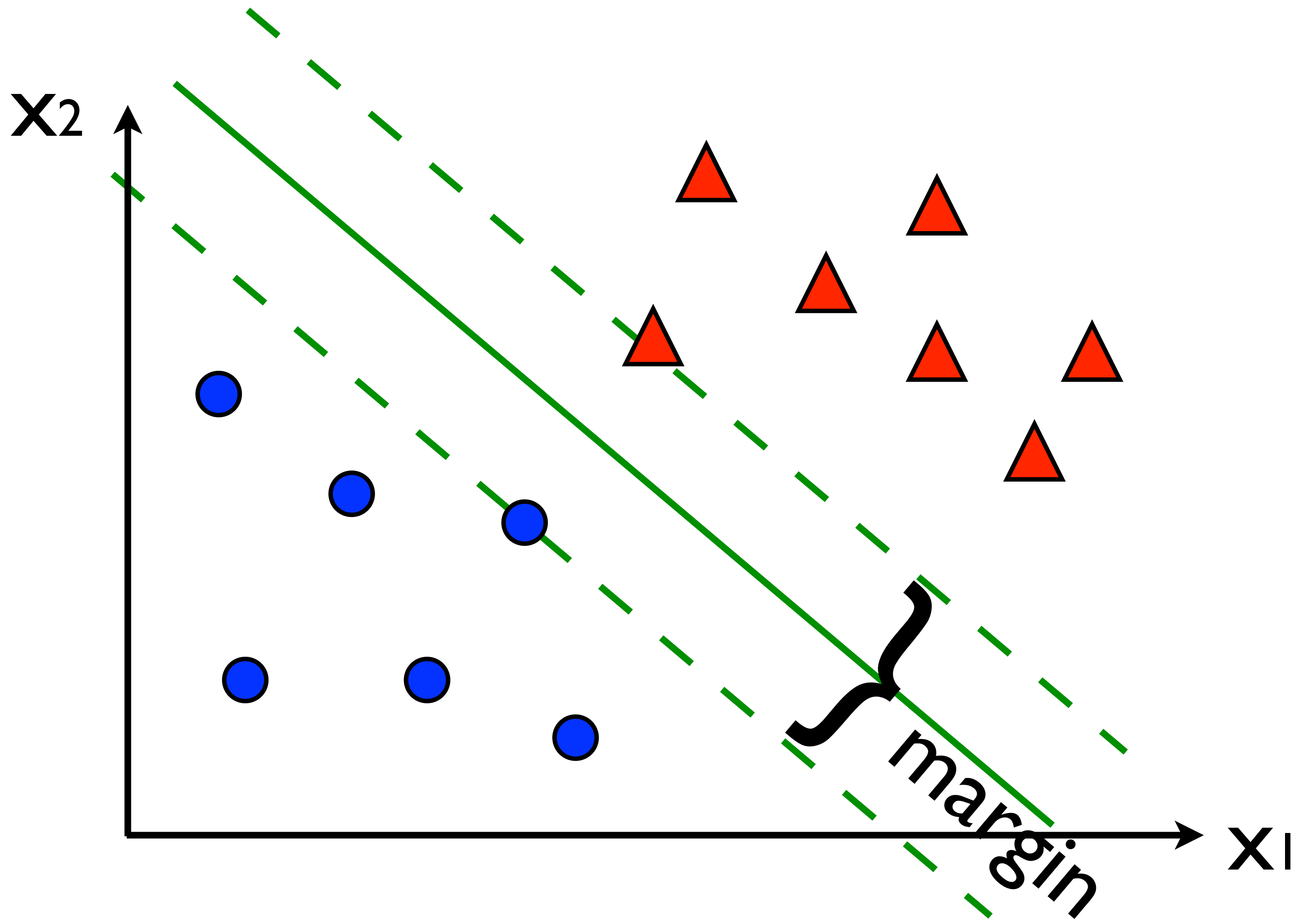
Support Vector Machines



Support Vector Machines

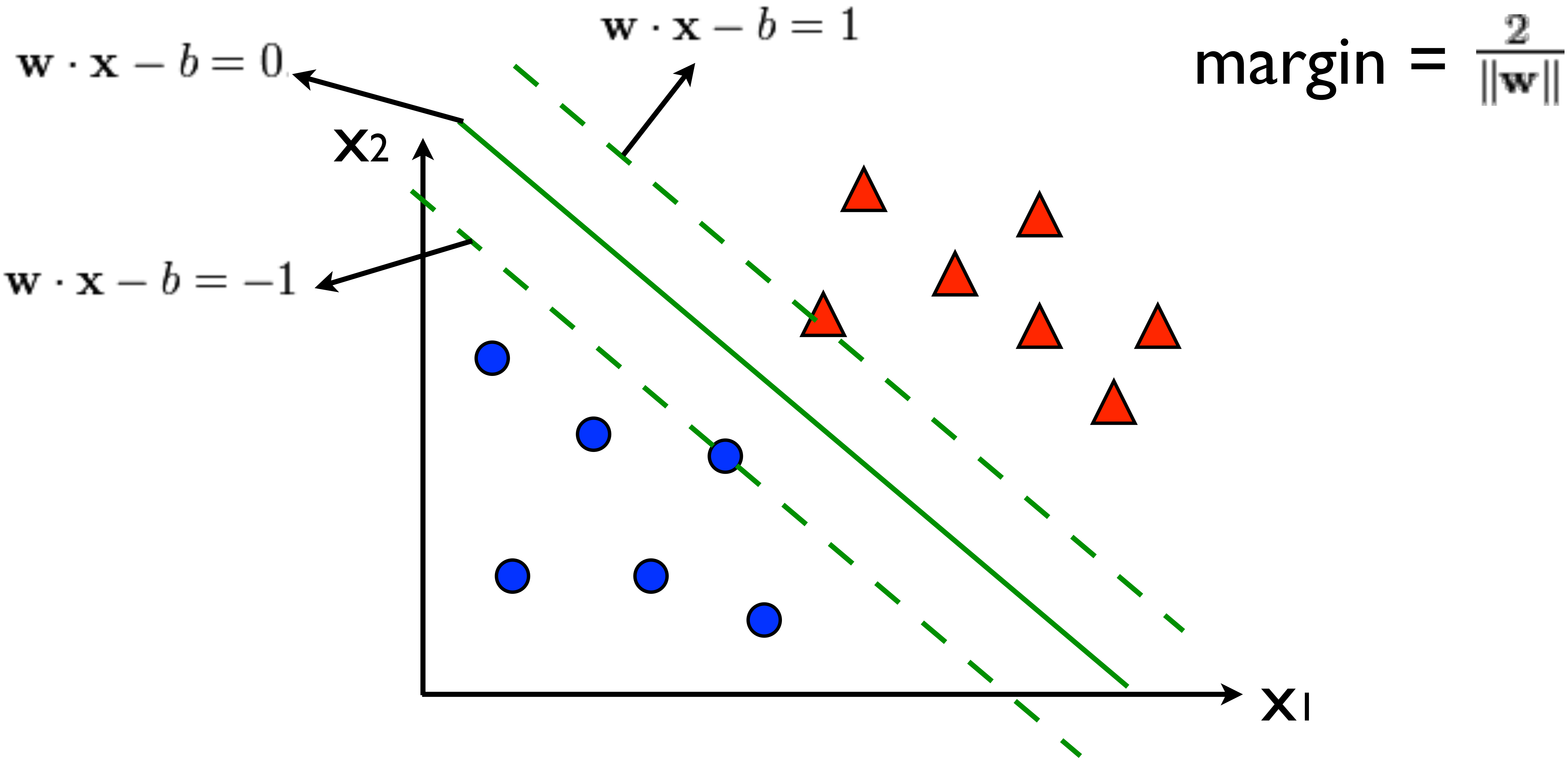


Support Vector Machines



Goal: maximize margin

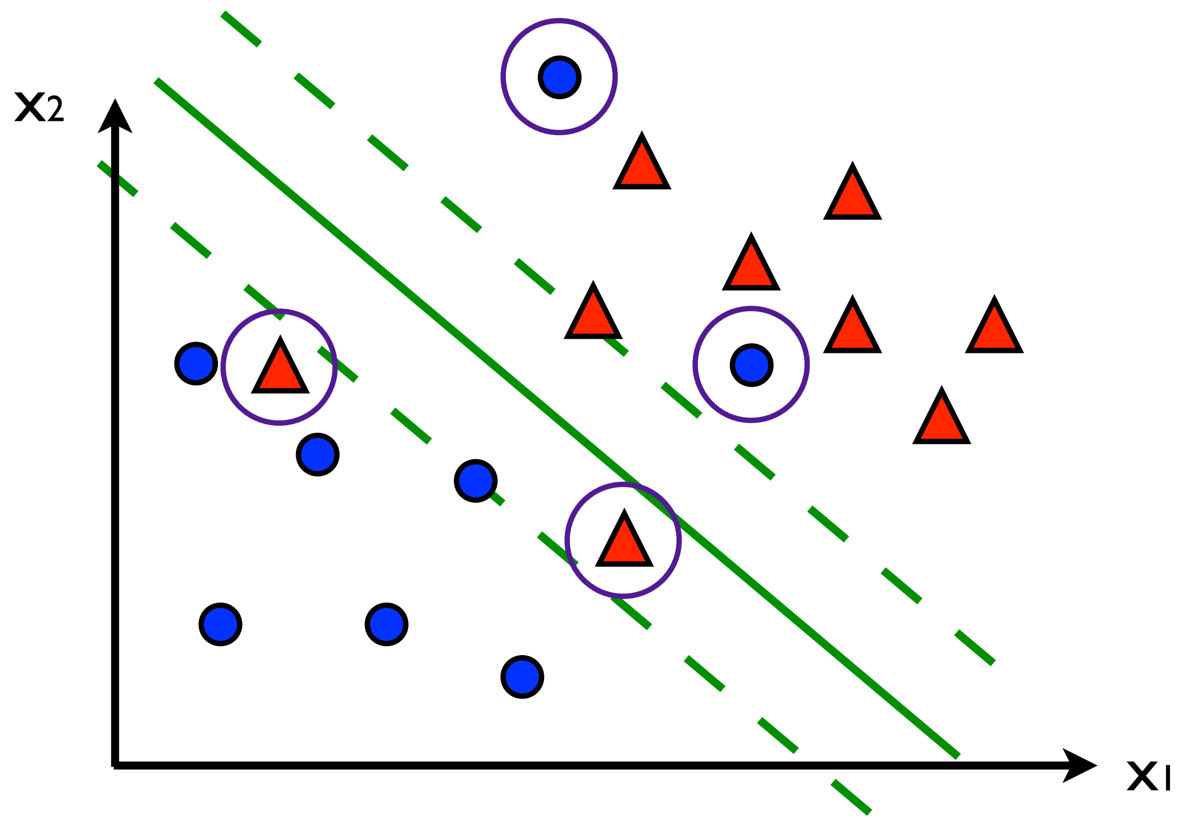
Support Vector Machines



$\min_{\mathbf{w},b} \|\mathbf{w}\|$, subject to $y_i(\mathbf{w}\cdot\mathbf{x_i}-b)\geq 1.$

Support Vector Machines

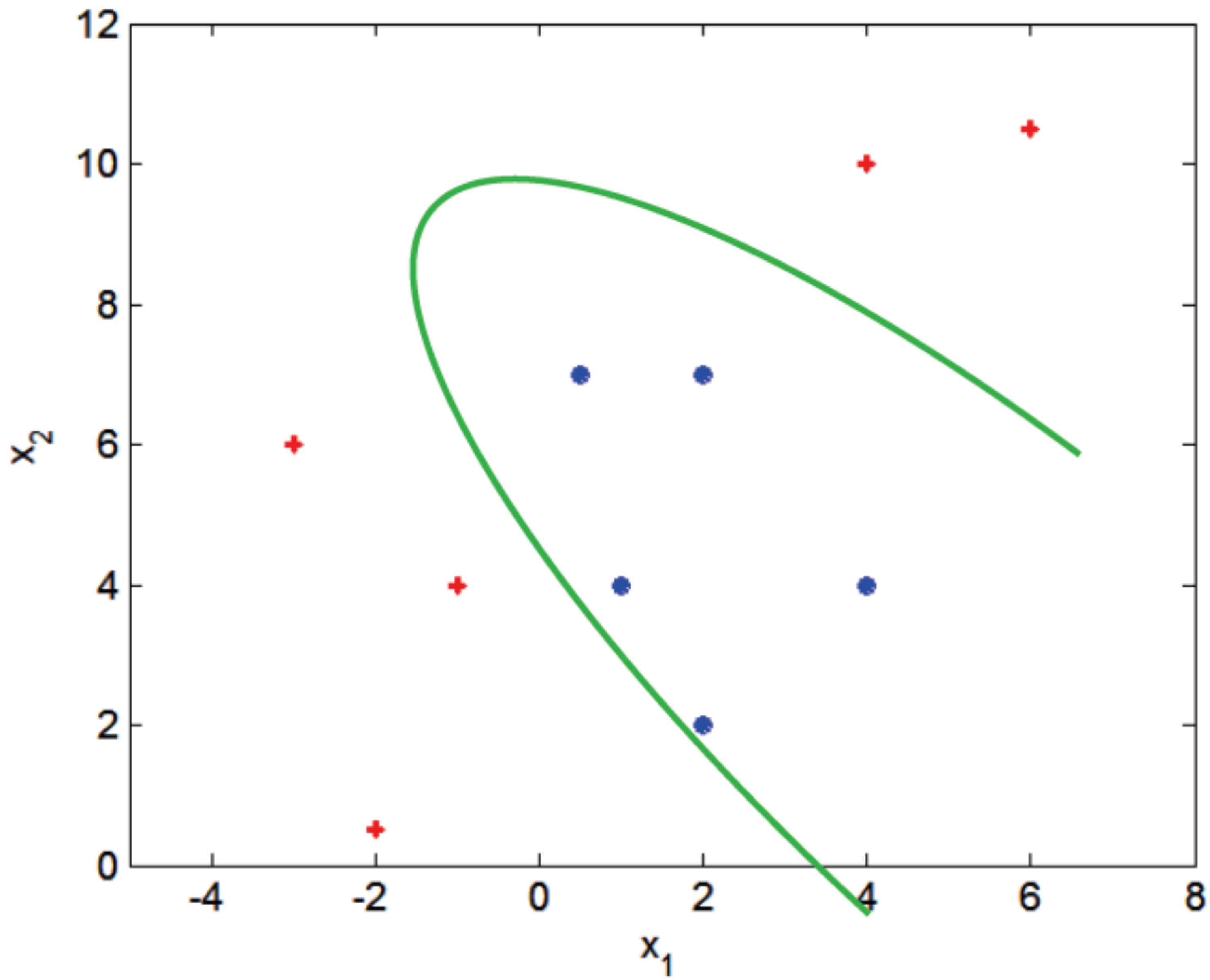
overlap in attribute space: soft margin



$$\min_{\mathbf{w},\xi,b} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i \right\}$$

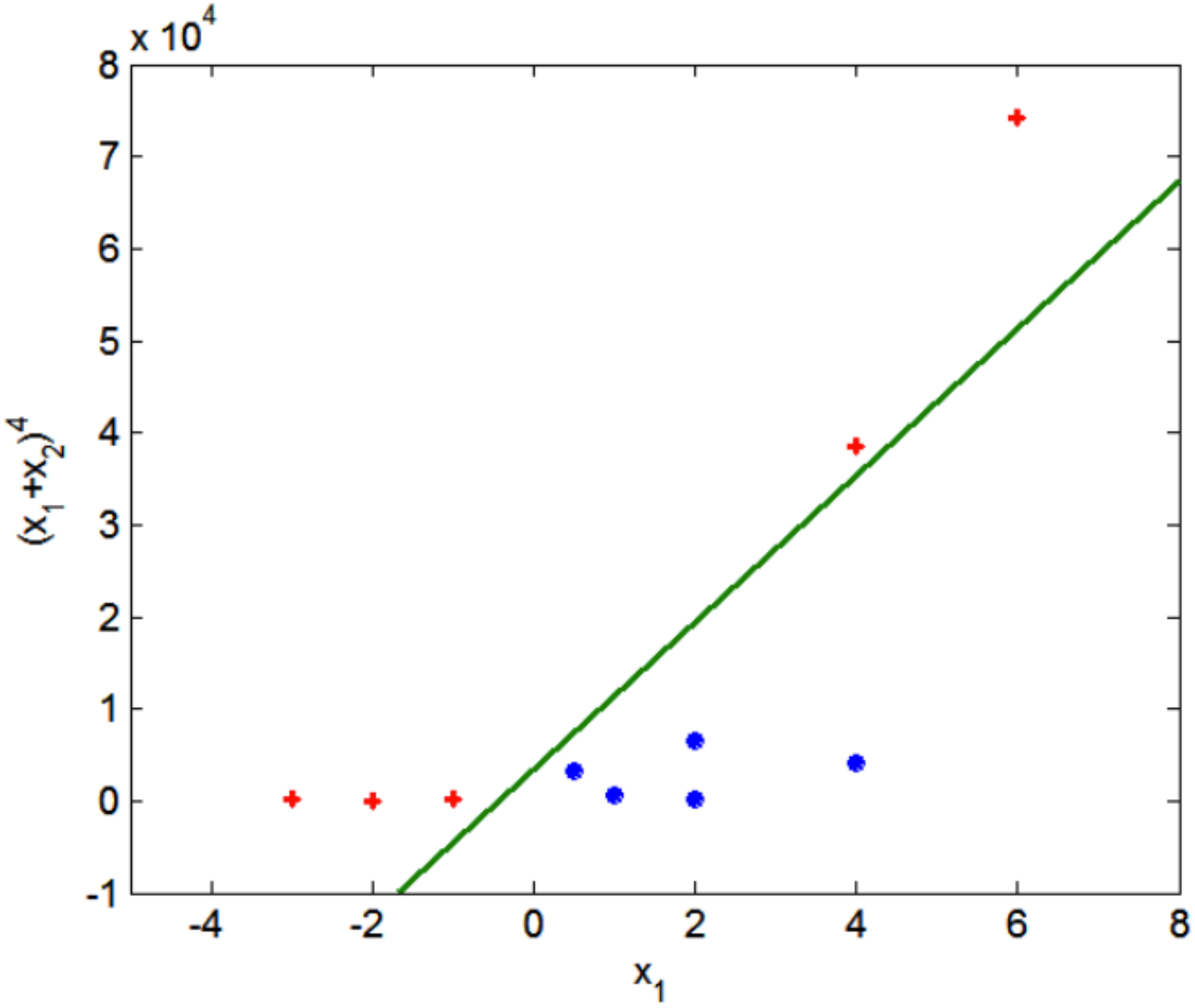
subject to $y_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1 - \xi_i, \quad \xi_i \geq 0$

Support Vector Machines
nonlinear classification



Support Vector Machines
nonlinear classification

Kernel trick: space transformation



Support Vector Machines
nonlinear classification

Kernel trick: space transformation

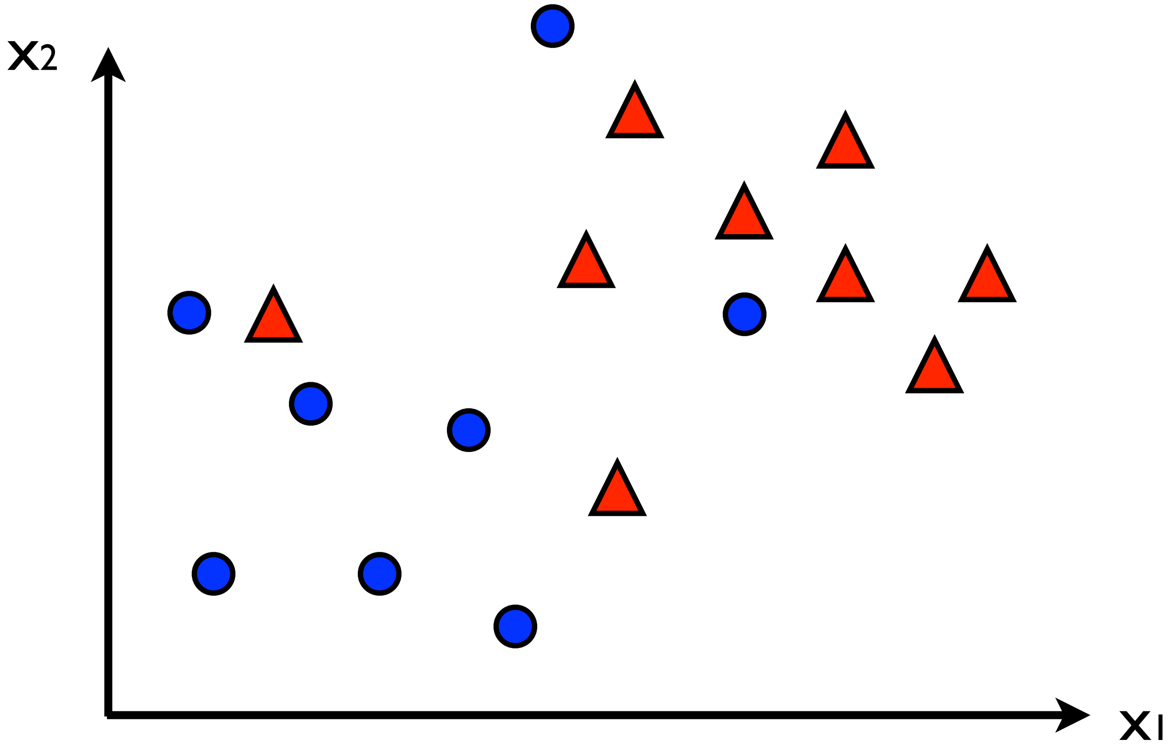
- linear $K(x, x') = \langle x, x' \rangle$
- polynomial $(\gamma\langle x, x' \rangle + r)^d$
- radial basis function $\exp(-\gamma|x - x'|^2)$
- sigmoid $\tanh(\gamma\langle x, x' \rangle + r)$

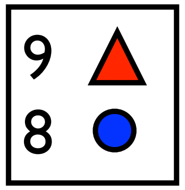
Separation hypersurface:

$$h(\boldsymbol{x}) = \sum_{i}^{N_{\mathrm{SV}}} a_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b_0 \qquad \boldsymbol{b} = \sum_{i}^{N} a_i y_i \boldsymbol{x}_i$$

Decision Trees

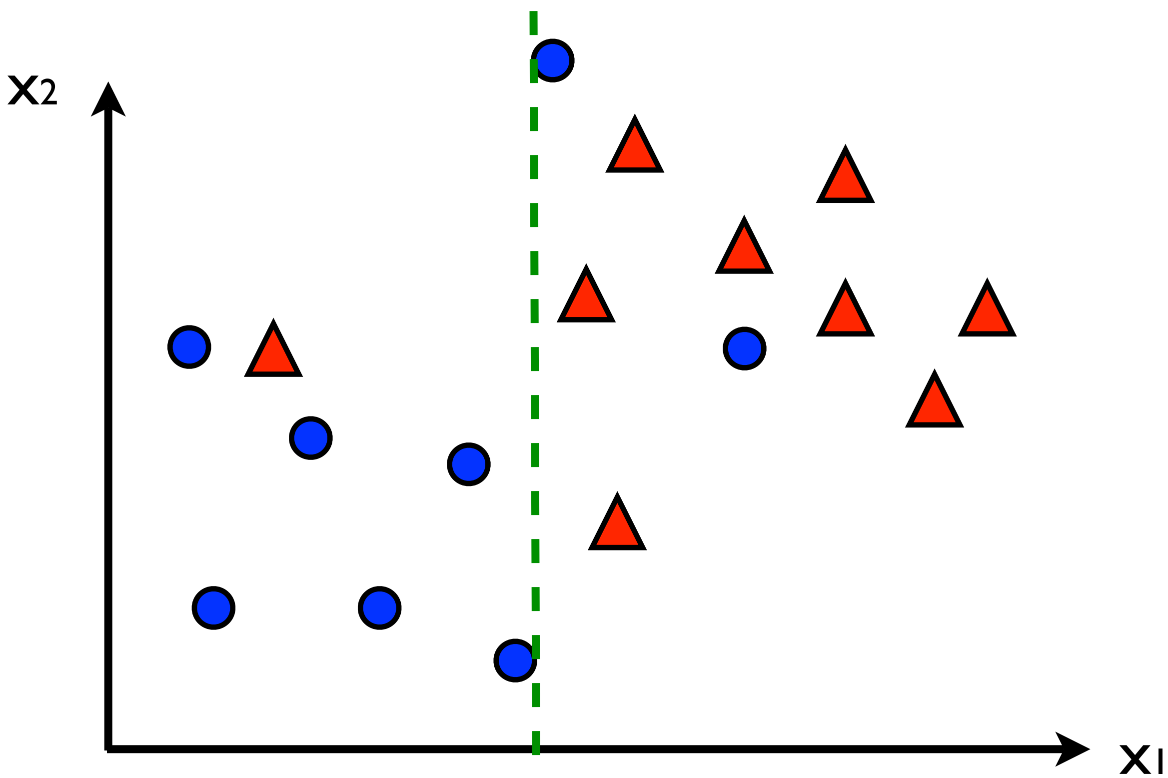
$<x_i, y_i>$

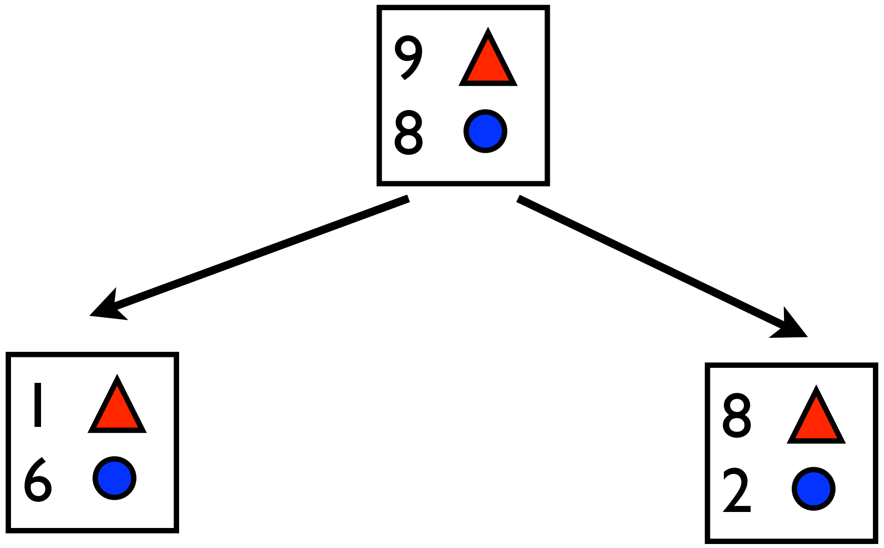




Decision Trees

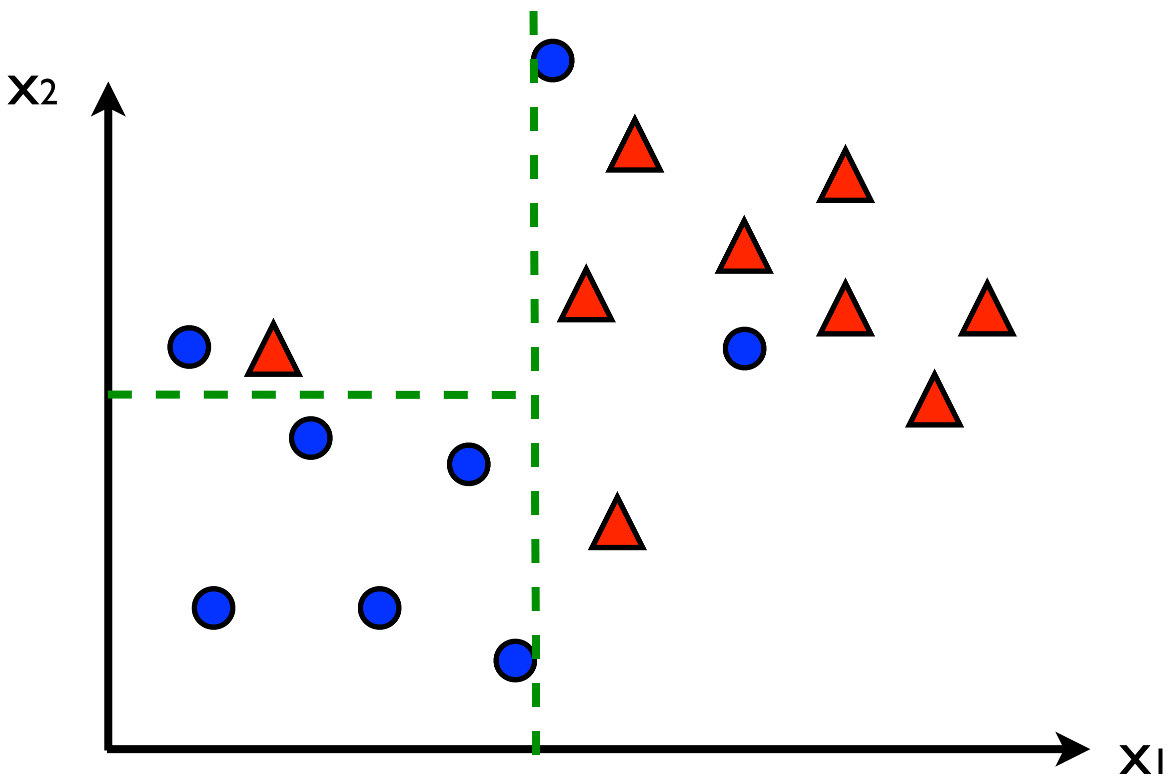
$\langle x_i, y_i \rangle$

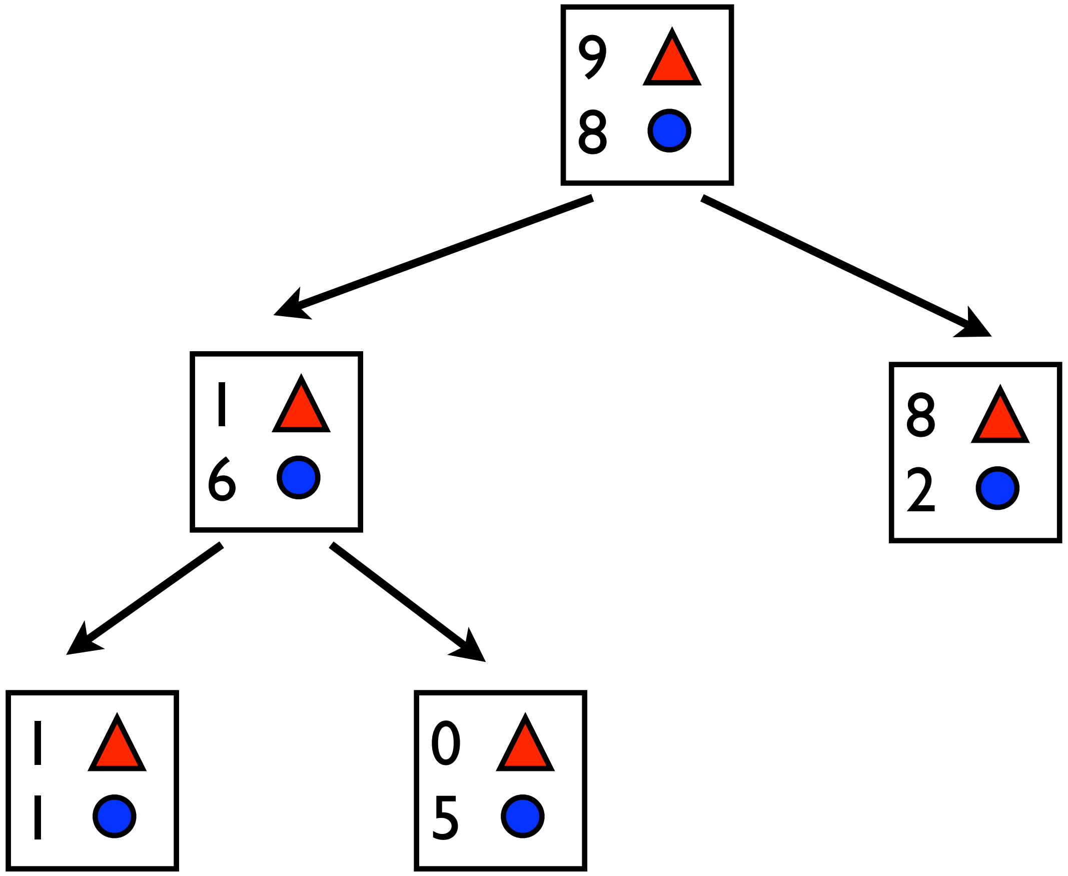




Decision Trees

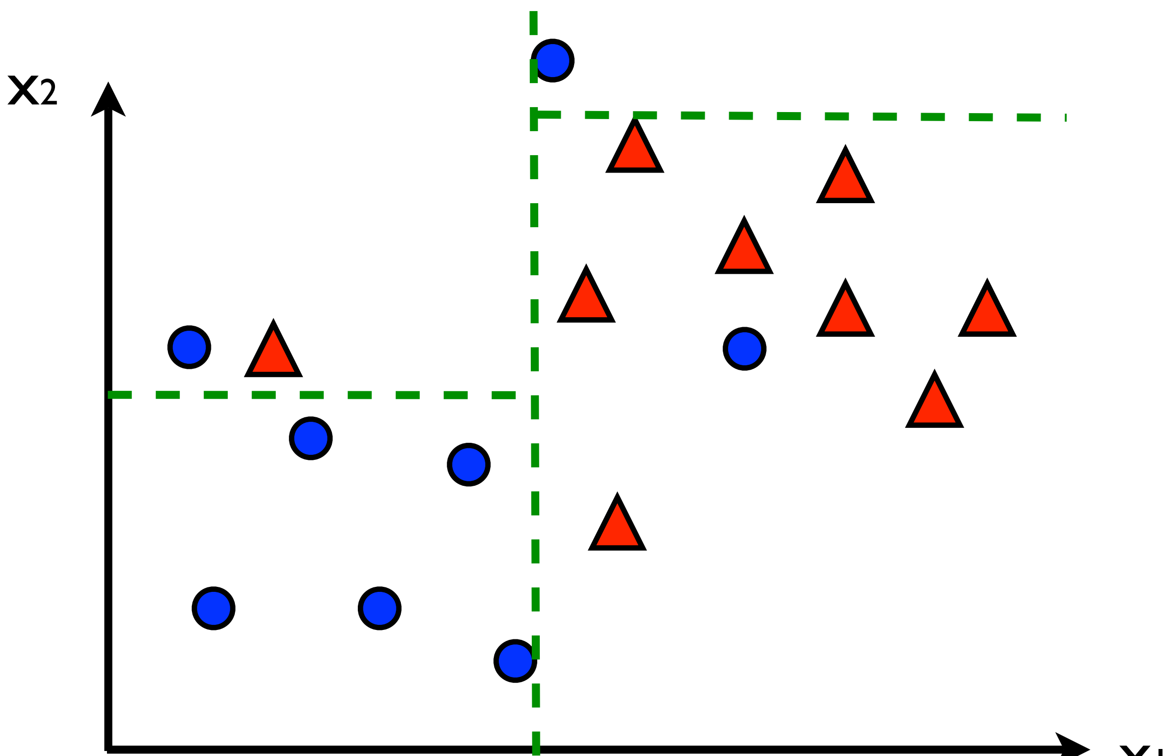
<x_i,y_i>

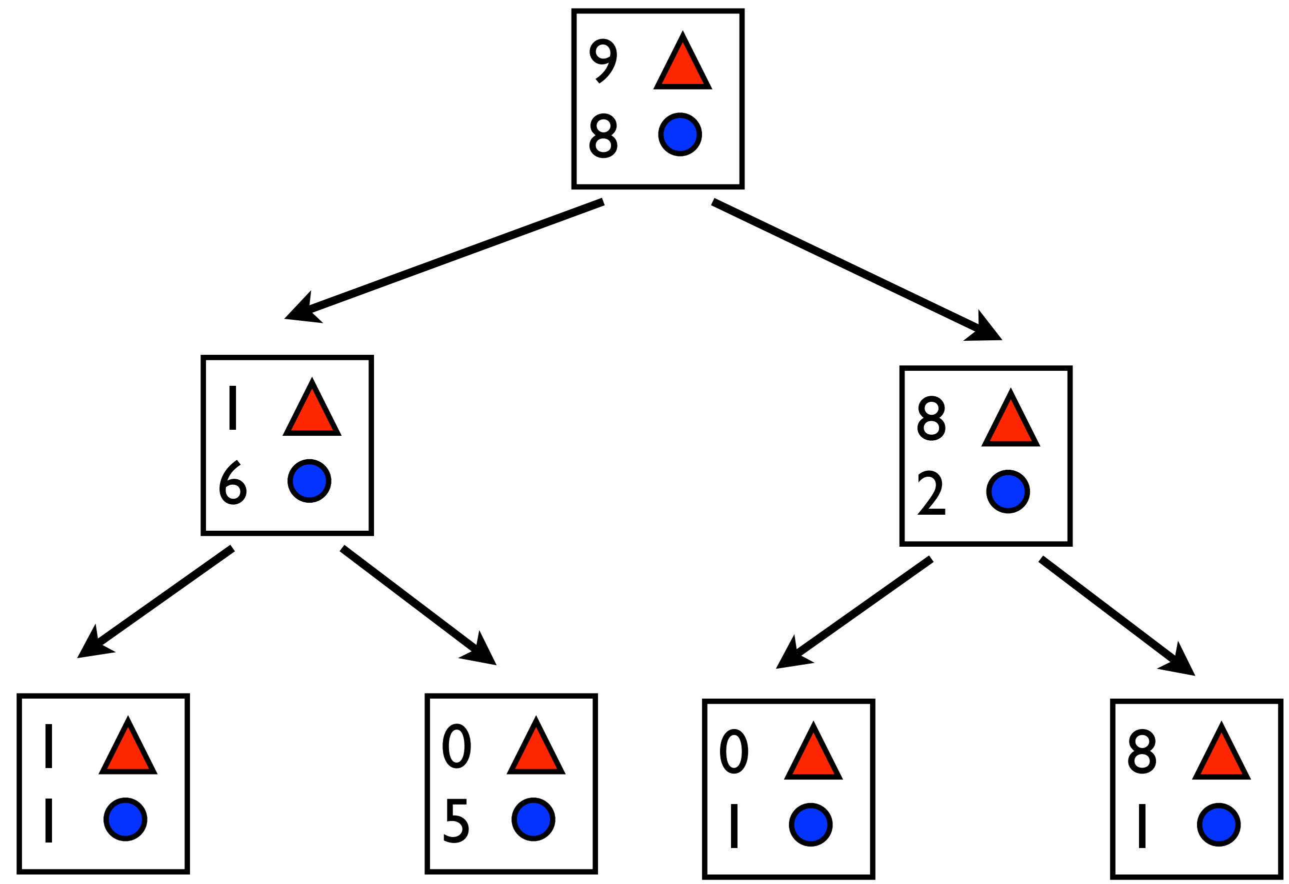




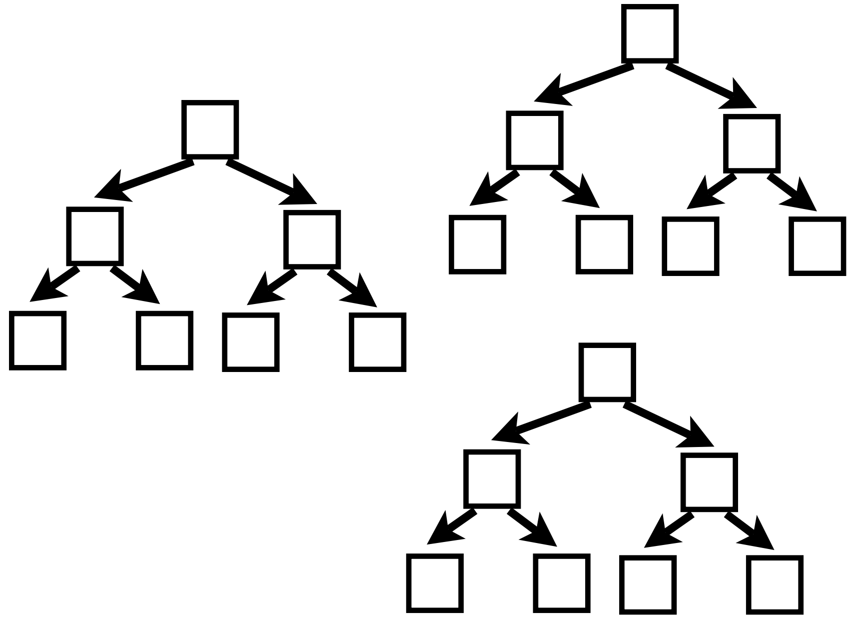
Decision Trees

$\langle x_i, y_i \rangle$





Random Forests



- n trees
- for each tree:
  - select a bootstrap sample (drawing with replacement)
  - train using m randomly chosen attributes for splitting each node
- $f(x) = \frac{1}{n} \sum f_i(x)$

Supervised Learning (Classification)

- Given a **training set** $\{<x_i, y_i>\}$
  - x_i: **attributes,** y_i: **classes**
- Determine a learning function $f : X \rightarrow Y$
  - Goal: predict class of a new set of attributes
  - $y = f(x)$
- Very important: a separate **test set** is used to validate our classificator.

Validation: estimating errors

$$\text{Accuracy} = \frac{\text{Number of correct classifications}}{\text{Total number of objects}}$$

$$\text{Error} = \frac{\text{Number of wrong classifications}}{\text{Total number of objects}}$$

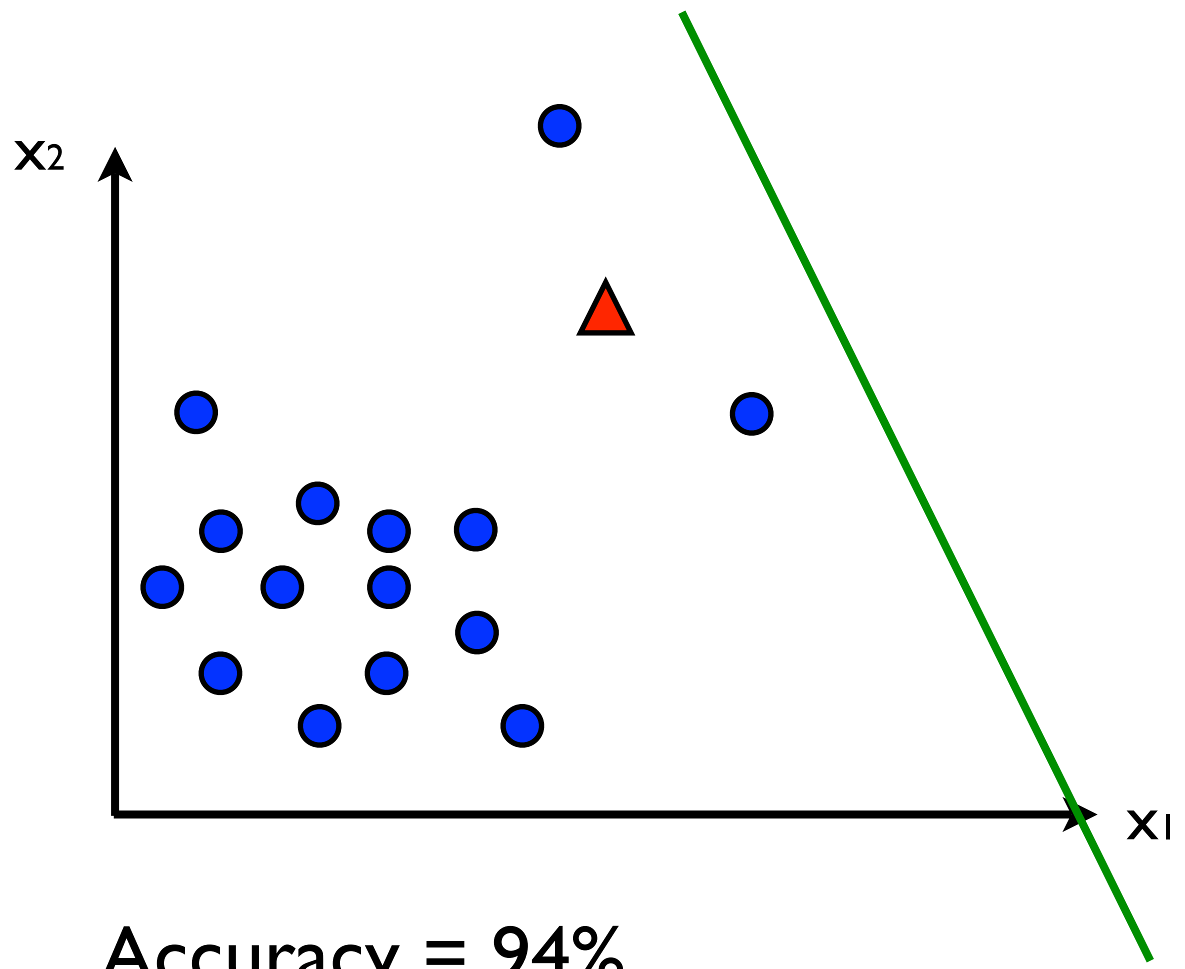
Errors for Binary Classification

True Positive Rate (sensitivity or recall) = TP / (TP + FN)
True Negative Rate (specificity)= TN /(FP + TN)

False Positive Rate (fall-out) = FP / (FP + TN)
False Negative Rate = FN / (TP + FN)

Positive Predictive Value (precision) = TP / (TP + FP)
Negative Predictive Value = TN / (TN + FN)

What if we have an unbalanced data set?



Accuracy = 94%
Error = 6%

Great! Isn't it?
But we're missing all  !

Balanced Accuracy and Error

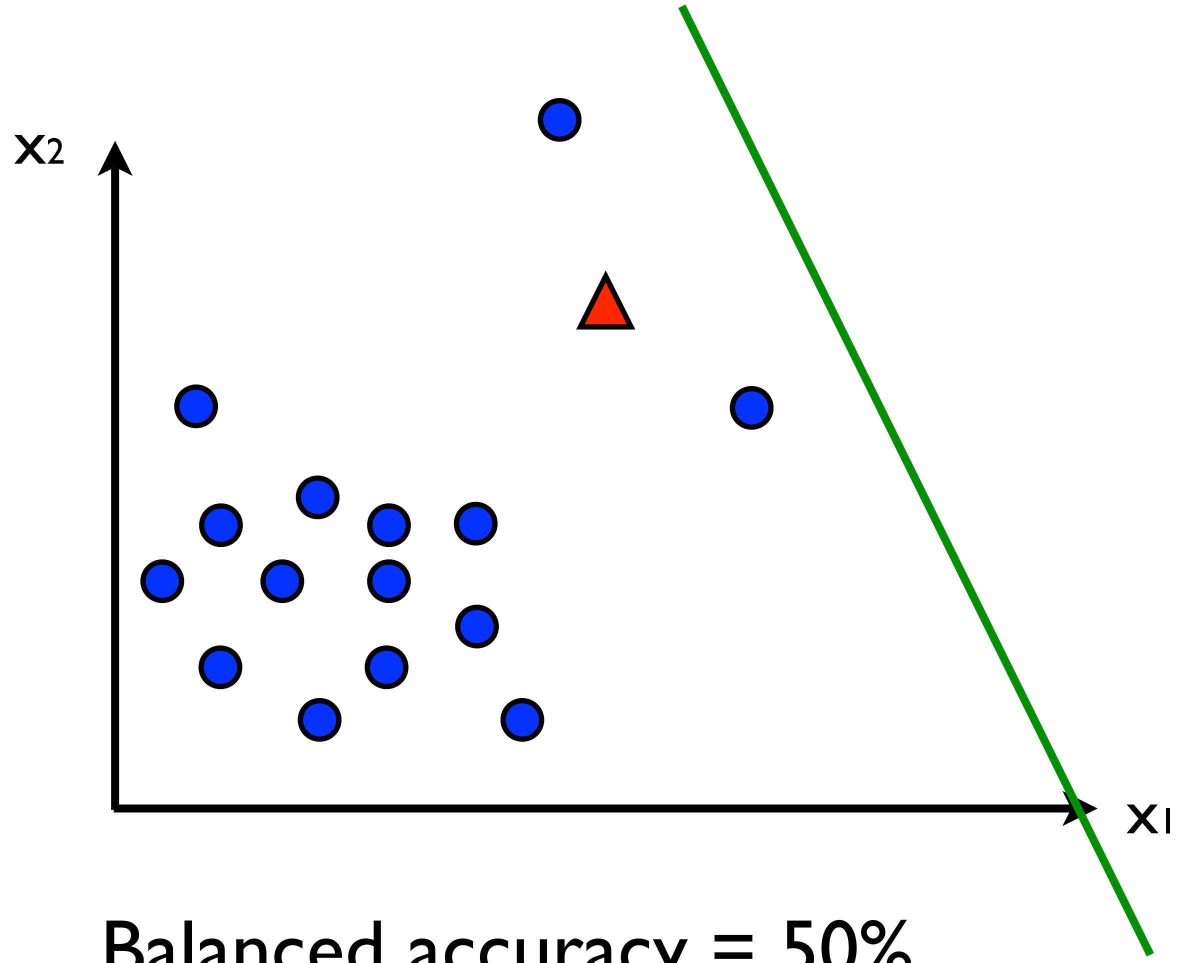
Balanced accuracy = $\dfrac{1}{N_{classes}} \sum_{i}^{N_{classes}} f(c_i|c_i)$

fraction of objects: $f(c_i|c_j) = \dfrac{N(c_i|c_j)}{N(c_j)}$

$N(c_j)$: number of objects of class c_j

Balanced error = 1 - Balanced accuracy

What if we have an unbalanced data set?



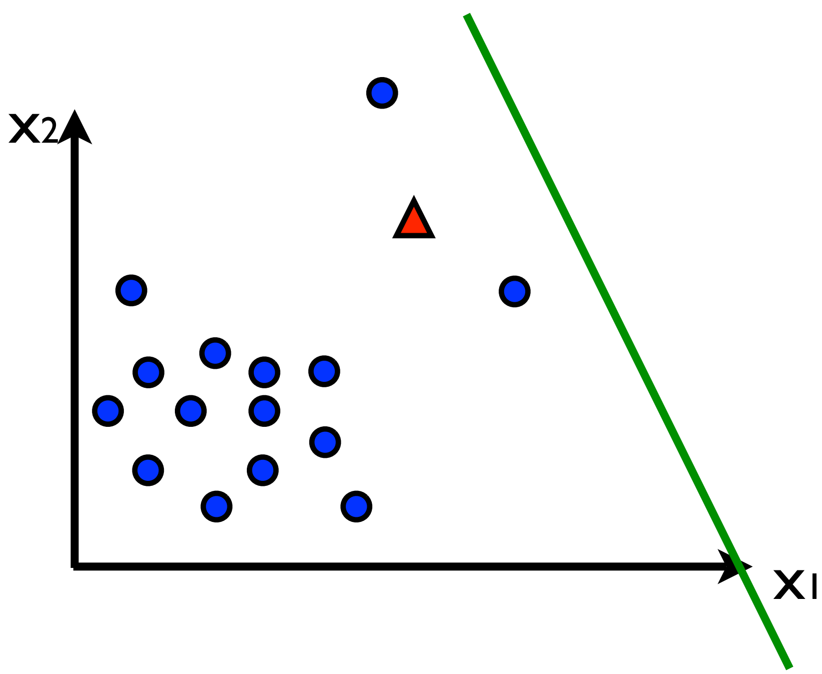
Balanced accuracy = 50%
Balanced error = 50%

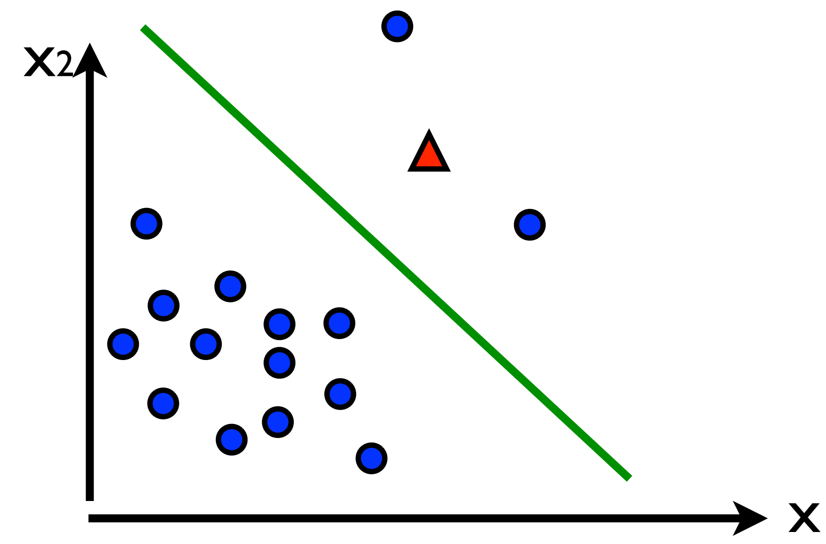
Not that good.

Confusion Matrix

		GROUND TRUTH (true class)							
		c1		**c2**		**....**	**cn**		
TEST OUTCOME	**c1**	N11	f11	N12	f12	…	N1n	f1n	
	c2	N21	f21	N22	f22	…	N2n	f2n	
	…	…	…	…	…	………	…	…	
	cn	Nn1	fn1	Nn2	fn2	…	Nnn	fnn	
									Accuracy

Confusion Matrix

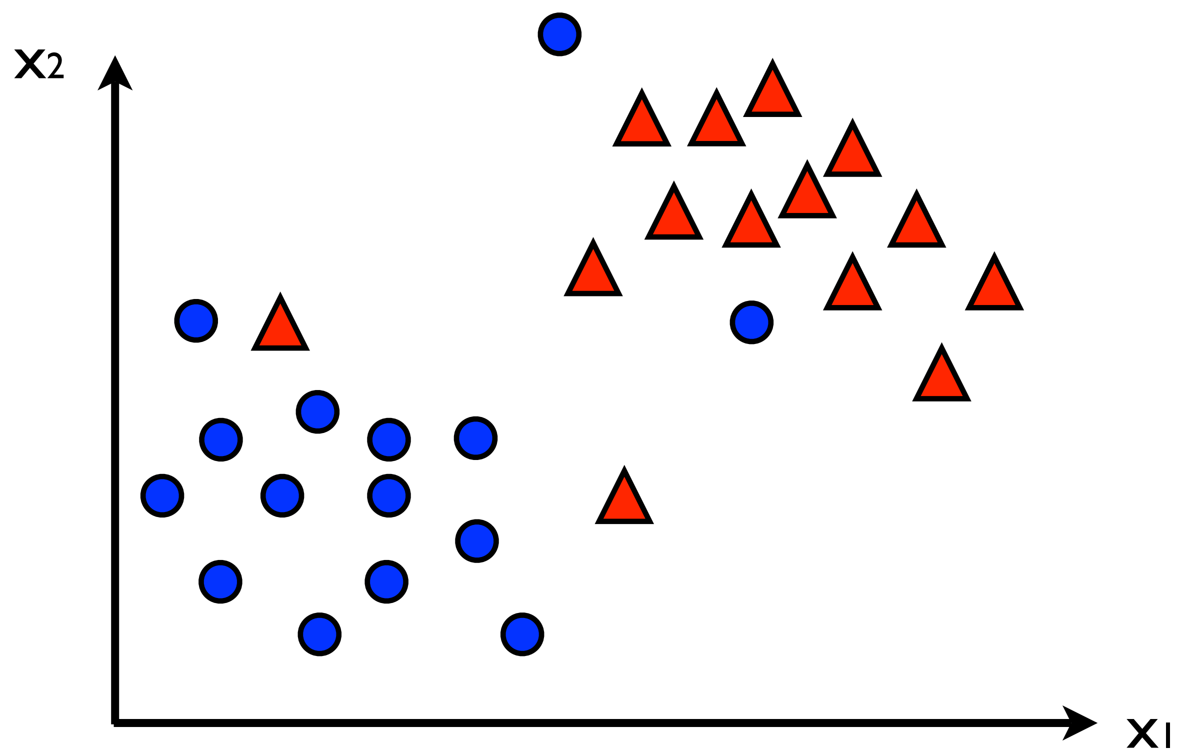




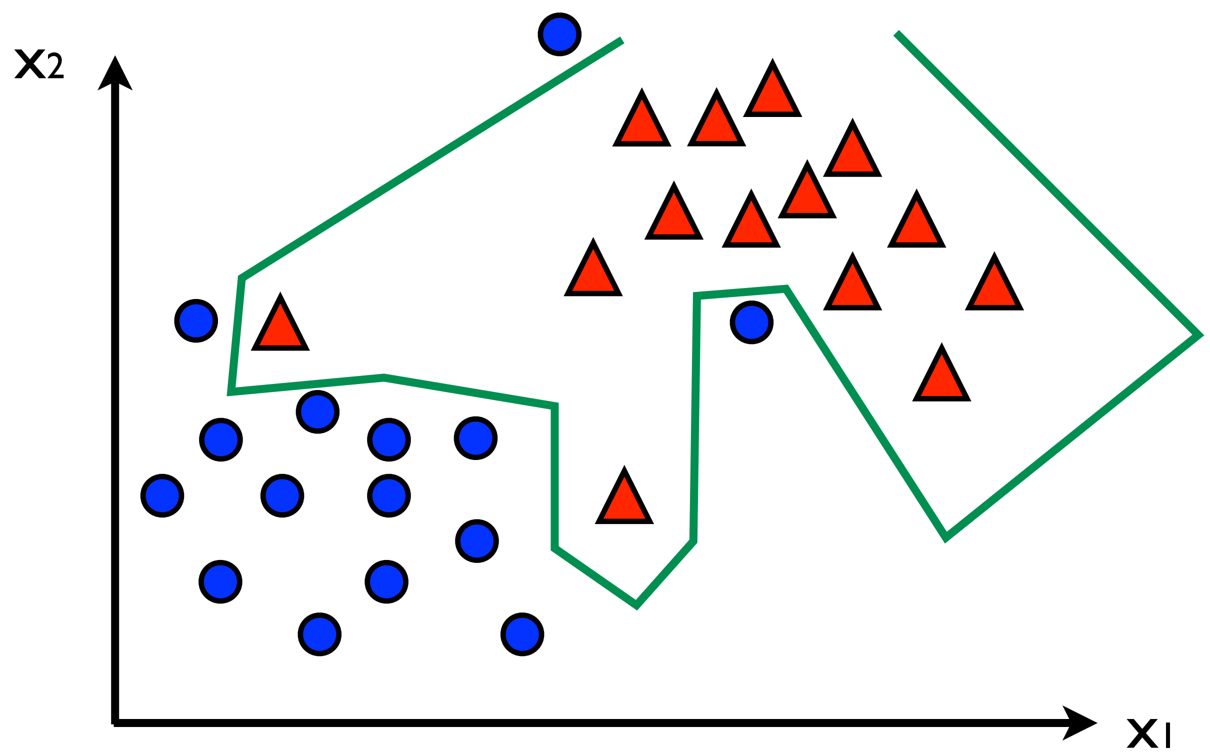
		GROUND TRUTH					
		●		▲			
TEST OUTCOME	●	15	100%	1	100%		
	▲	0	0%	0	0%	Acc	Bal Acc
						94%	47%

		GROUND TRUTH					
		●		▲			
TEST OUTCOME	●	13	87%	0	0%		
	▲	2	13%	1	100%	Acc	Bal Acc
						88%	91%

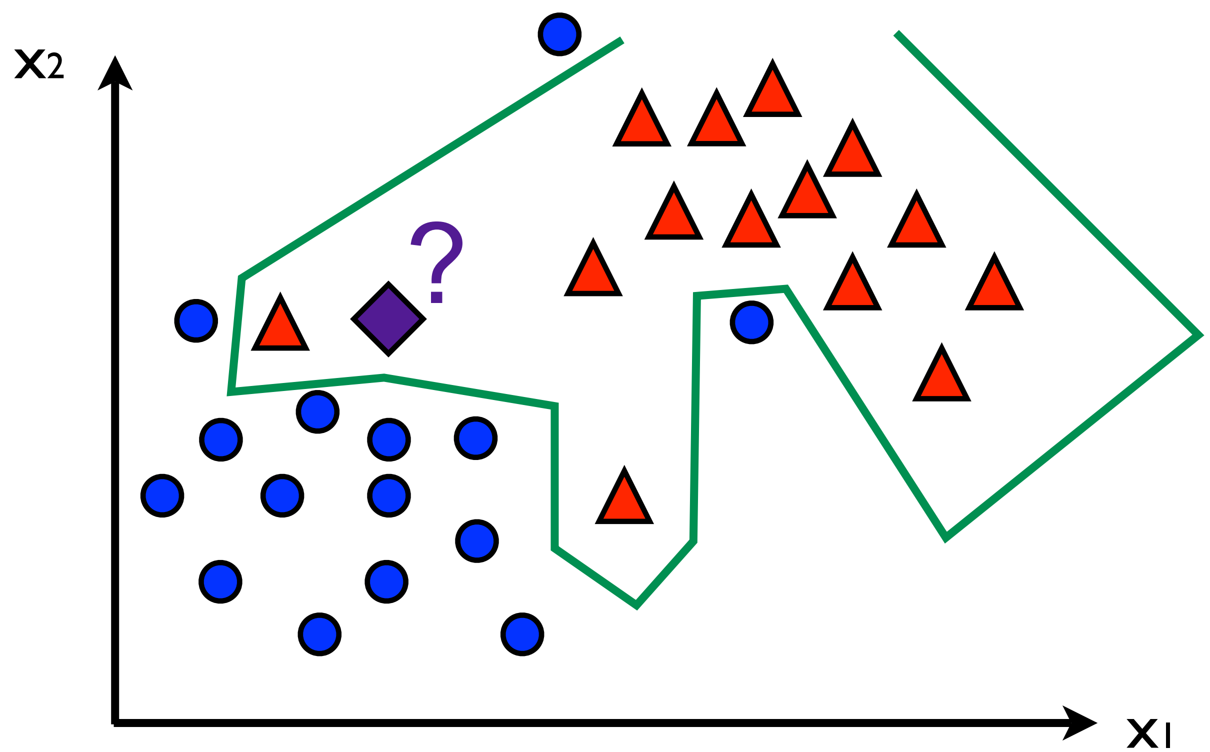
Overfitting



Overfitting



Overfitting



Overfitting

- TEST YOUR CLASSIFICATION MODEL OVER UNSEEN DATA!!

- Holdout Method
- Random subsampling
- Cross-validation
- Bootstrap

Holdout Method

- Separate the training set into two disjoint sets.
- Train over one set and validate over the other set.
- Usually 2/3 for training, 1/3 for testing.
- Calculate accuracy and confussion matrix over the test set.

Random subsampling

- Randomly divide the data into train and test sets.
- Perform the holdout method for each division.
- Accuracy is calculated as the average of the accuracies obtained.

k fold cross-validation

- Divide the dataset into k disjoint subsets (k-fold cross-validation).
- Leave one set for testing and use the other k-1 for training.
- Repeat k times using each subset once for validation.

Bootstrap

- Generate m subsets of size n’< n, sampling from D randomly with replacement.
- Records not included in the training set become part of the test set.
- On average, a bootstrap training set of size n contains 63.2% of the records in the original data.

Supervised Learning (Classification)

- Given a **training set** $\{<x_i, y_i>\}$
  - x_i: **attributes,** y_i: **classes**
- Determine a learning function $f : X \rightarrow Y$
- Goal: predict class of a given set of attributes
  - $y = f(x)$
- Very important: a separate **testing set** is used to validate our classificator. **Cross-validation.**

To be continued...

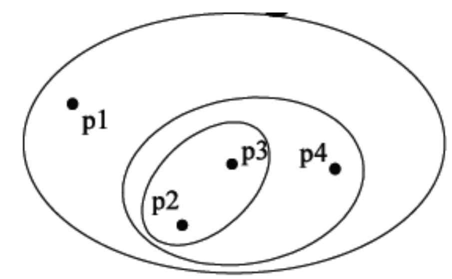
Unsupervised Learning

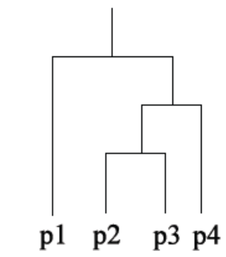
Two approaches

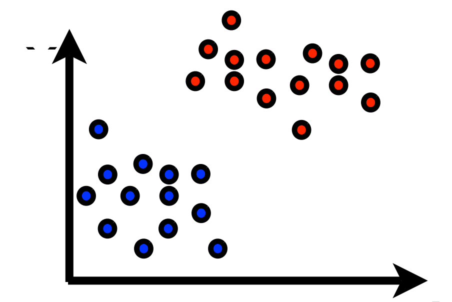
- Do we have some already labeled data?
- Yes: Supervised Learning
  - ANN, SVM, Decision Trees, Bayesian Classificators, Nearest Neighbours, etc...
- No: Unsupervised Learning
  - Clustering: K-Means, Hierarchical Clustering, DBSCAN, etc...

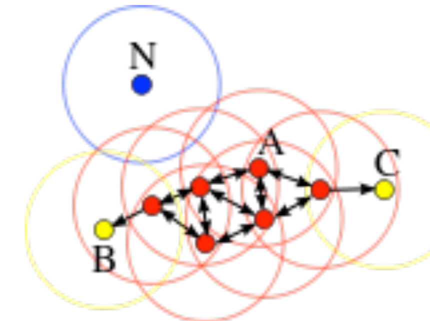
Unsupervised Learning

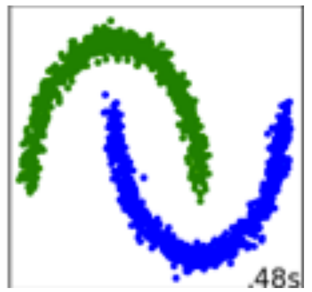
- Find structure in un-labeled data
  - Clustering (partitioning / hierarchical)
    - k-means
    - DBSCAN
  - Density Estimation
    - Histograms
    - Kernel Density Estimation
    - Gaussian Mixture Models

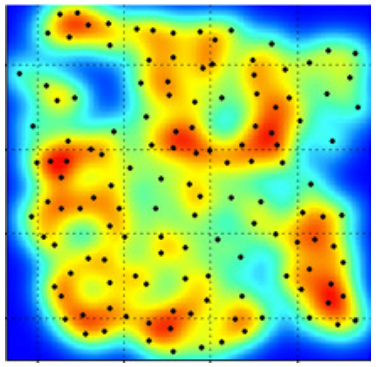


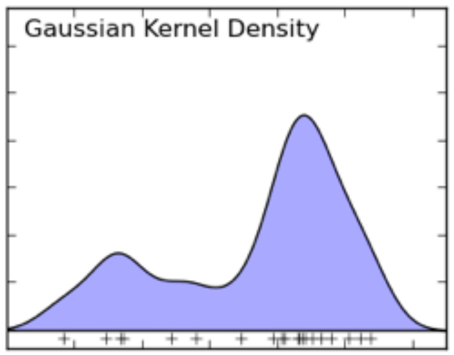






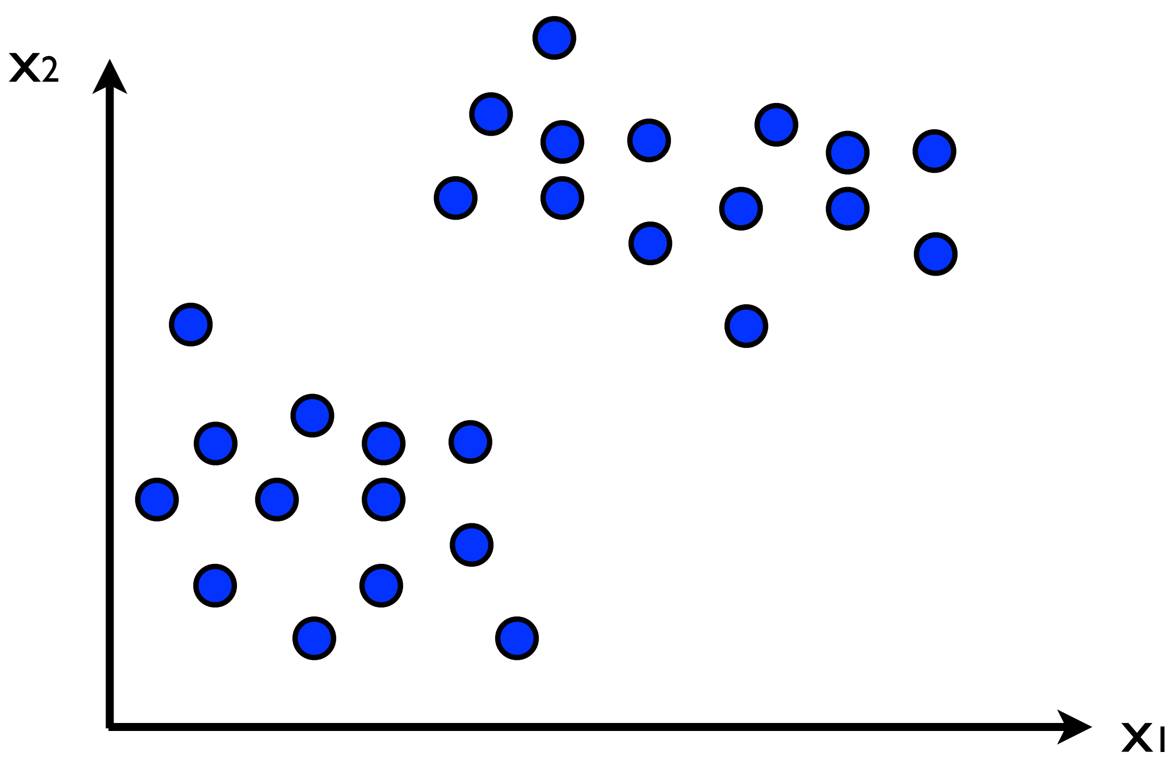






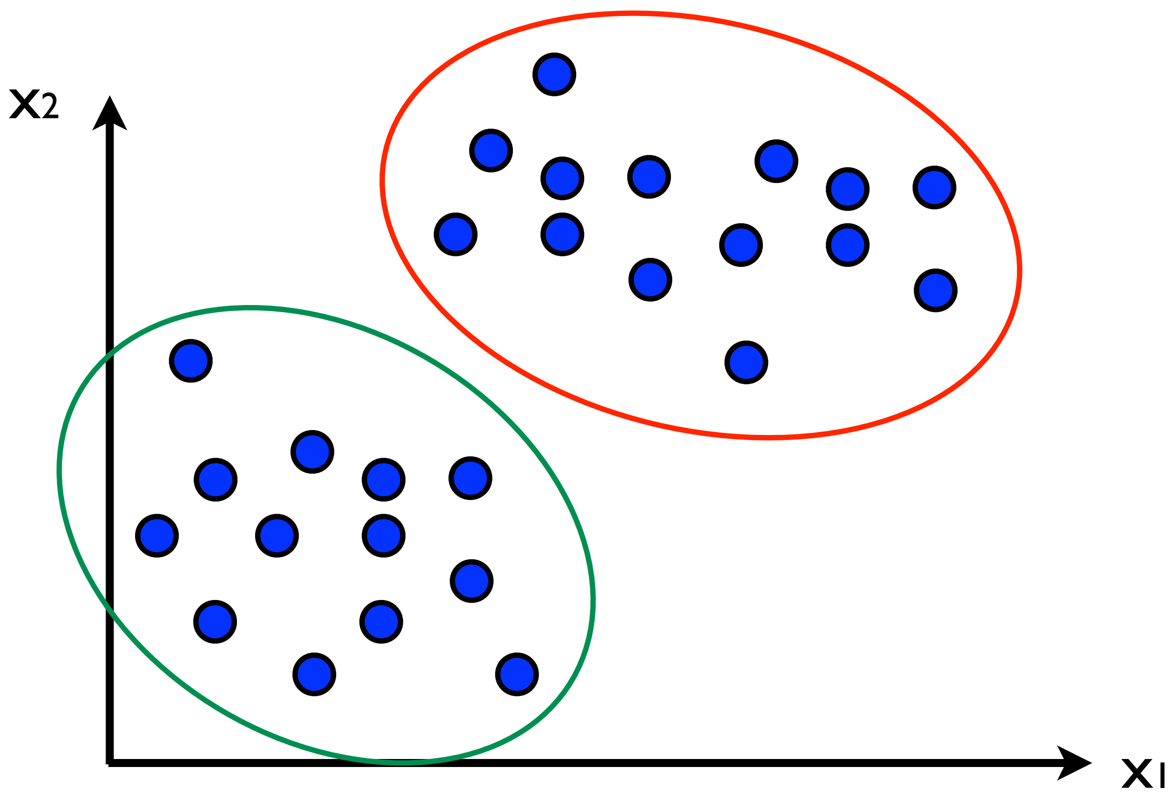
Unsupervised Learning: Clustering

- Find sets of objects such that objects inside each set are similar to each other (or related), and are different (or not related) to objects from other groups.



Unsupervised Learning: Clustering

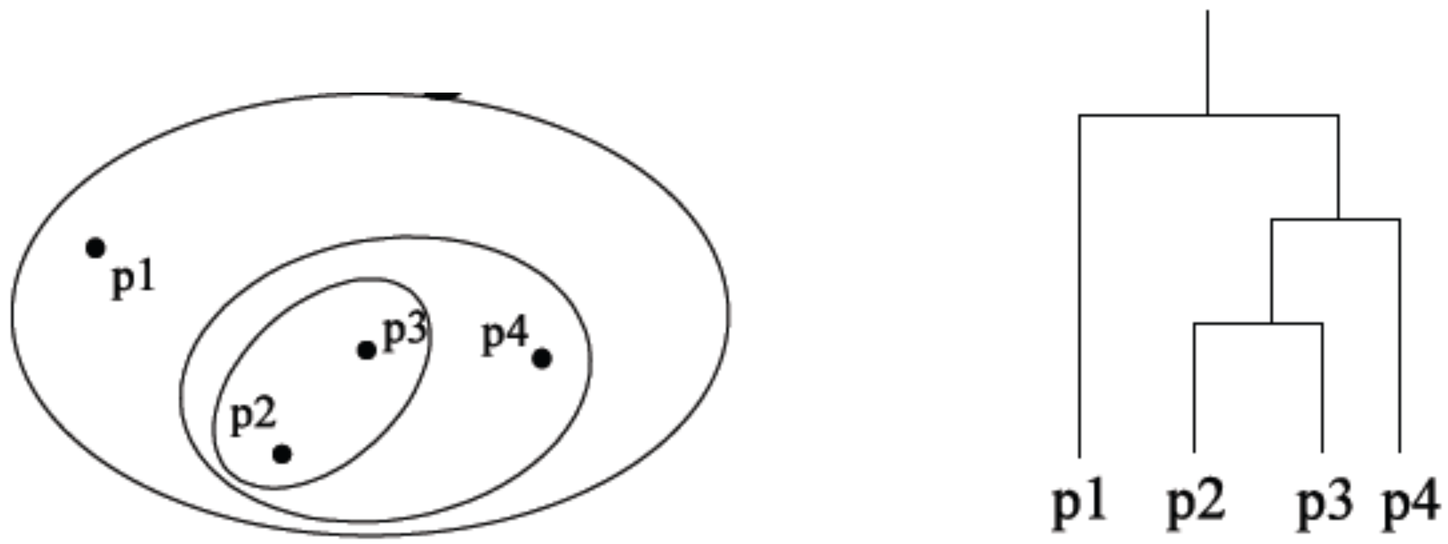
- Find sets of objects such that objects inside each set are similar to each other (or related), and are different (or not related) to objects from other groups.



Clustering

- partitioning clustering

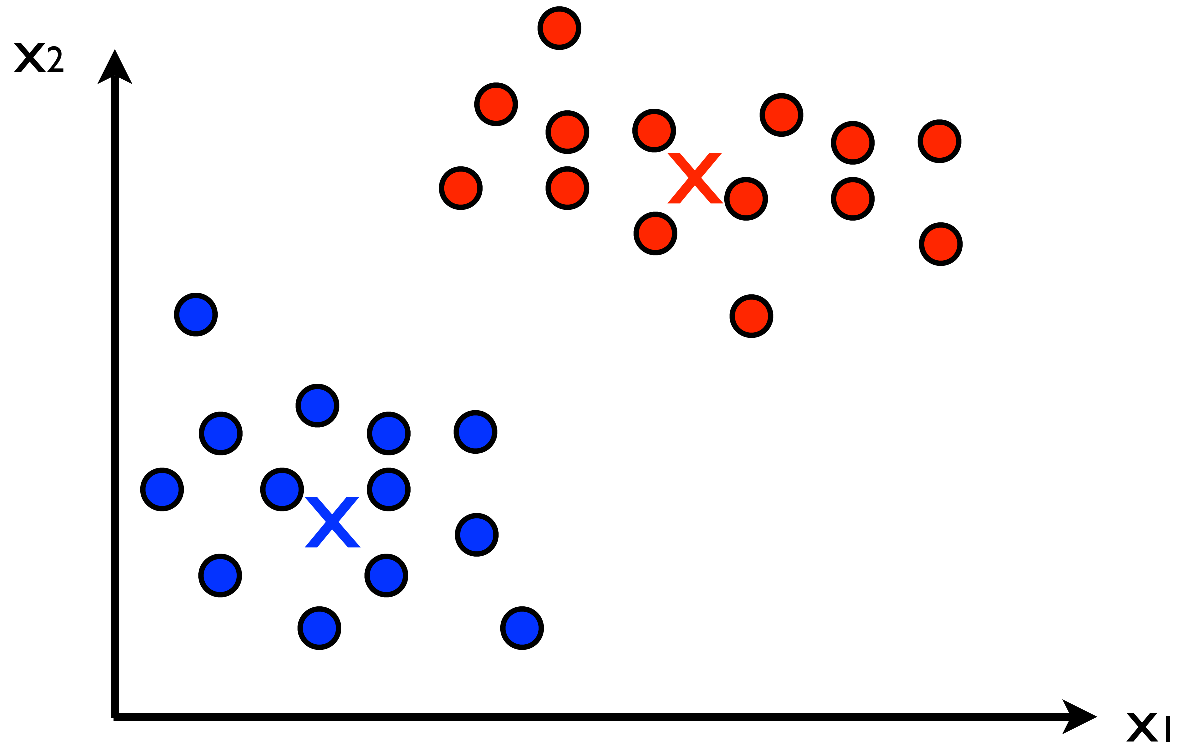
- hierarchical clustering



figures from B. Bustos

k-means

- Divide objects into k clusters.
- Each cluster is described by a centroid.
- Each object is associated to the closest centroid.

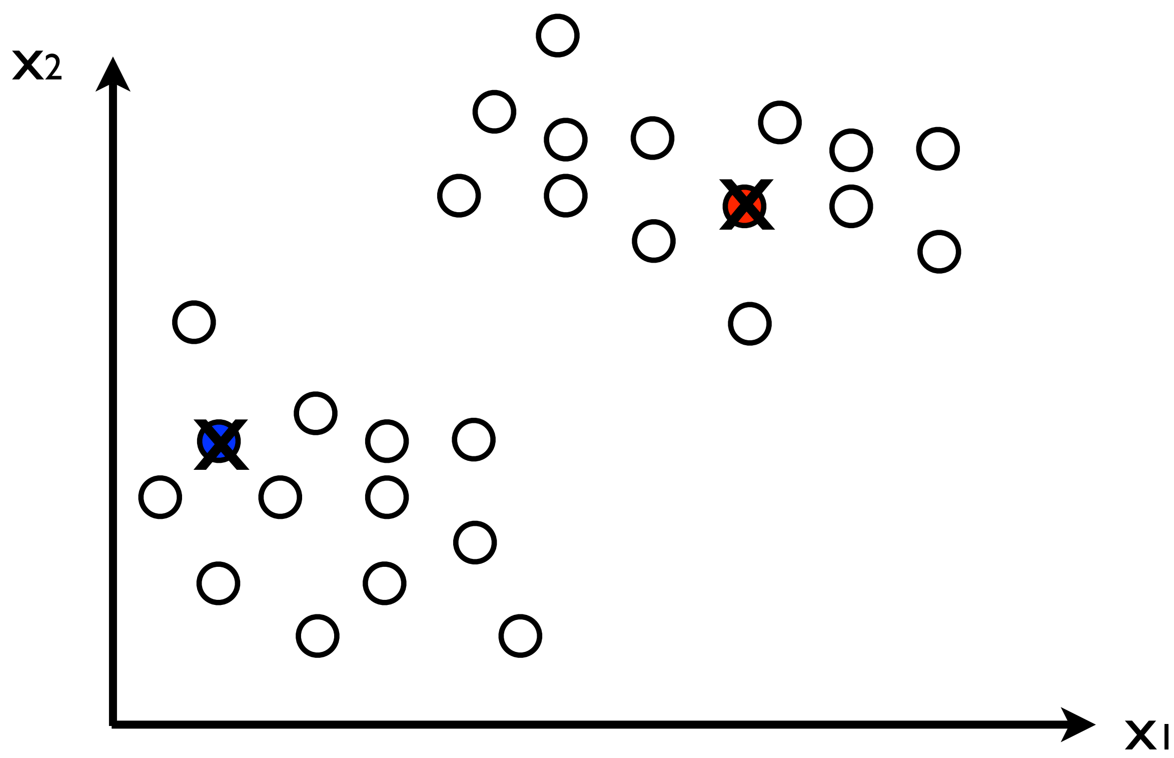


k-means: how do we define the centroids?

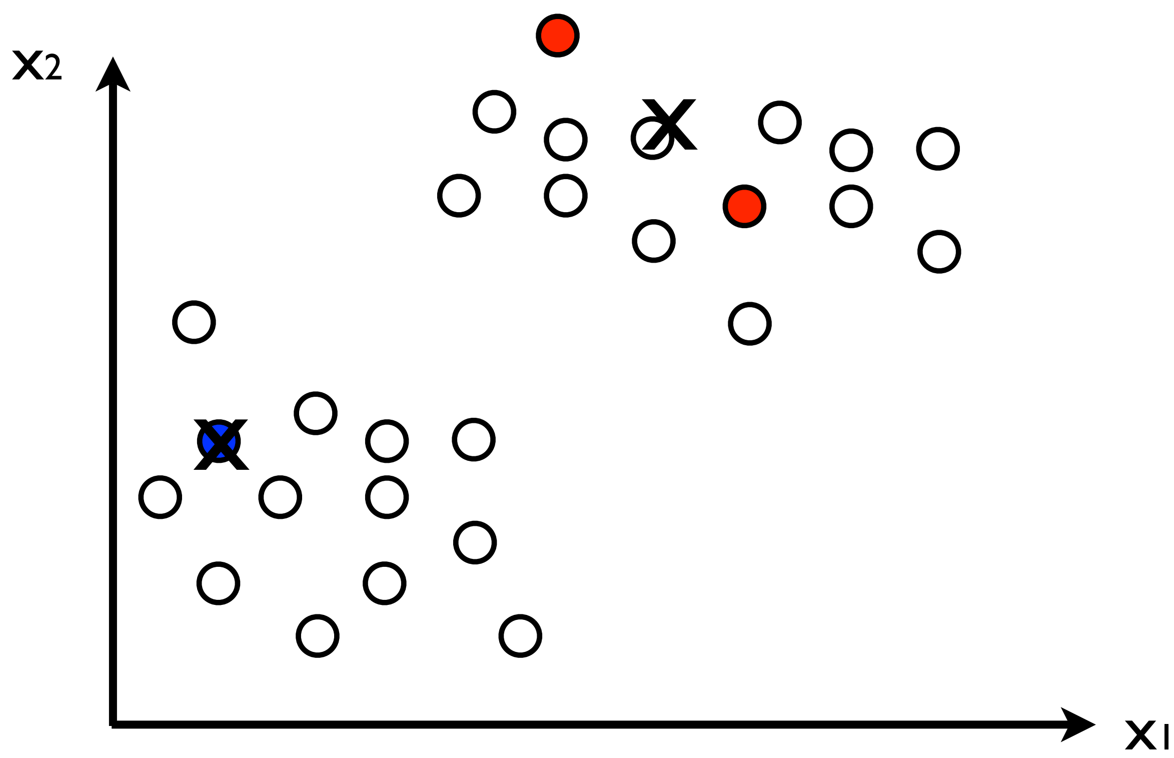
BASIC ALGORITHM

- Choose initial centroids
  - randomly
  - clusters depend on the initial centroids
- Randomly pick a new object and associate it to the closest centroid.
- Centroids are re-defined as the mean of the objects in the cluster.
- Convergence is achieved after I iterations (when clusters dont change much).

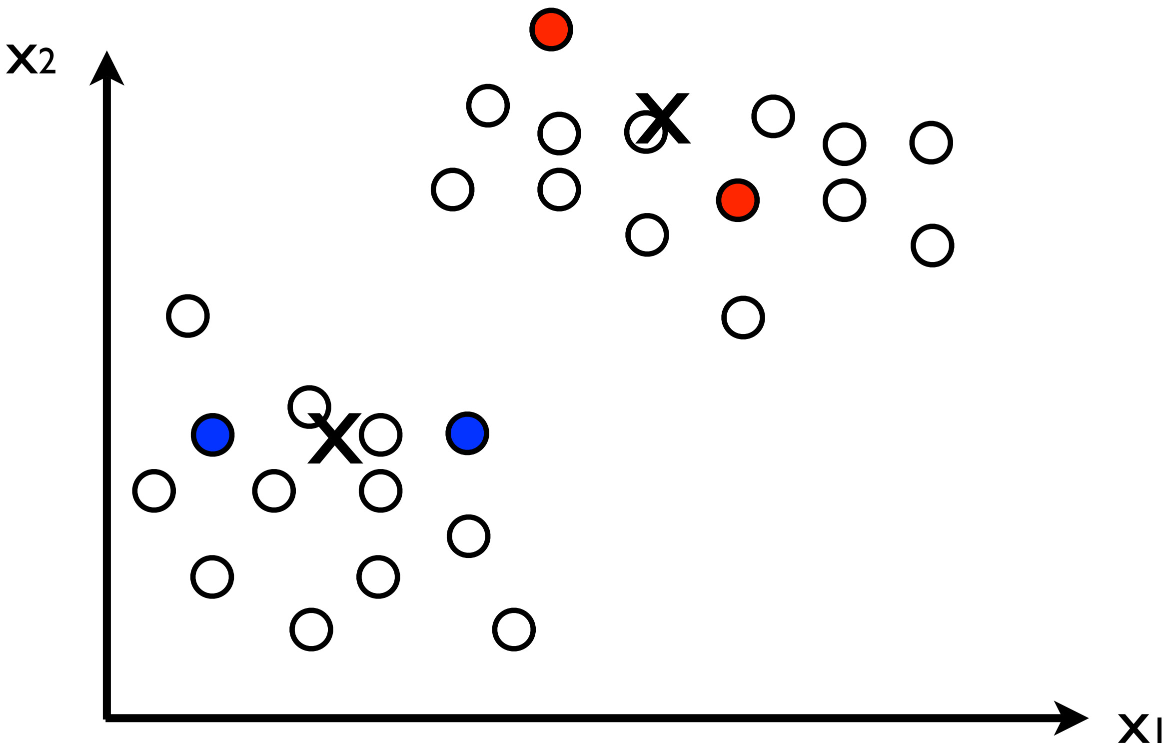
k-means



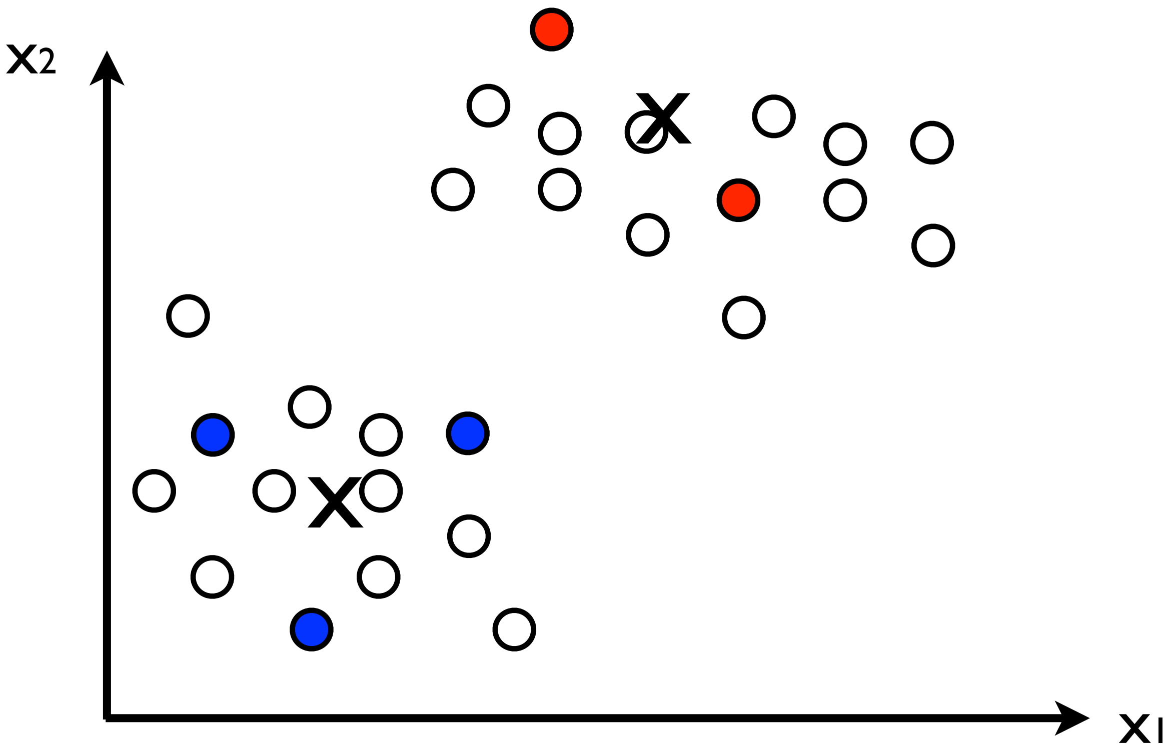
k-means



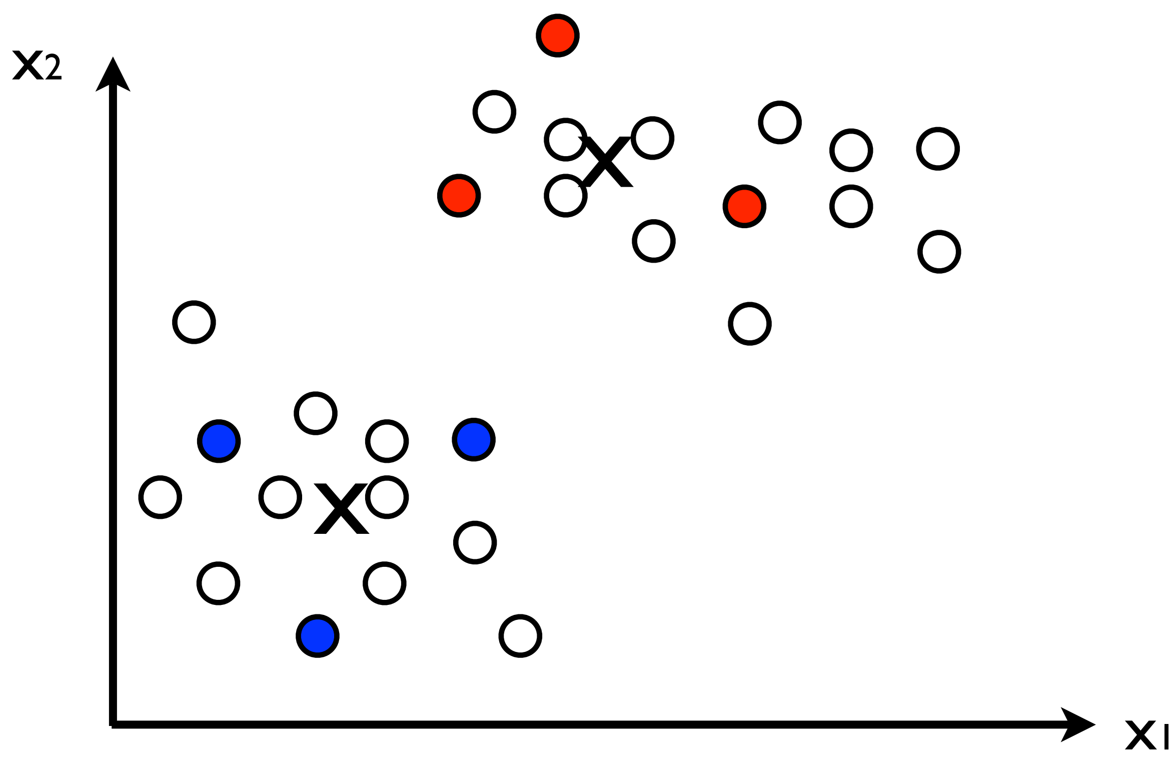
k-means



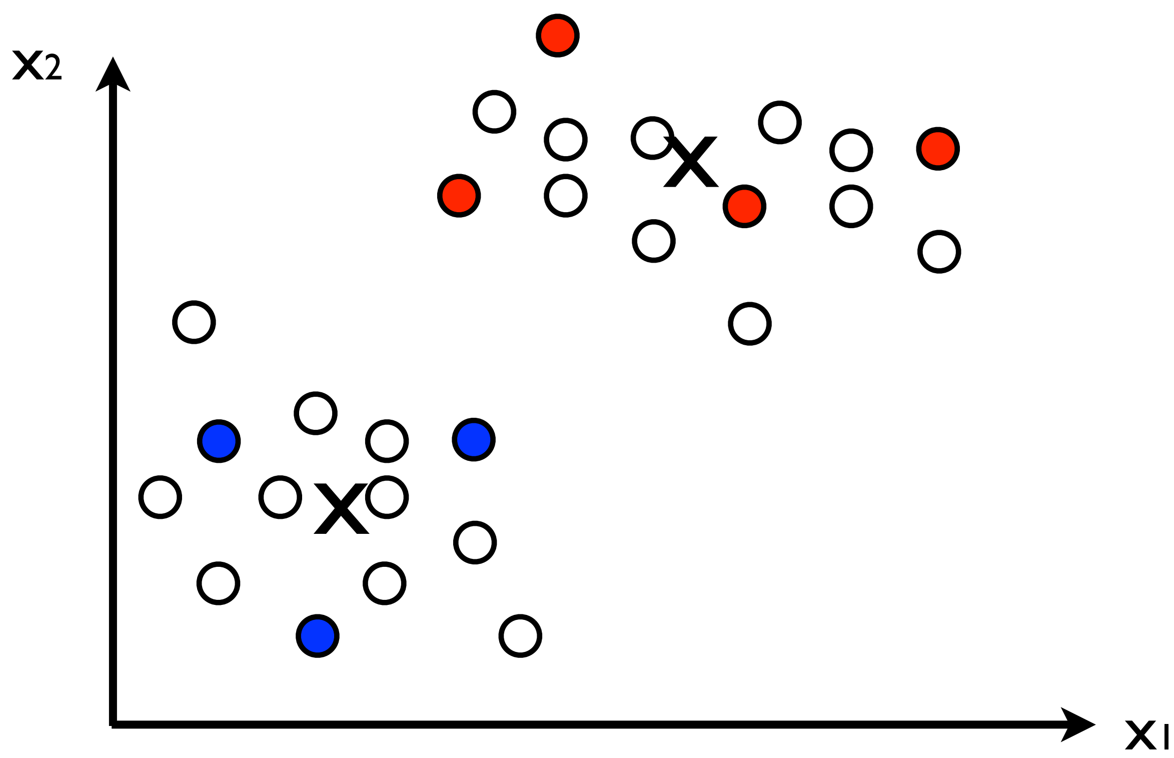
k-means



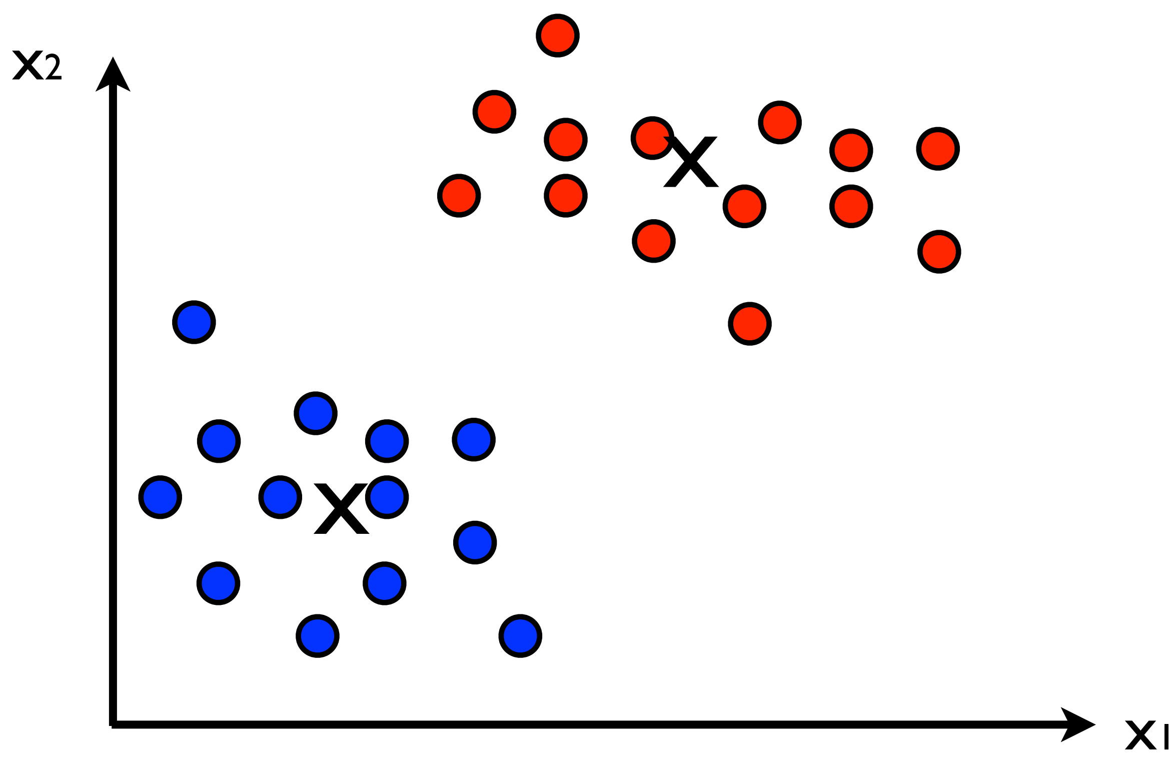
k-means



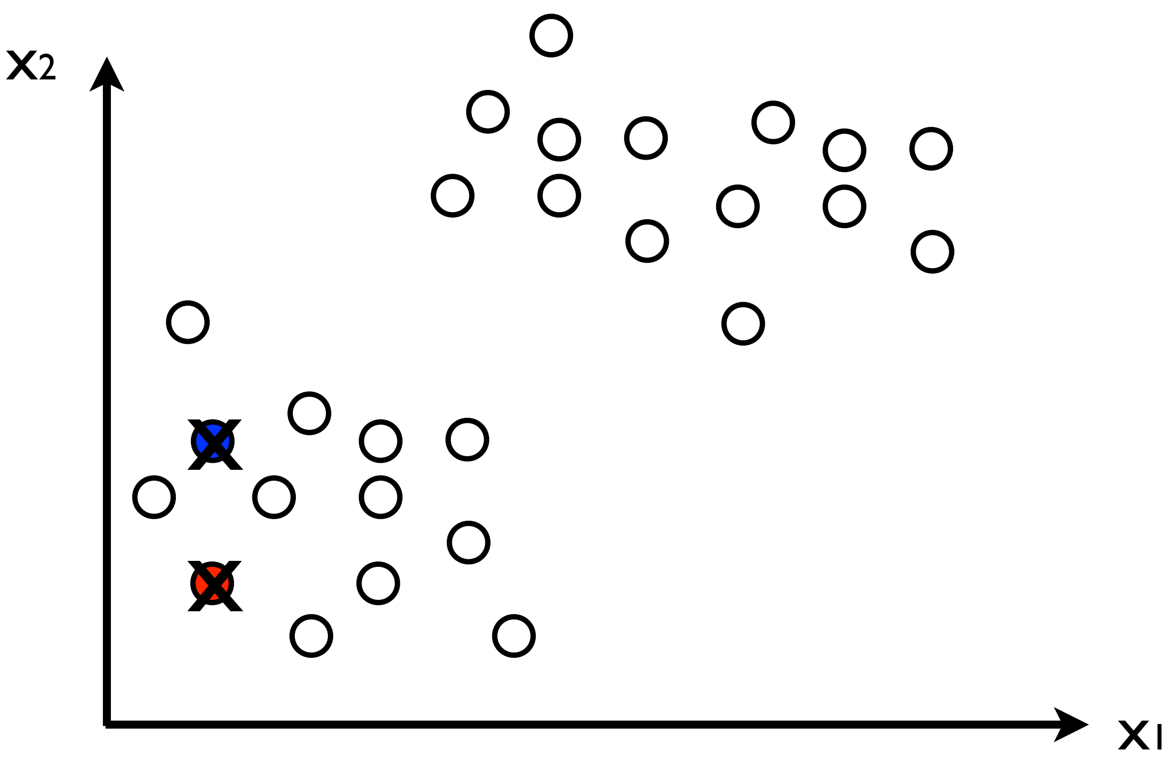
k-means



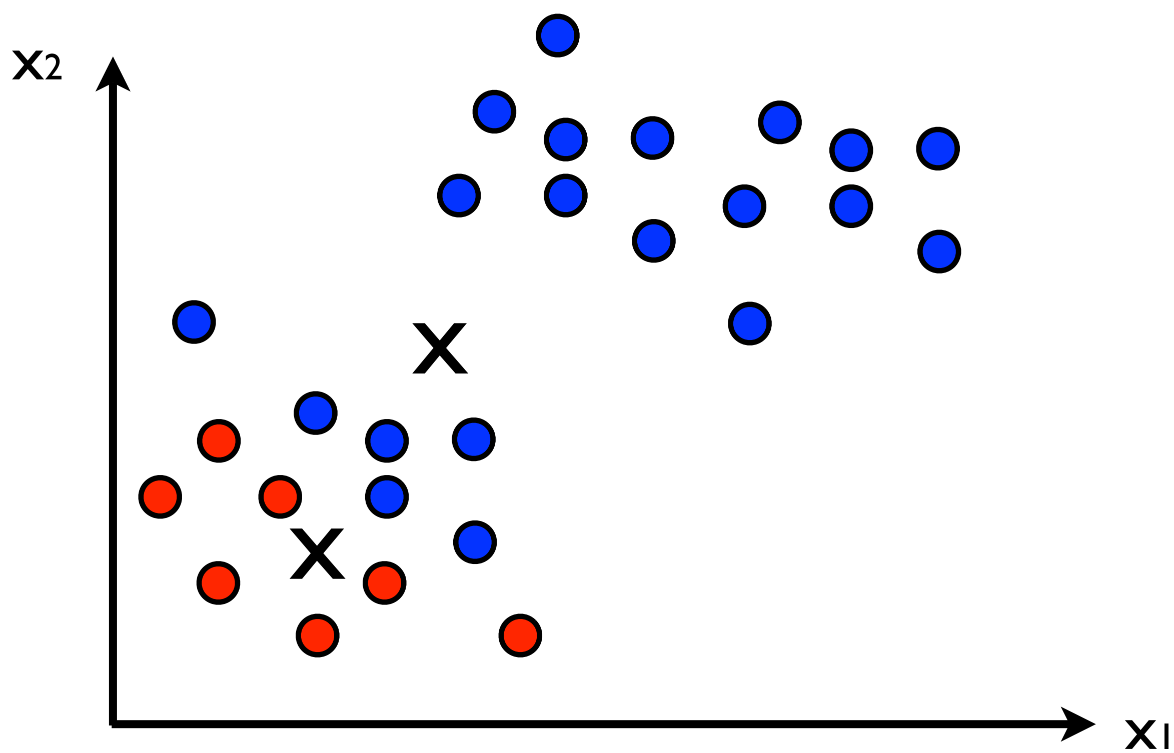
k-means



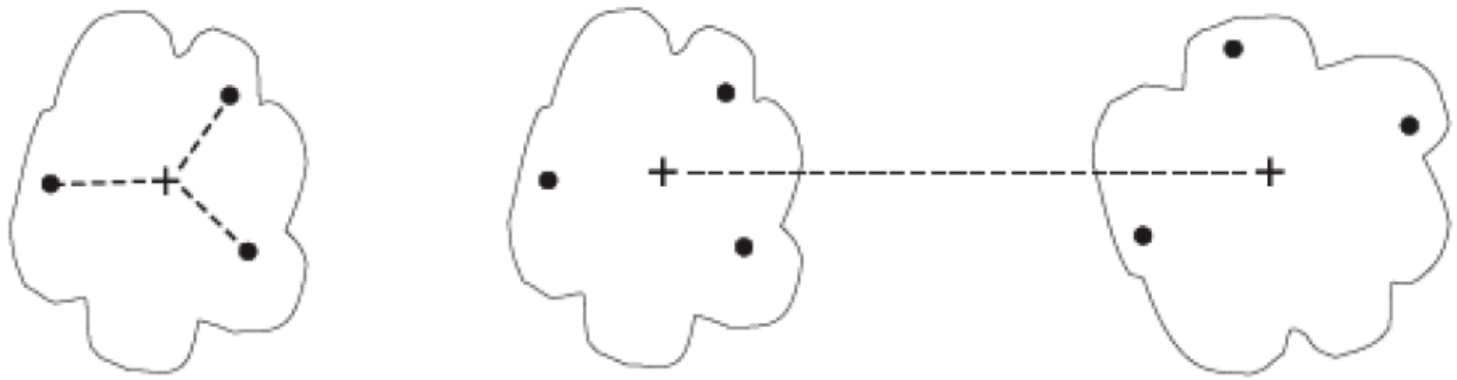
k-means: different initial centers



k-means: different initial centers



Evaluating Clusters



(a) Cohesion. (b) Separation.

- Cohesion: Sum of Squared Errors

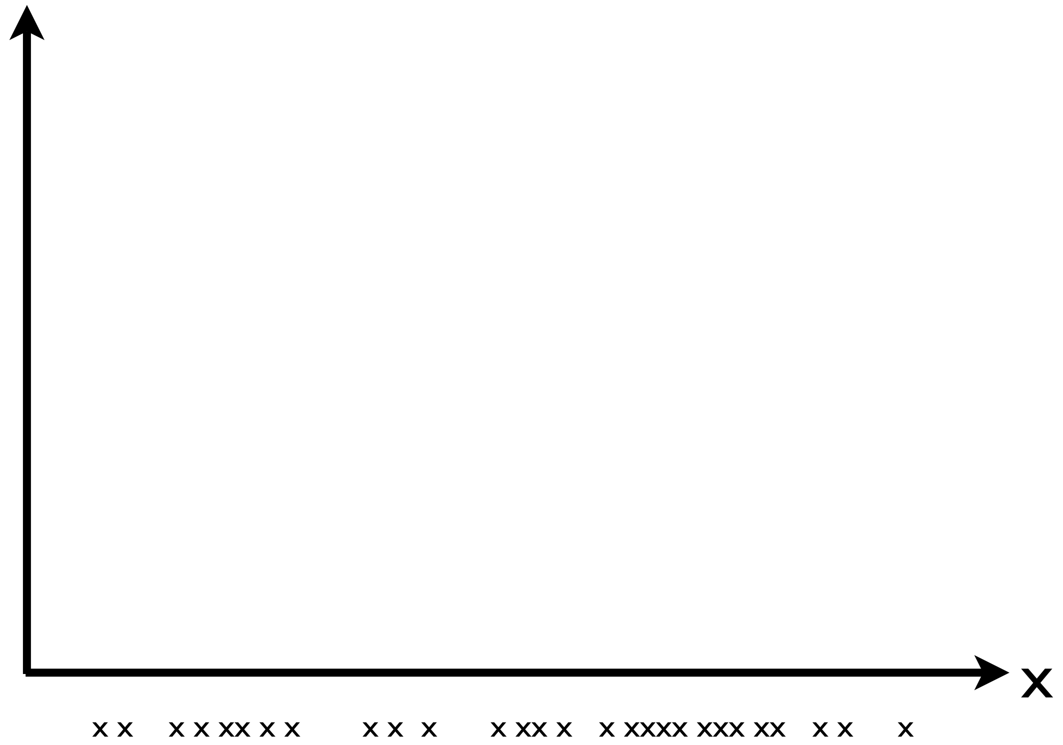
$$\mathrm{SSE} = \sum_{i=1}^{K} \sum_{\mathbf{x} \in C_i} dist(\mathbf{c}_i, \mathbf{x})^2 = \sum_{i=1}^{K} \sum_{x \in C_i} (c_i - x)^2$$

- Separation: Between Groups Sum of Squares

$$\mathrm{SSB} = \sum_{i=1}^{K} m_i \, dist(\mathbf{c}_i, \mathbf{c})^2$$

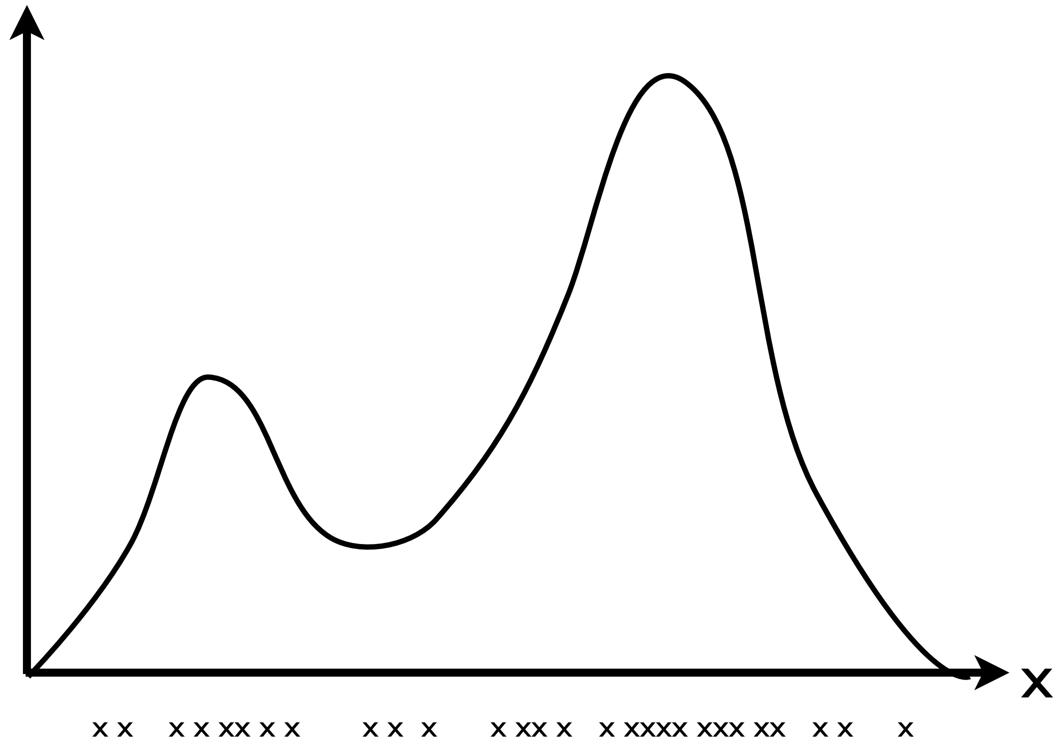
Density Estimation

- Building a probability density function by using data.



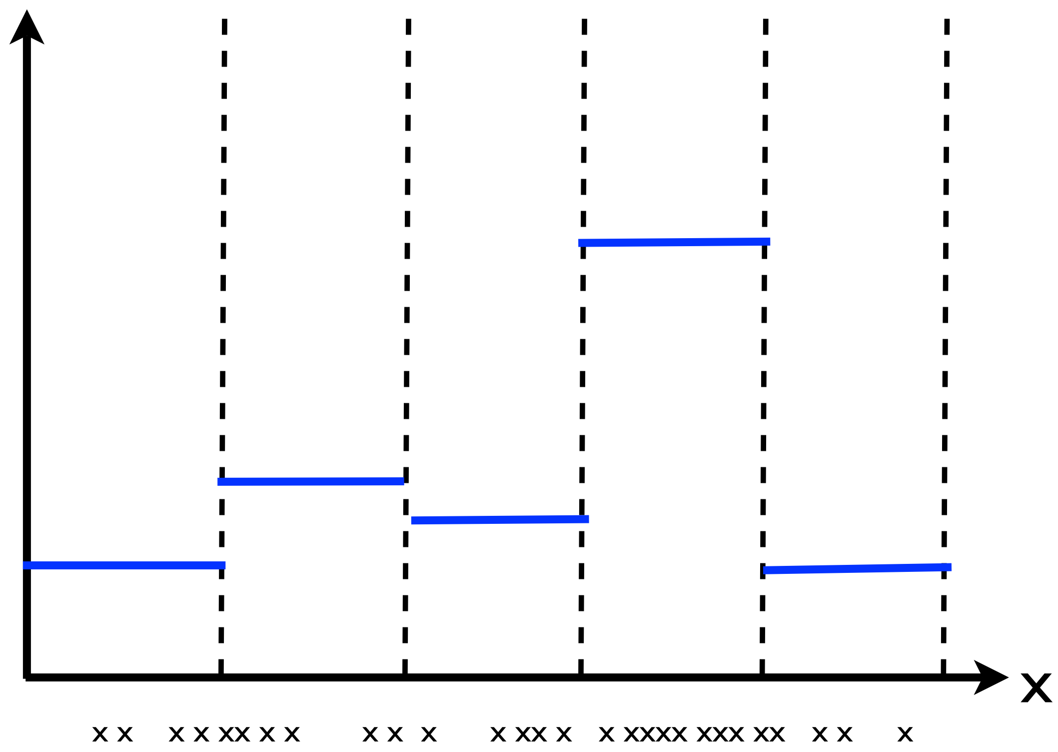
Density Estimation

- Building a probability density function by using data.



Histogram

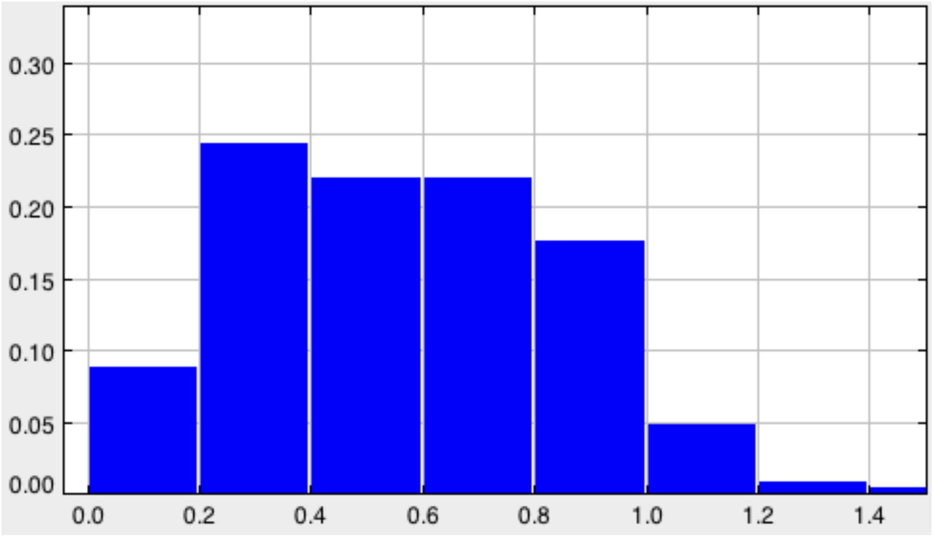
- Divide x in bins and count number of objects per bin.



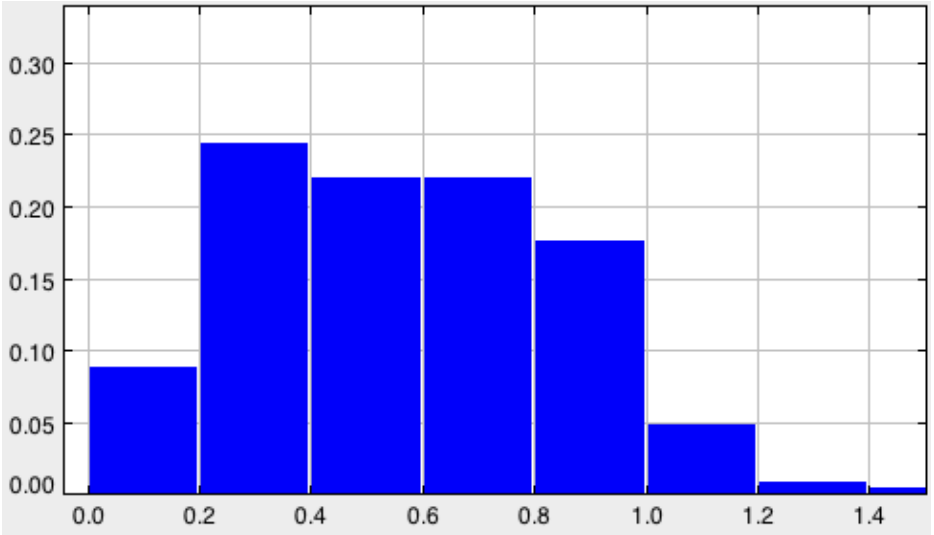
- Issues to consider: Number of points versus number ob bins.

Histogram

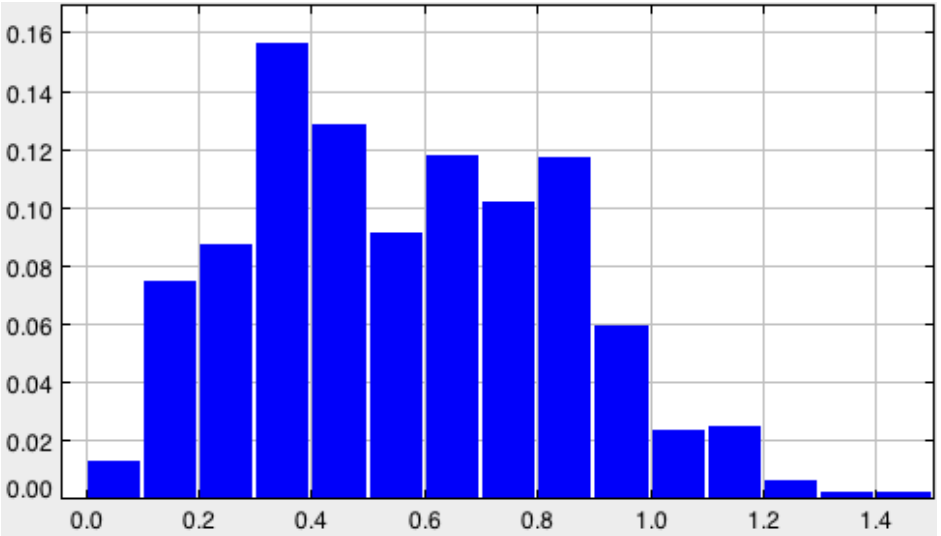
Histogram



Histogram

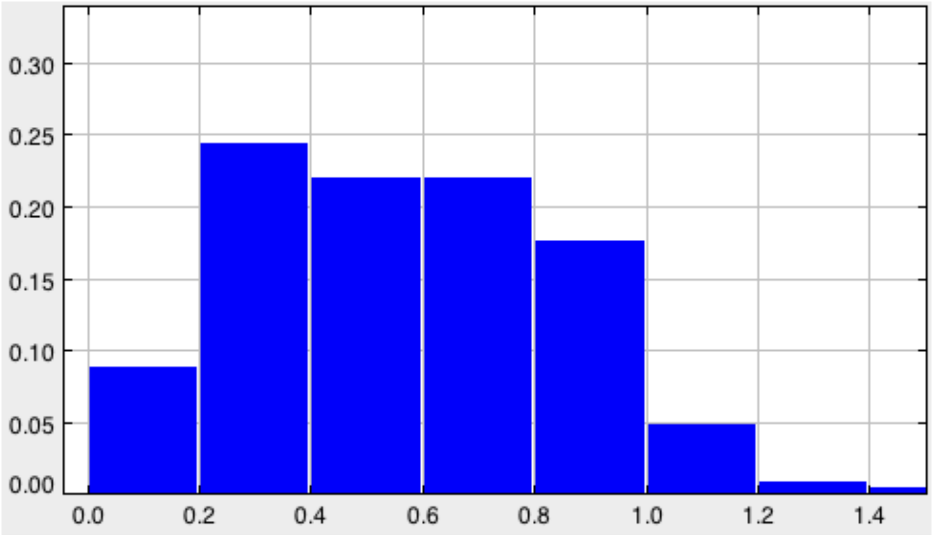


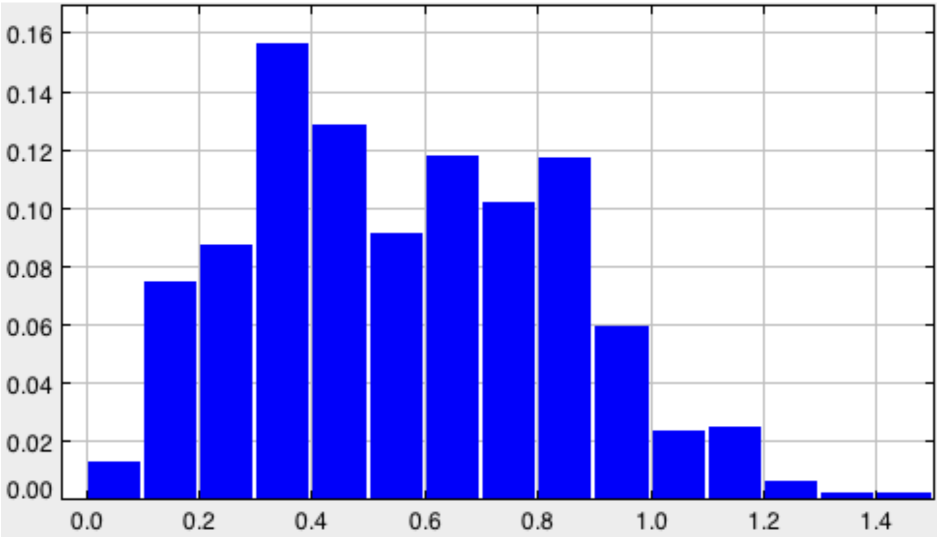
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4

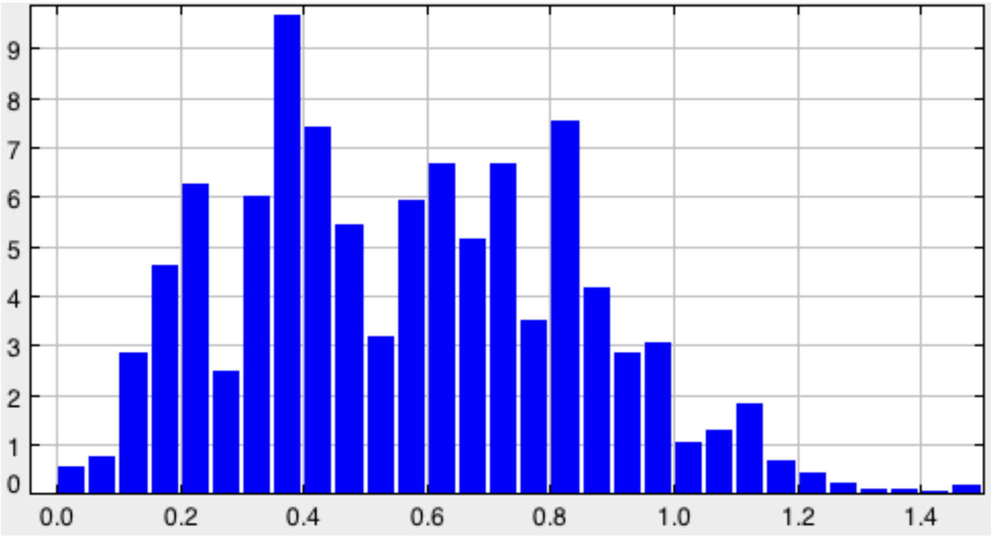


0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4

Histogram


0.30
0.25
0.20
0.15
0.10
0.05
0.00
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4


0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4


9
8
7
6
5
4
3
2
1
0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4

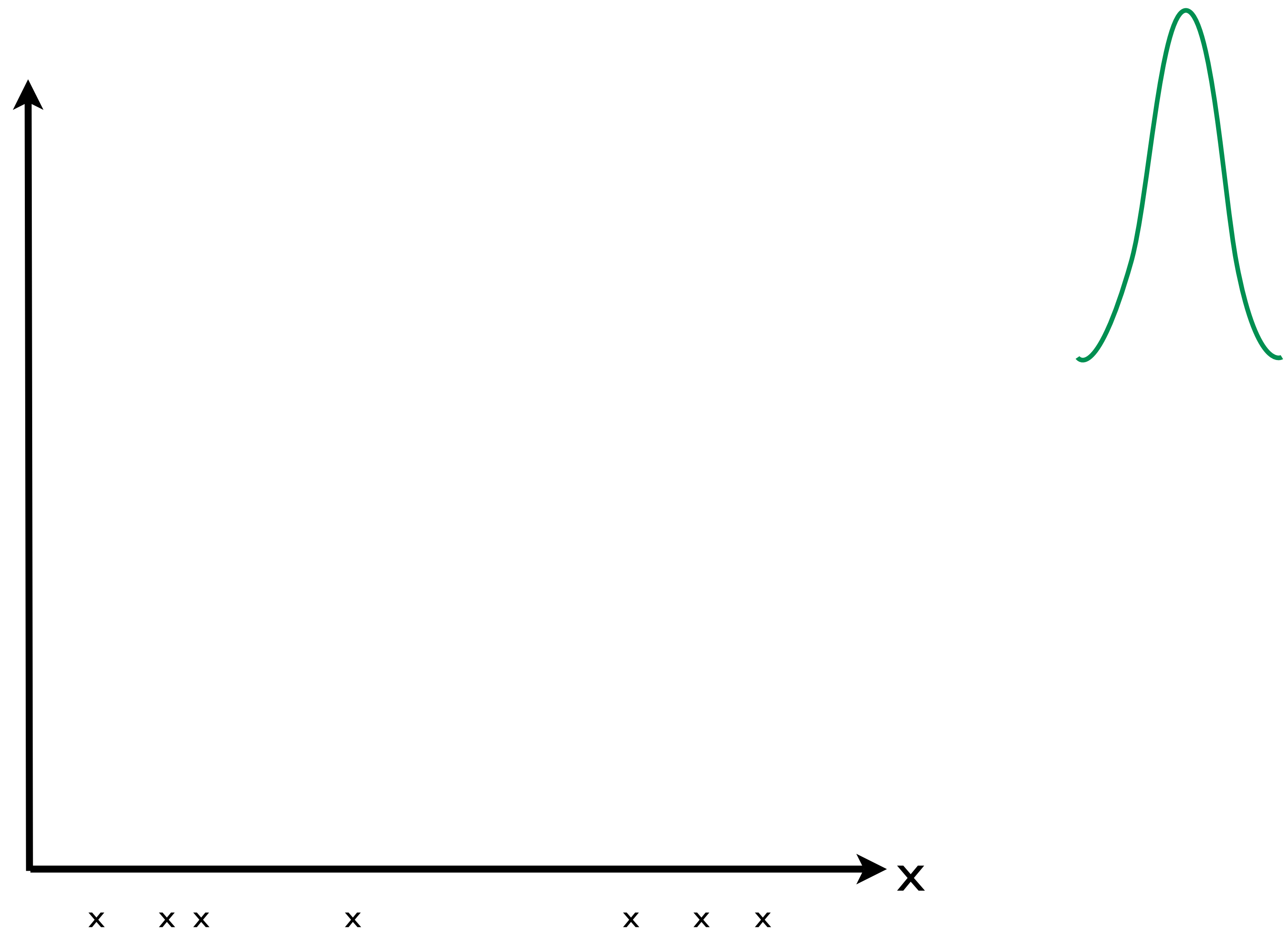
Kernel Density Estimation

- “Non-parametric” density estimation.
- Each data point is described by a kernel.
- The probability density function is estimated as the sum of the kernels

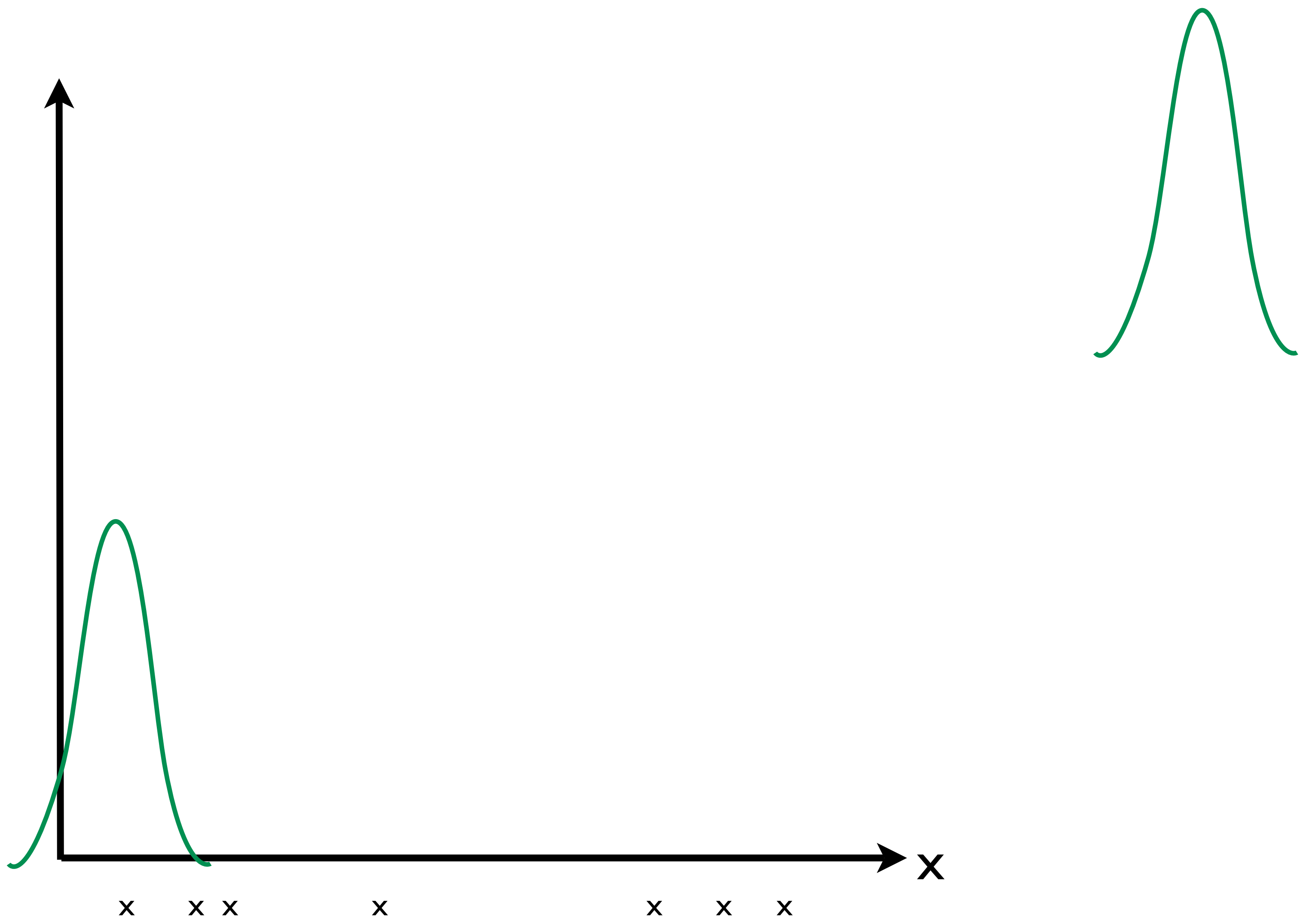
$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) \quad = \frac{1}{nh}\sum_{i=1}^{n} K\Big(\frac{x - x_i}{h}\Big)$$

- h is the badwidth parameter that defines the size of the Kernel.

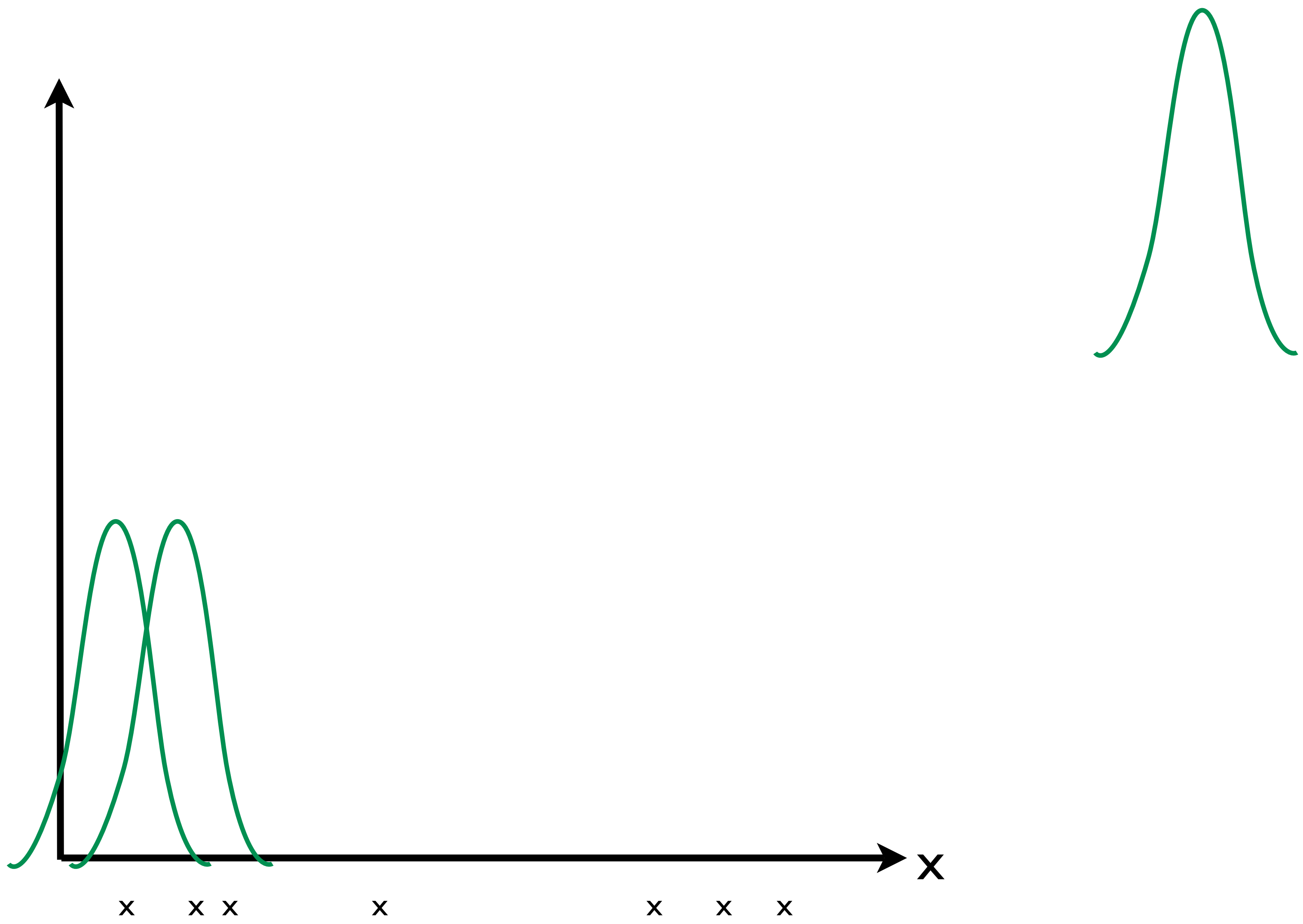
Kernel Density Estimation



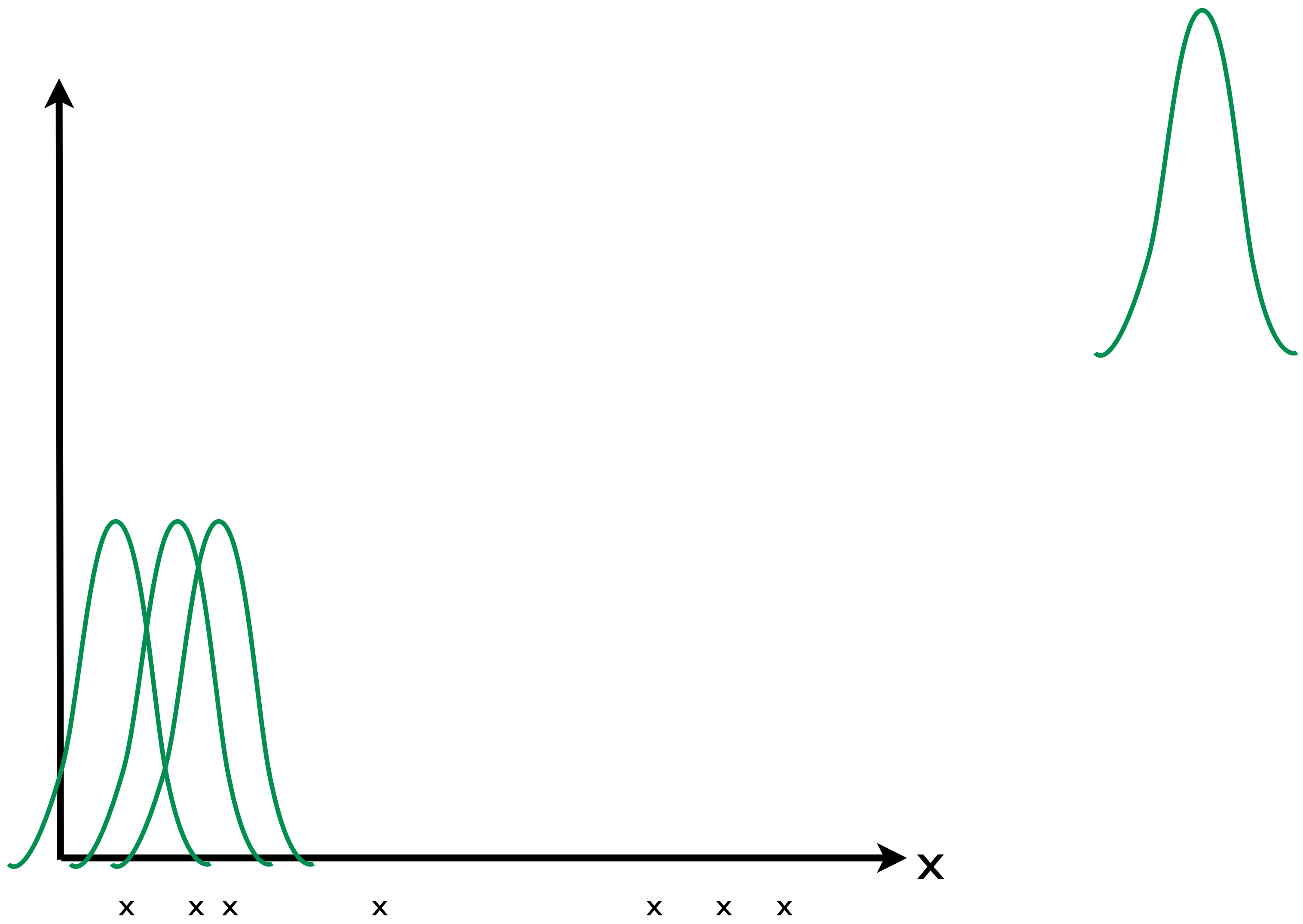
Kernel Density Estimation



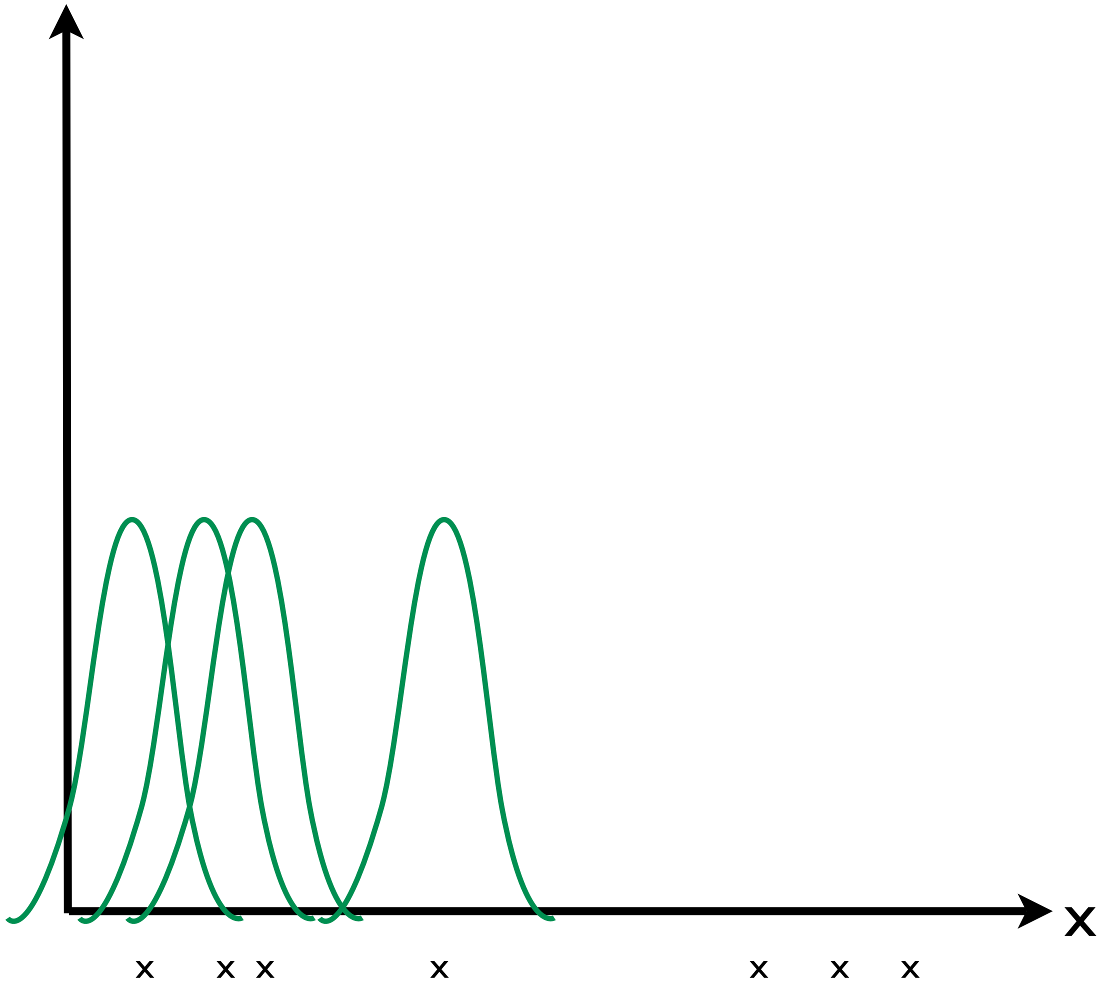
Kernel Density Estimation

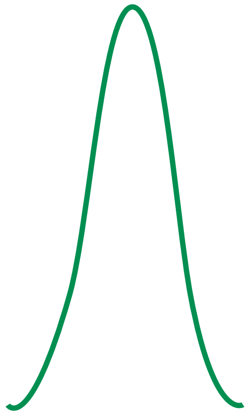


Kernel Density Estimation

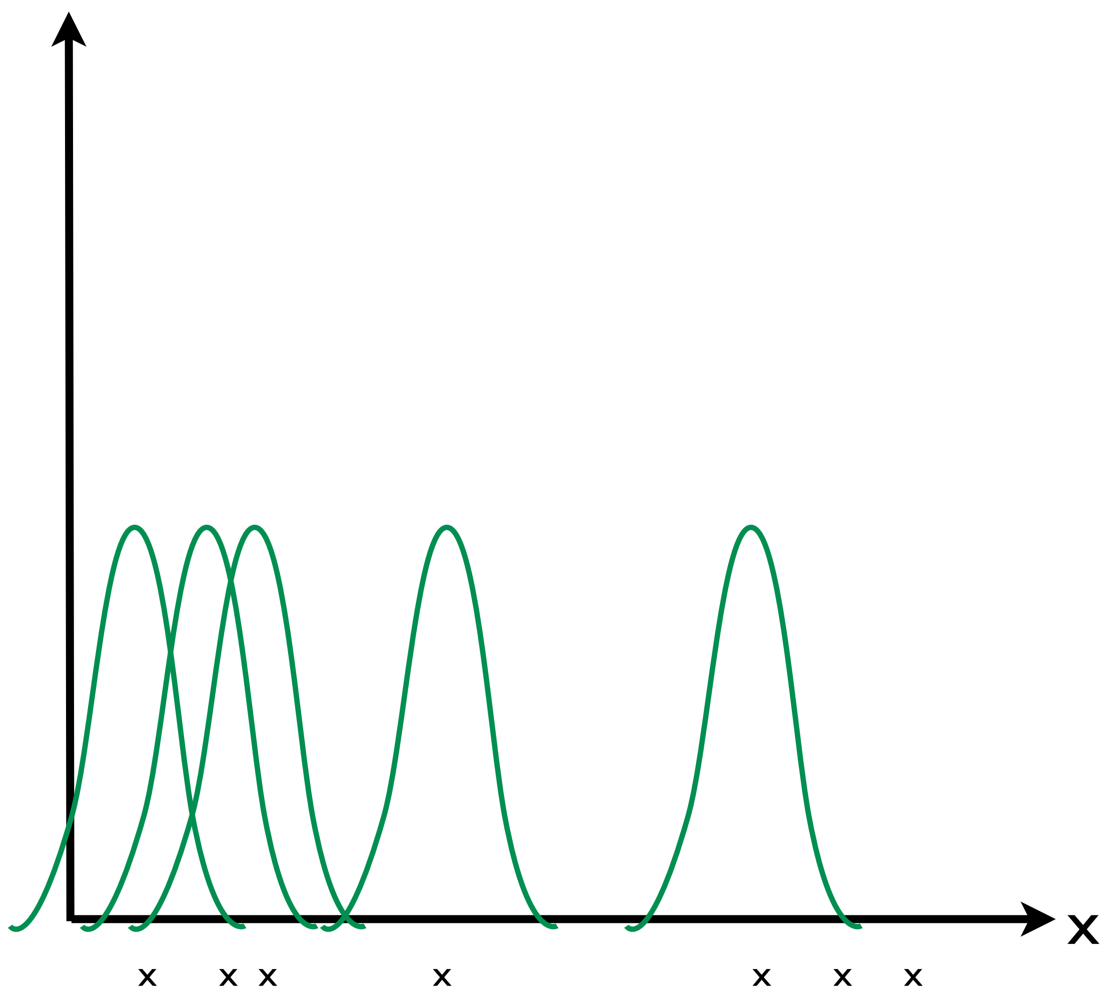


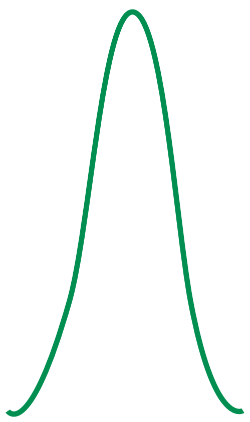
Kernel Density Estimation



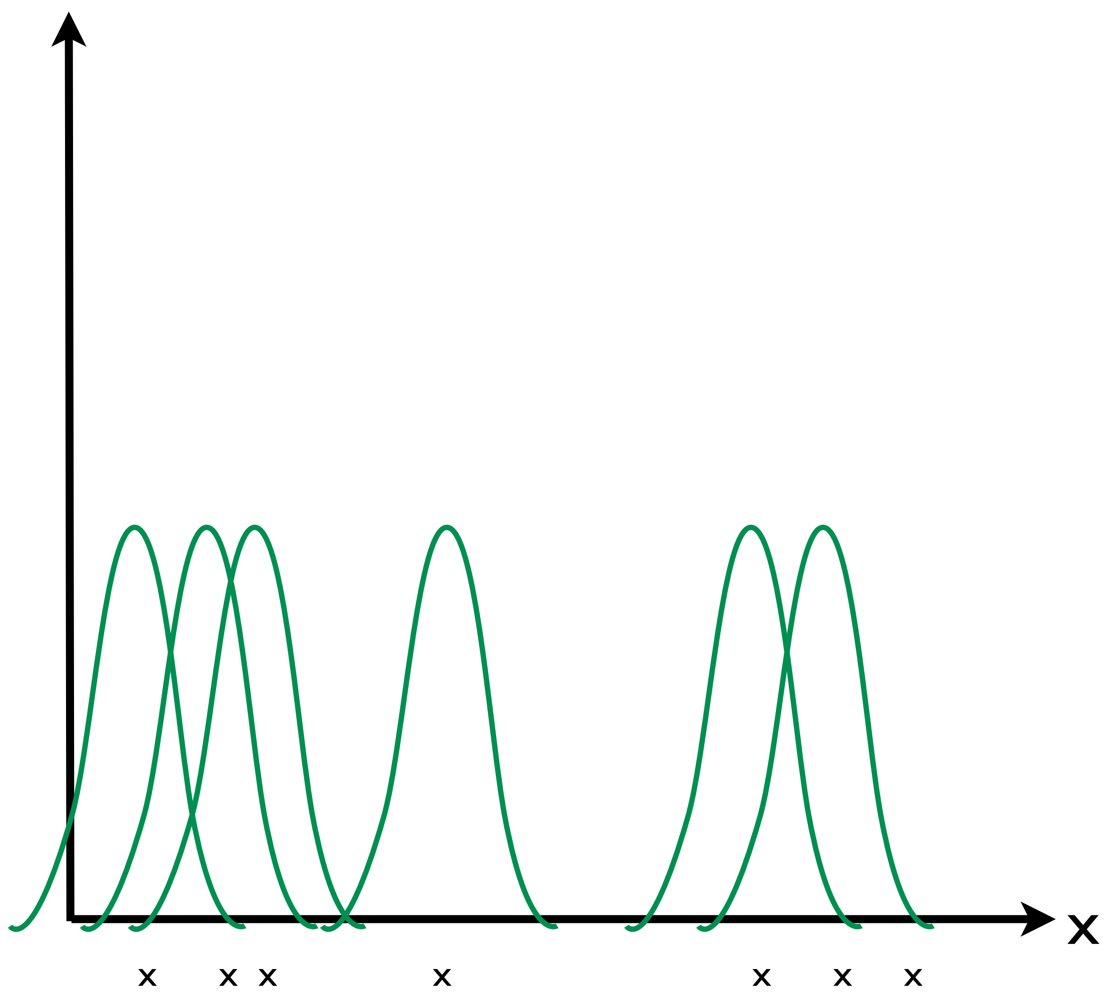


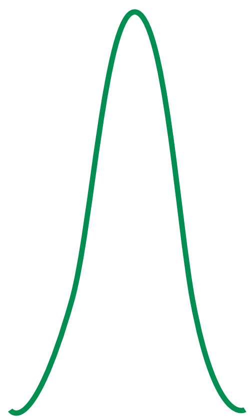
Kernel Density Estimation



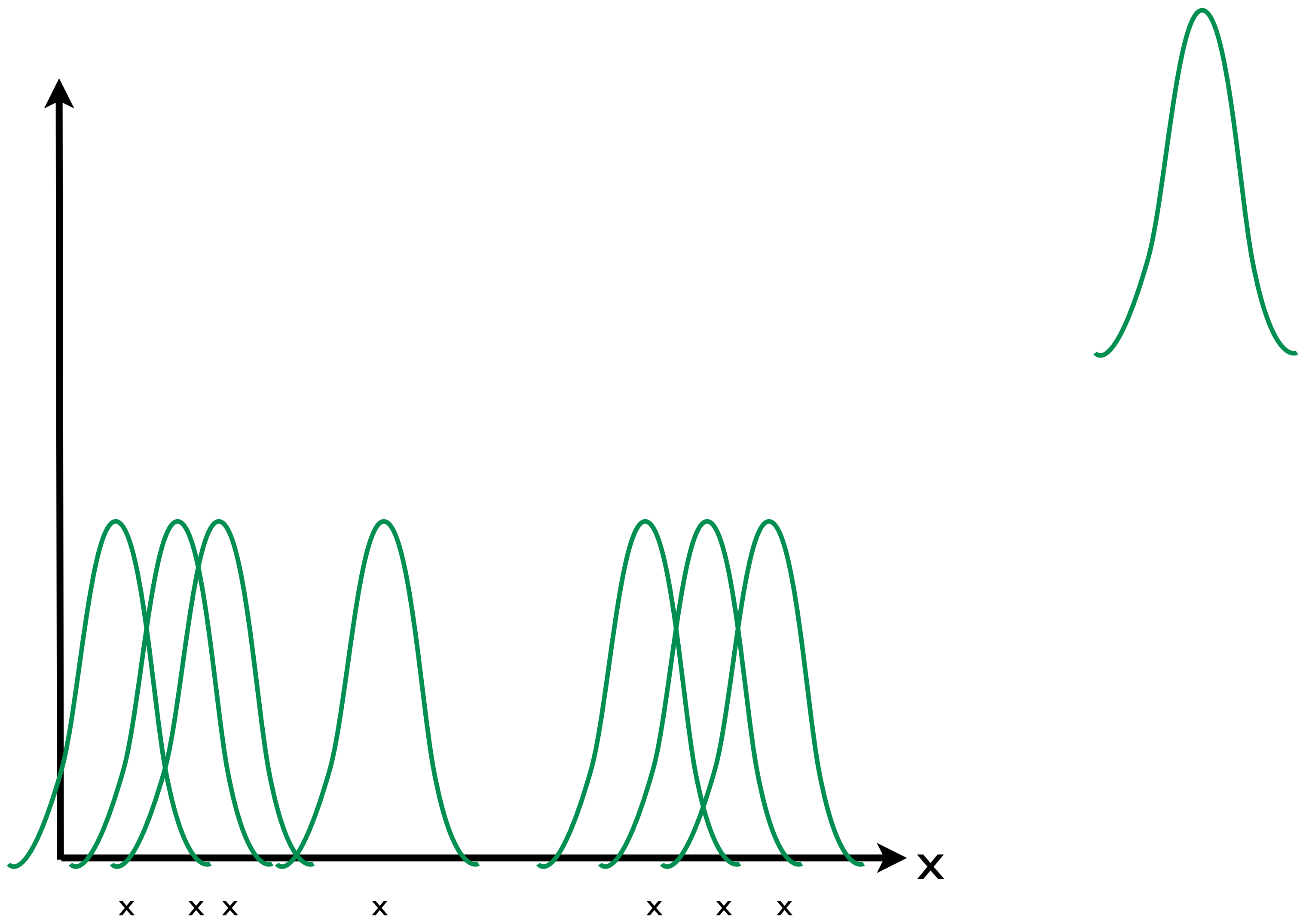


Kernel Density Estimation

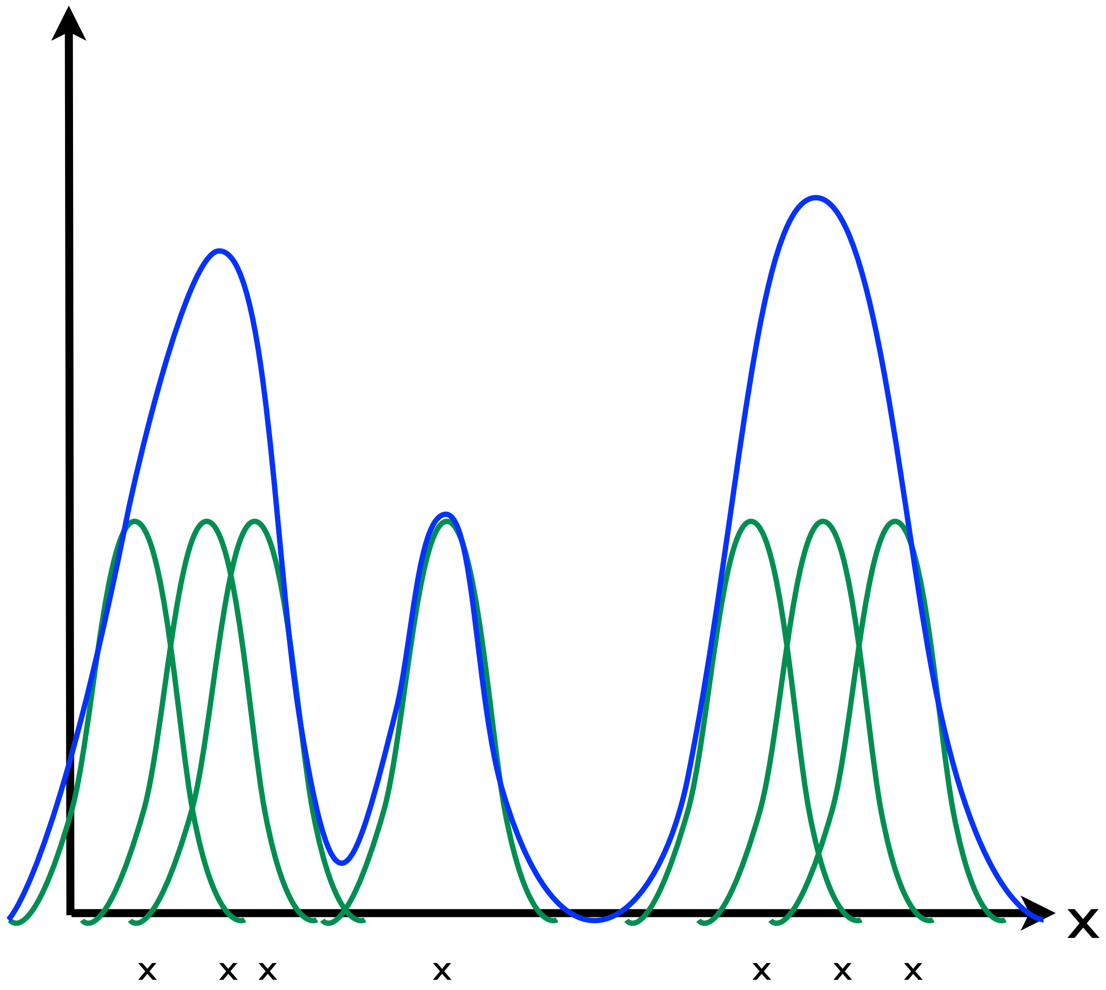


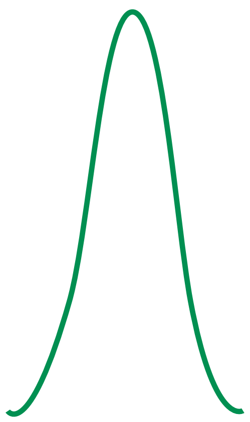


Kernel Density Estimation

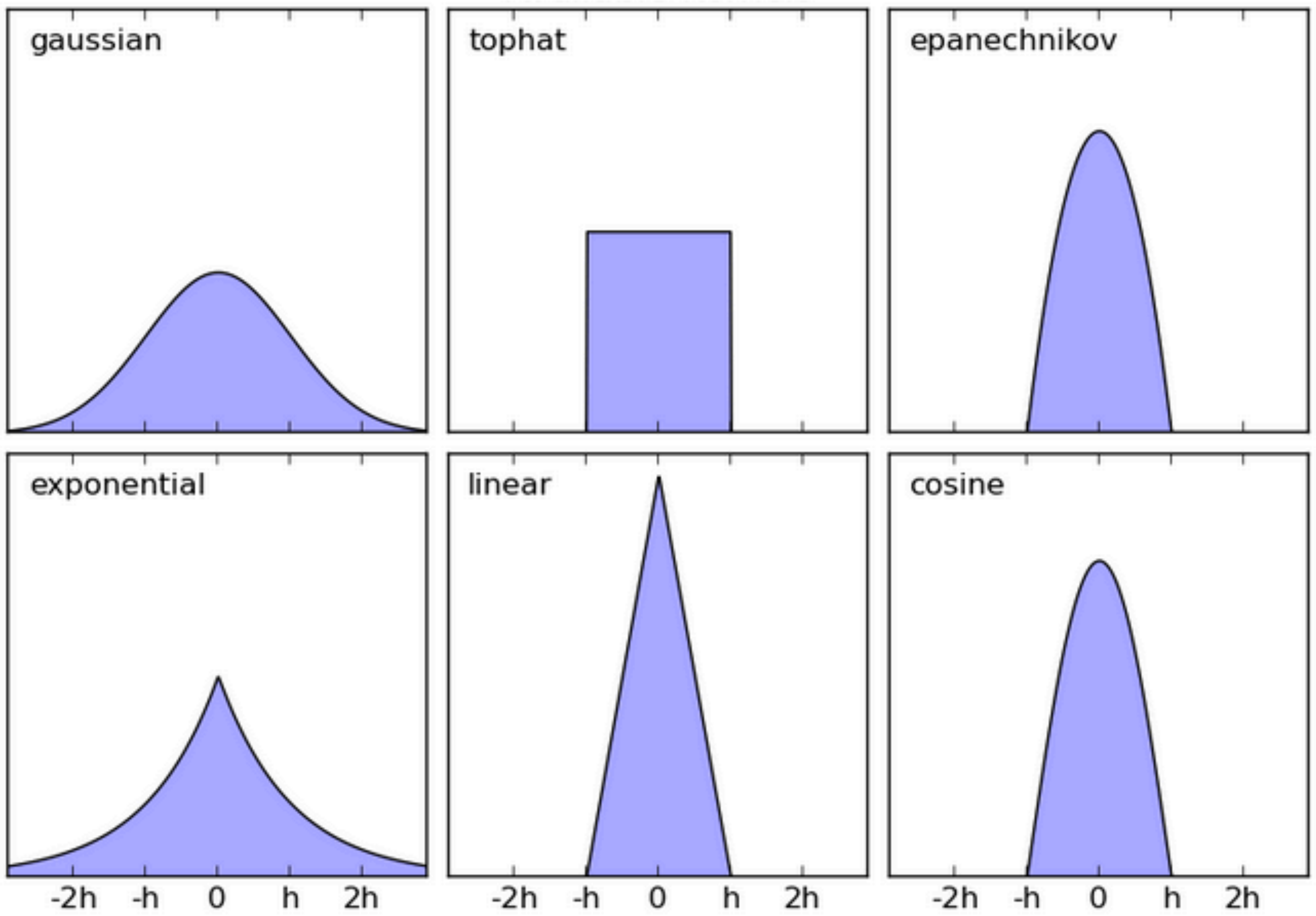


Kernel Density Estimation





Some Kernels



scikit-learn.org

- Gaussian kernel (`kernel = 'gaussian'`)

  $K(x;h) \propto \exp(-\frac{x^2}{2h^2})$

- Tophat kernel (`kernel = 'tophat'`)

  $K(x;h) \propto 1$ if $x < h$

- Epanechnikov kernel (`kernel = 'epanechnikov'`)

  $K(x;h) \propto 1 - \frac{x^2}{h^2}$

- Exponential kernel (`kernel = 'exponential'`)

  $K(x;h) \propto \exp(-x/h)$

- Linear kernel (`kernel = 'linear'`)

  $K(x;h) \propto 1 - x/h$ if $x < h$

- Cosine kernel (`kernel = 'cosine'`)

  $K(x;h) \propto \cos(\frac{\pi x}{2h})$ if $x < h$

Bandwidth Selection

- For a Gaussian Kernel $h = \left(\frac{4\hat{\sigma}^5}{3n} \right)^{\frac{1}{5}}$

- where $\hat{\sigma}$ is the standard deviation of the samples

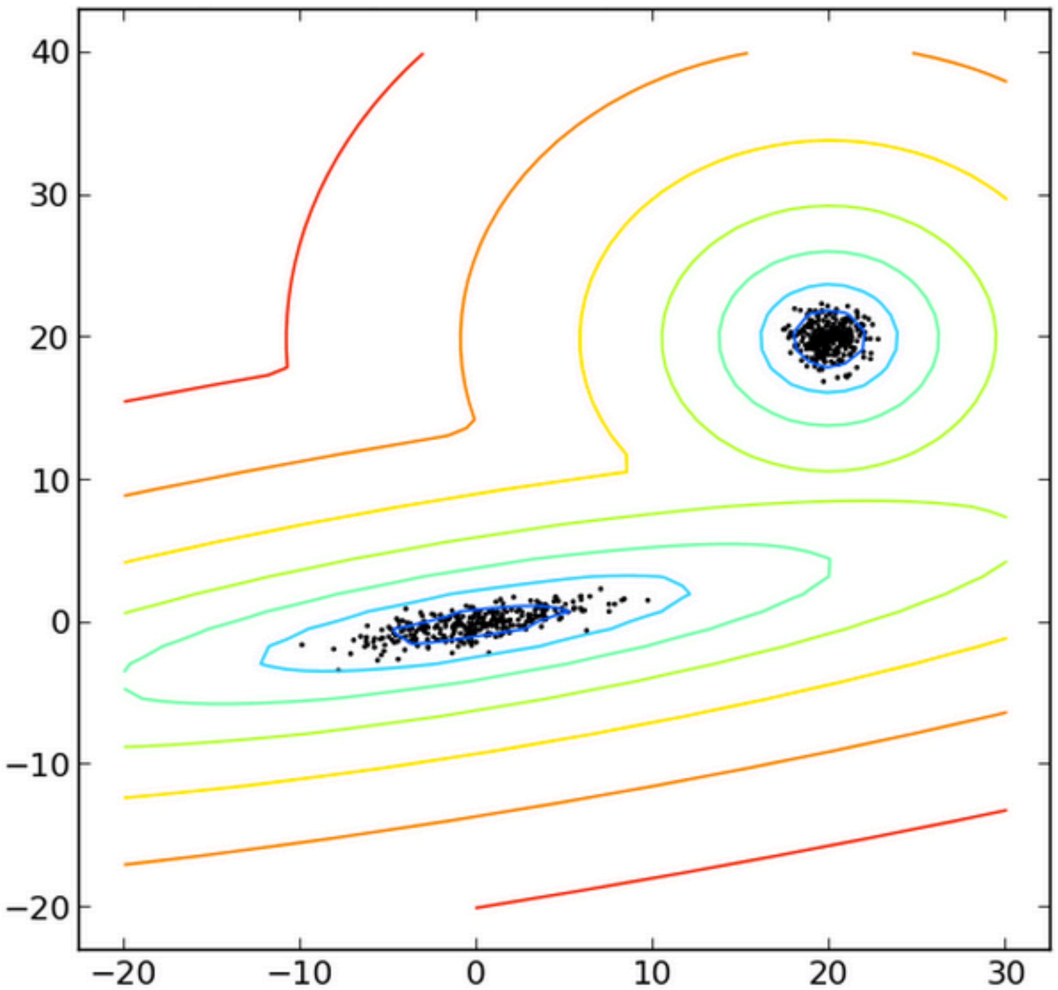
- For other kernels use the cross-validation score function

$$\widehat{J}(h) = \frac{1}{nh^2} \sum_{i=1}^{n} \sum_{j=1}^{n} K^* \left(\frac{X_i - X_i}{h} \right) + \frac{2}{nh} K(0)$$

$$K^*(x) = K^{(2)}(x) - 2K(x)$$

- For details see Silverman, B.W. (1998). Density Estimation for Statistics and Data Analysis.

Gaussian Mixture Models



$$
\begin{aligned}
p(\mathbf{x}) &= \sum_{z} p(\boldsymbol{x}|z)p(z) \\
&= \sum_{k=1}^{m} p(z=e_k)p(\boldsymbol{x}|z=e_k) \\
&= \sum_{k=1}^{m} \alpha_k N(\boldsymbol{x}|\mu_k, C_k)
\end{aligned}
$$

$$
\begin{aligned}
e_1 &= (1,0,0,\ldots,0) \\
e_2 &= (0,1,0,\ldots,0) \\
&\ldots
\end{aligned}
$$

Gaussian Mixture Model

- For a given dataset fit m Gaussians.
- Expectation-Maximization:
  - Given a set X of observed data, a set of latent data (missing values) Z, and unknown parameters θ ,
  - The maximum likelihood estimate of the unknown parameters is determined by the marginal likelihood of the observed data

$$L(\theta; X) = p(X|\theta) = \sum_Z p(X, Z|\theta)$$

  - **E-step**: Calculate the expected log likelihood with respect to the conditional distribution of Z given X under the current estimate of θ
  - **M-step:** Find θ that maximizes this expected log likelihood

Choosing m

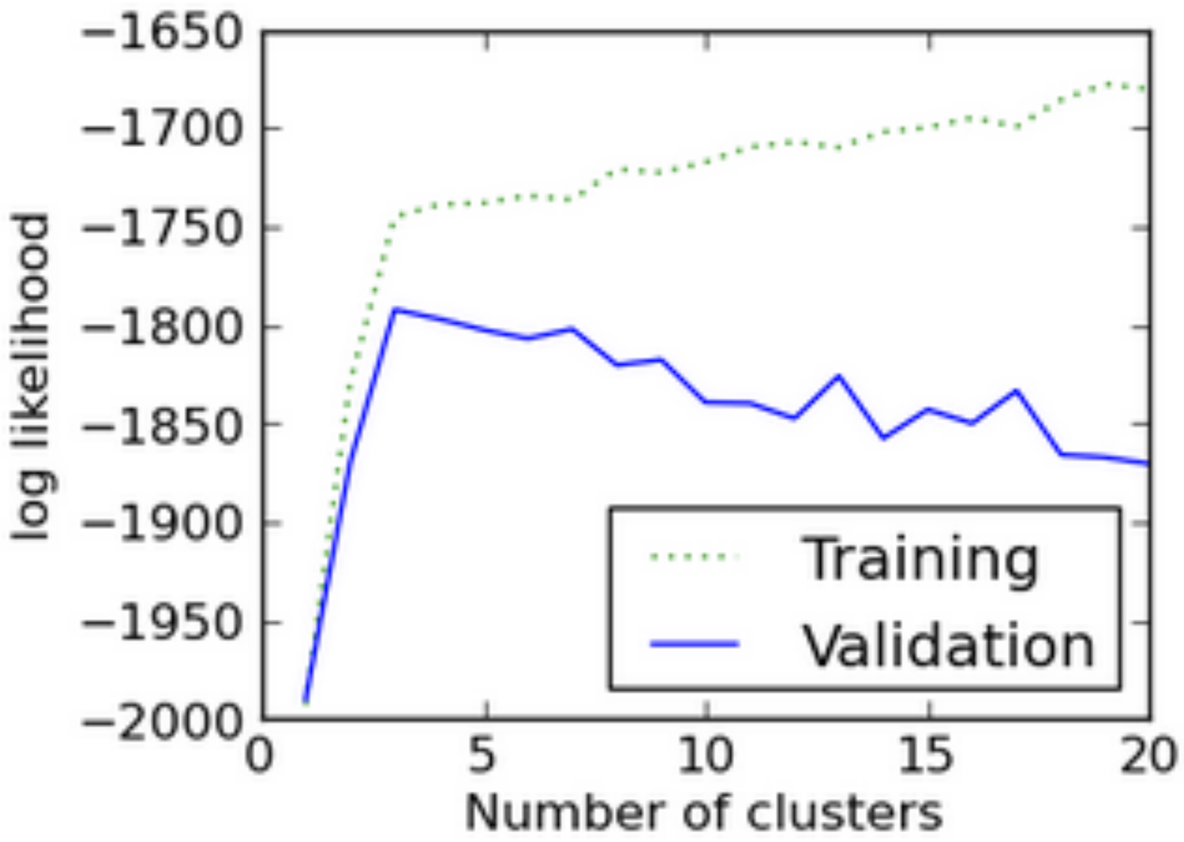
- Bayesian Information Criterion

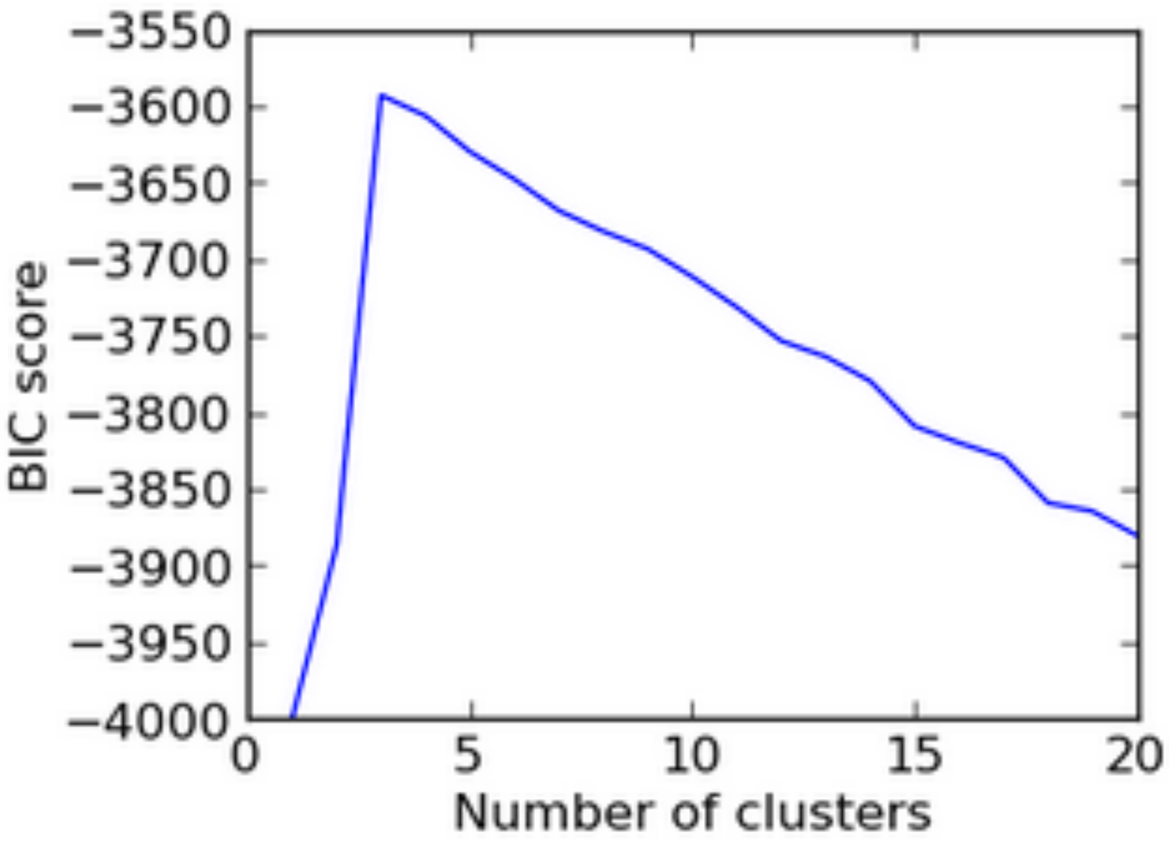
$$\mathrm{BIC} = -2\ln \hat{L} + k\ln(n)$$

- k = number of free parameters

  = m -1 + m(D + D(D+1)/2)

Choosing m





source: http://pypr.sourceforge.net/examples.html#bicexample

Summary

- Supervised Learning
    - ANN, SVM, Decision Trees, Bayesian Classificators, Nearest Neighbours, etc...
- Unsupervised Learning
    - Clustering: K-Means, Hierarchical Clustering, DBSCAN, etc...
    - Density Estimation: Histograms, Kernel Density Estimation, Gaussian Mixture Models
- Validate your methods!!!

Summary

- Supervised Learning
  - ANN, SVM, Decision Trees, Bayesian Classificators, Nearest Neighbours, etc...
- Unsupervised Learning
  - Clustering: K-Means, Hierarchical Clustering, DBSCAN, etc...
  - Density Estimation: Histograms, Kernel Density Estimation, Gaussian Mixture Models
- Validate your methods!!!

Thank You!