

VAPNIK-CHEVONENKIS LEARNING THEORY

Václav Hlaváč

Czech Technical University, Faculty of Electrical Engineering
Department of Cybernetics, Center for Machine Perception
121 35 Praha 2, Karlovo nám. 13, Czech Republic

hlavac@fel.cvut.cz, <http://cmp.felk.cvut.cz>

LECTURE PLAN

1. Classifier design.
2. Mathematical formulation of the risk describing process of learning.
3. Upper bound = guaranteed risk.
4. VC-dimension calculation.
5. Structural risk minimization.

CLASSIFIER DESIGN (1)

The object of interest is characterized by observable properties $x \in X$ and its class membership (unobservable, hidden state) $k \in K$, where X is the space of observations and K the set of hidden states.

The objective of a classifier design is to find the optimal decision function $q^*: X \rightarrow K$.

Bayesian decision theory solves the problem of minimization of risk

$$R(q) = \sum_{x,k} p_{XK}(x, k) W(k, q(x))$$

given the following quantities:

- ◆ $p_{XK}(x, k)$, $\forall x \in X, k \in K$ – the statistical model of the dependence of the observable properties (measurements) on class membership.
- ◆ $W(k, q(x))$ the loss of decision $q(x)$ if the true class is k .

Constraints or penalties for different errors depend on the application problem formulation.

However, in applications typically:

- ◆ None of the probabilities are known, e.g., $p(x|k)$, $p(k)$, $\forall x \in X$, $k \in K$.
- ◆ The designer is only given a **training multiset** $T = \{(x_1, k_1) \dots (x_L, k_L)\}$, where L is the length (size) of the training multiset.
- ◆ The desired properties of the classifier $q(x)$ are assumed.

Note: Non-Bayesian decision theory offers the solution to the problem if $p(x|k)$, $\forall x \in X$, $k \in K$ are known, but $p(k)$ are unknown (or do not exist).

CLASSIFIER DESIGN via PARAMETER ESTIMATION

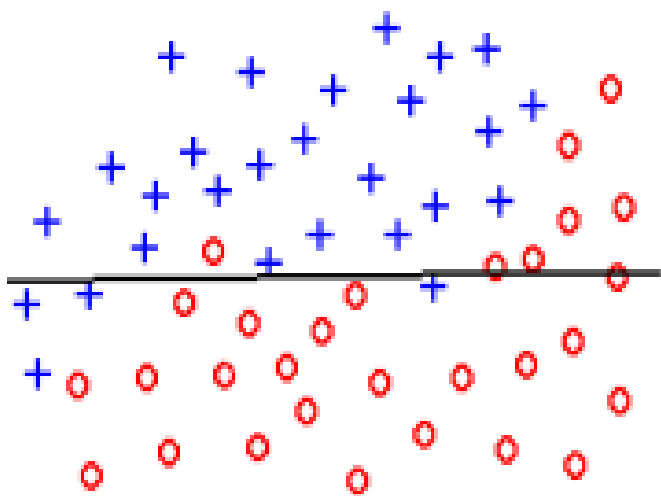
- ◆ Assume $p(x, k)$ have a particular form, e.g., a mixture of Gaussians, piece-wise constant, etc., with a finite (i.e., small) number of parameters Θ_k .
- ◆ Estimate the parameters Θ_k from the using training set T .
- ◆ Solve the classifier design problem (i.e., minimise the risk) by **substituting** the estimated $\hat{p}(x, k)$ for the true (and unknown) probabilities $p(x, k)$.
 - : There is no direct relationship between known properties of estimated $\hat{p}(x, k)$ and the properties (typically the risk) of the obtained classifier $q'(x)$.
 - : If the true $p(x, k)$ is not of the assumed form then $q'(x)$ may be arbitrarily bad, even if the size of training set L approaches infinity!
 - + : Implementation is often straightforward, especially if parameters Θ_k for each class are assumed independent.
 - + : Performance on real data can be predicted empirically from performance on training set (divided to training set and validation set, e.g., crossvalidation).

LEARNING in STATISTICAL PATTERN RECOGNITION

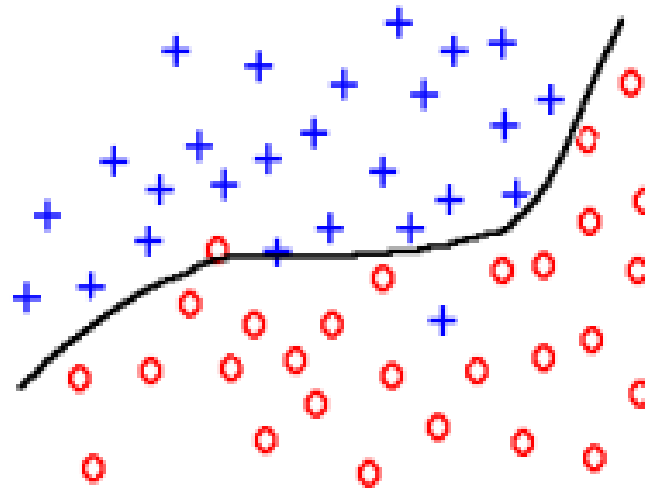
- ◆ Choose a class Q of decision functions (classifiers) $q: X \rightarrow K$.
- ◆ Find $q^* \in Q$ by minimizing some criterion function on the training set that approximates the risk $R(q)$ (which cannot be computed).
- ◆ Learning paradigm is defined by the approximating criterion function:
 1. Maximizing likelihood.
Example:
 2. Using a non-random training set.
Example: Image analysis.
 3. Empirical risk minimization in which the true risk is approximated by the error rate on the training set.
Examples: Perceptron, Neural nets (Back-propagation), etc.
 4. Structural risk minimization.
Example: SVM (Support Vector Machines).

OVERFITTING AND UNDERFITTING

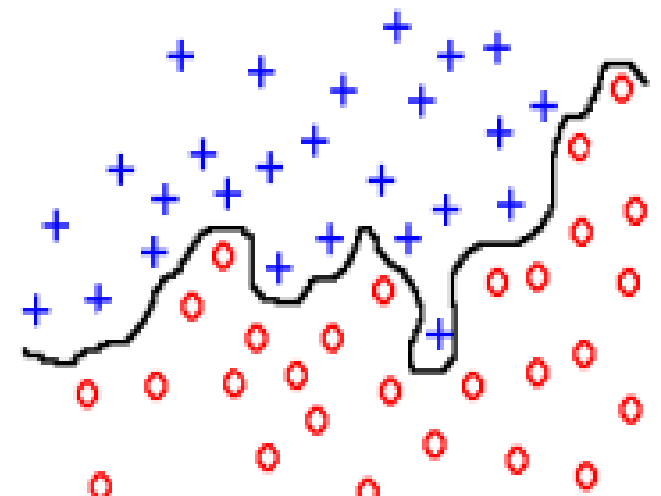
- ◆ How rich class \mathcal{Q} of classifiers $q(x, \Theta)$ should be used?
- ◆ The problem of generalization is a key problem of pattern recognition: a small empirical risk R_{emp} need not imply a small true expected risk R !



underfit



fit



overfit

ASYMPTOTIC BEHAVIOR

- ◆ For infinite training data, the law of large number assures

$$\lim_{L \rightarrow \infty} R_{\text{emp}}(\Theta) = R(\Theta) .$$

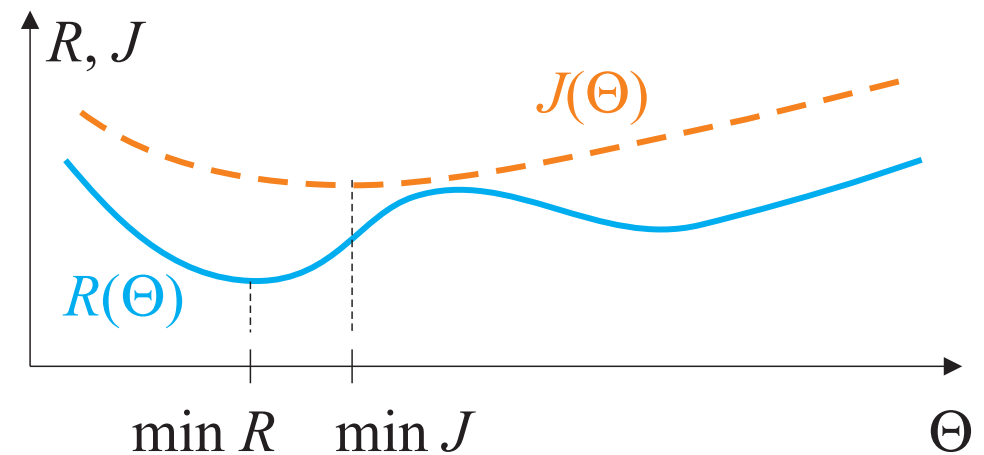
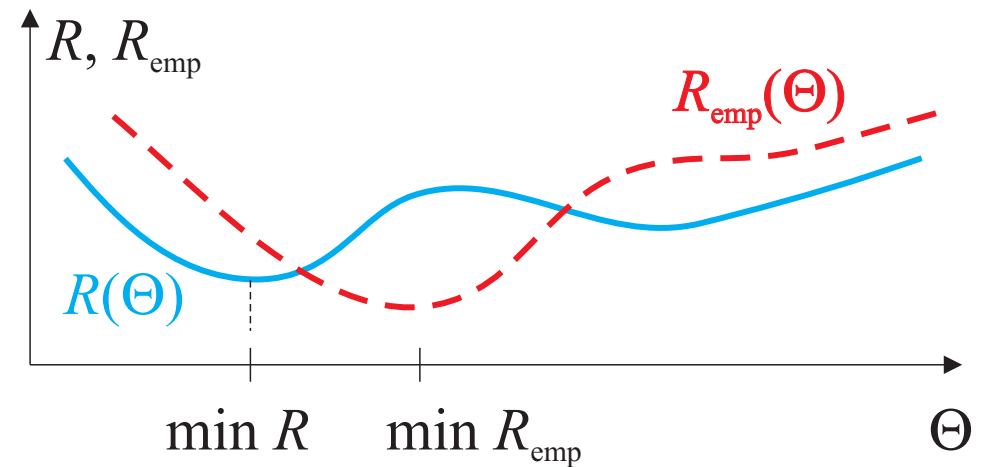
- ◆ In general, unfortunately, there is no guarantee for a solution based on expected risk minimization because

$$\operatorname{argmin}_{\Theta} R_{\text{emp}}(\Theta) \neq \operatorname{argmin}_{\Theta} R(\Theta) .$$

Performance on training data is often be better than on test data (or real performance).

IDEA OF THE GUARANTEED RISK

- ◆ Idea: add a prior (called also regularizer).
- ◆ This regularizer favors a simpler strategy, cf., Occam razor.
- ◆ Vapnik-Chervonenkis learning theory introduces a **guaranteed risk** $J(\Theta)$, $R(\Theta) \leq J(\Theta)$, with the probabilistic confidence η .
- ◆ The upper bound $J(\Theta)$ may be so large (meaning pessimistic) that it can be useless.



UPPER BOUND OF A TRUE RISK

- ◆ The upper bound was derived by Chervonenkis and Vapnik in the 1970s.
- ◆ With the confidence η , $0 \leq \eta \leq 1$,

$$R(\Theta) \leq J(\Theta) = R_{\text{emp}}(\Theta) + \sqrt{\frac{h \left(\log \left(\frac{2L}{h} \right) + 1 \right) - \log \left(\frac{\eta}{4} \right)}{L}}.$$

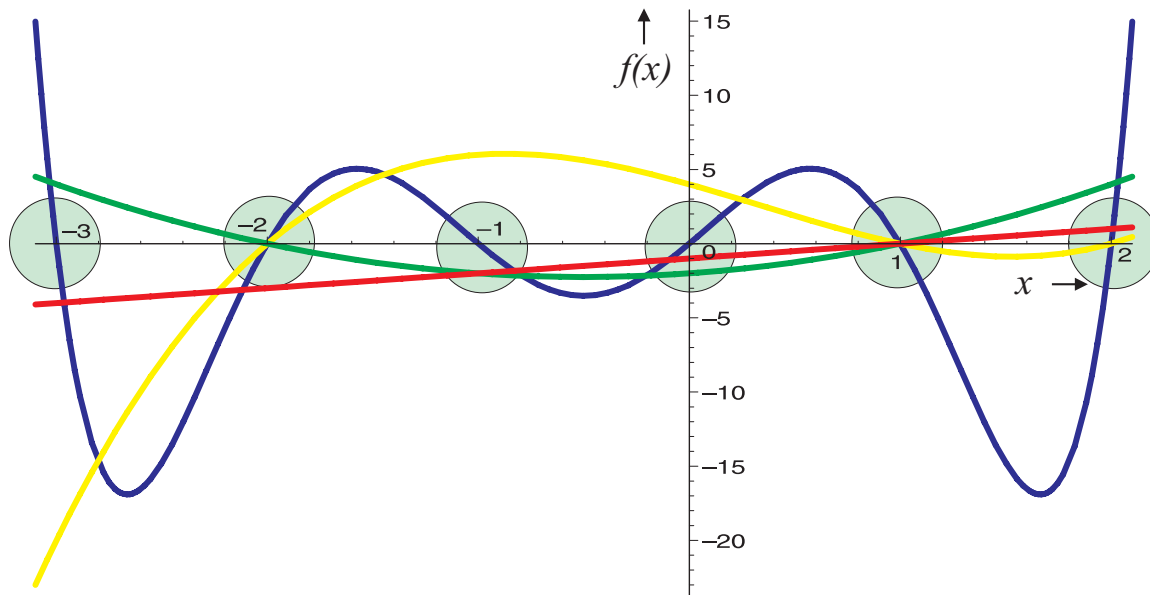
where L is the length of the training multi-set, h is the VC-dimension of the class of strategies $q(x, \Theta)$.

- ◆ Note that the above **upper bound is independent of the true $p(x, k)$!!**
- ◆ It is a worst case upper bound valid for all possible $p(x, k)$.
- ◆ **Structural risk minimization** means minimizing the upper bound $J(\Theta)$.
(We will return to structural risk minimization after we explain how to compute VC-dimension.)

VAPNIK-CHERVONENKIS DIMENSION

- ◆ It is a number characterizing the decision strategy.
- ◆ Abbreviated **VC-dimension**.
- ◆ Named after Vladimir Vapnik and Alexey Chervonenkis
(Appeared in their book in Russian. V. Vapnik, A. Chervonenkis: Pattern Recognition Theory, Statistical Learning Problems, Nauka, Moskva, 1974).
- ◆ It is one of the core concepts in Vapnik-Chervonenkis theory of learning.
- ◆ In the original 1974 publication, it was called **capacity of a class of strategies**.
- ◆ The VC dimension is a measure of the capacity of a statistical classification algorithm.

VC-DIMENSION, THE IDEA INFORMALLY



$$f_1(x) = (x - 1)$$

$$f_2(x) = (x - 1)(x + 2)$$

$$f_3(x) = (x - 2)(x - 1)(x + 2)$$

$$f_6(x) = (x - 2)(x - 1) x (x + 1) \\ (x + 2)(x + 3)$$

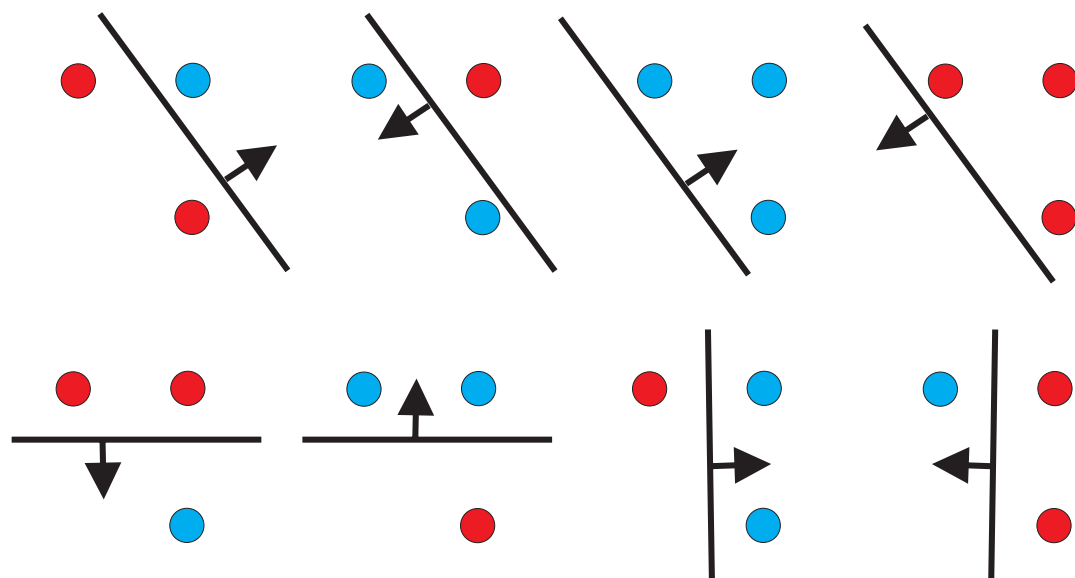
Light blue circles symbolize data points.

- ◆ The capacity of a classification strategy tells how complicated it can be.
- ◆ An example could be thresholding a high-degree polynomial. If a very high-degree polynomial is used, it can be very wiggly, and can fit a training set exactly (overfit). Such a polynomial has a high capacity and problems with generalization.
- ◆ A linear function, e.g., has a low capacity.

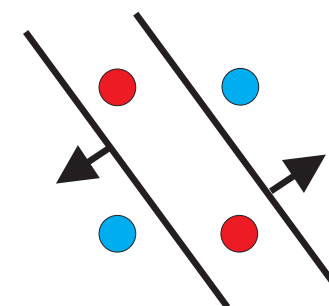
SHATTERING

- ◆ Consider a classification strategy q with some parameter vector Θ .
- ◆ The model q can shatter a set of data points x_1, x_2, \dots, x_n if, for all assignments of labels $k \in K$ to data points, there exists a parameter Θ such that the model q makes no errors when evaluating that set of data points.

Shattering example: q is a line in 2D feature space.



3 points, shattered



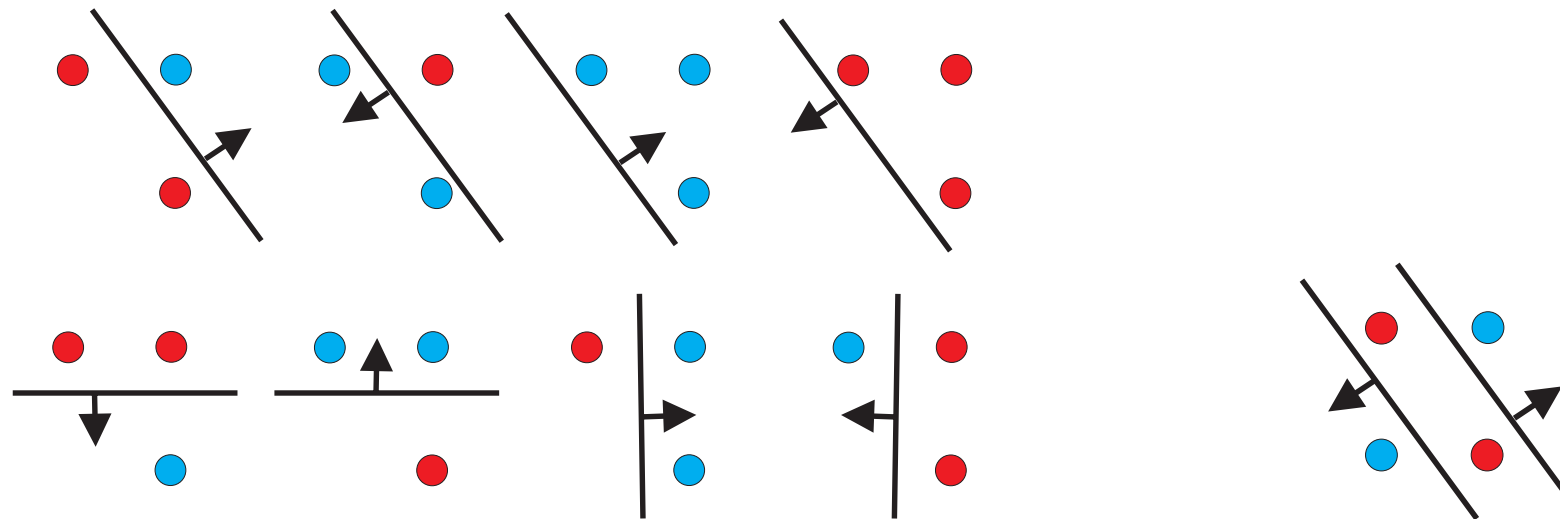
4 points, undivisible

VC-DIMENSION h , DEFINITION

- ◆ Consider a set of dichotomic strategies $q(x, \Theta) \in Q$.
- ◆ The set consisting of h data points (observations) can be labelled in 2^h possible ways.
- ◆ A strategy $q \in Q$ exists which assigns labels correctly to all possible configurations.
(Process of finding all possible configurations with correctly assigned labels is called shattering.)
- ◆ VC-dimension (definition) is the maximal number h of data points (observations) that can be shattered.

VC-DIMENSION OF A LINEAR STRATEGY IN A 2D FEATURE SPACE

- ◆ A set of parameters $\Theta = \{\Theta_0, \Theta_1, \Theta_2\}$.
A linear strategy $q(x, \Theta) = \Theta_1 x_1 + \Theta_2 x_2 + \Theta_0$.
- ◆ Shattering example (revisited):

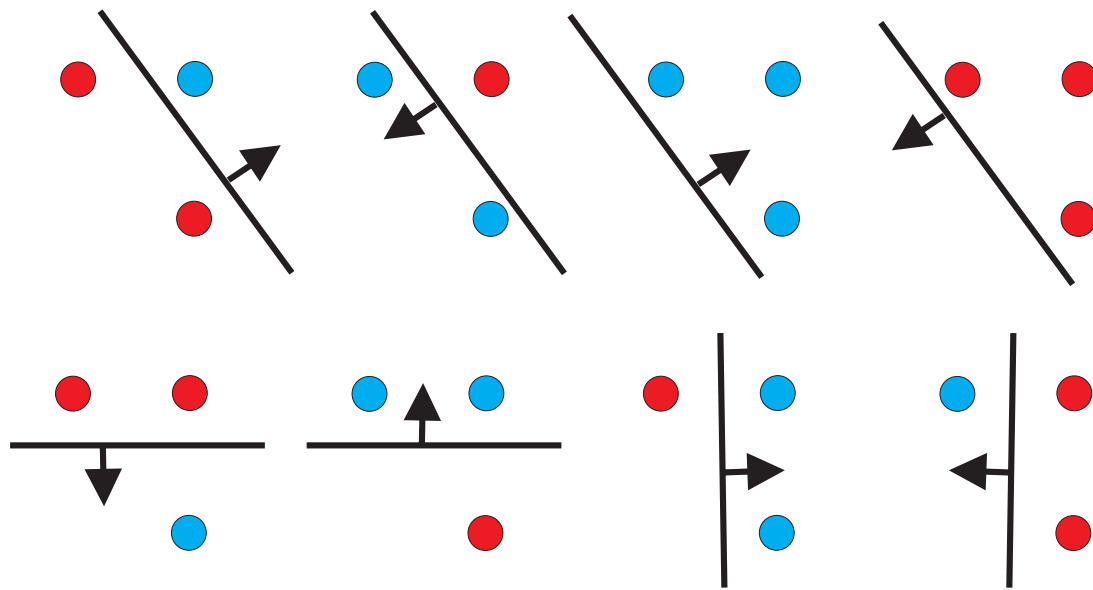


3 points, shattered

4 points, undivisible

- ◆ 3 points in 2D space ($n = 2$) can be shattered.
There was counter example given that 4 points cannot be shattered.
 \Rightarrow VC-dimension $h = 3$.

VC-DIMENSION FOR A LINEAR STRATEGY IN A n -DIMENSIONAL SPACE



A special case, $n=2$.

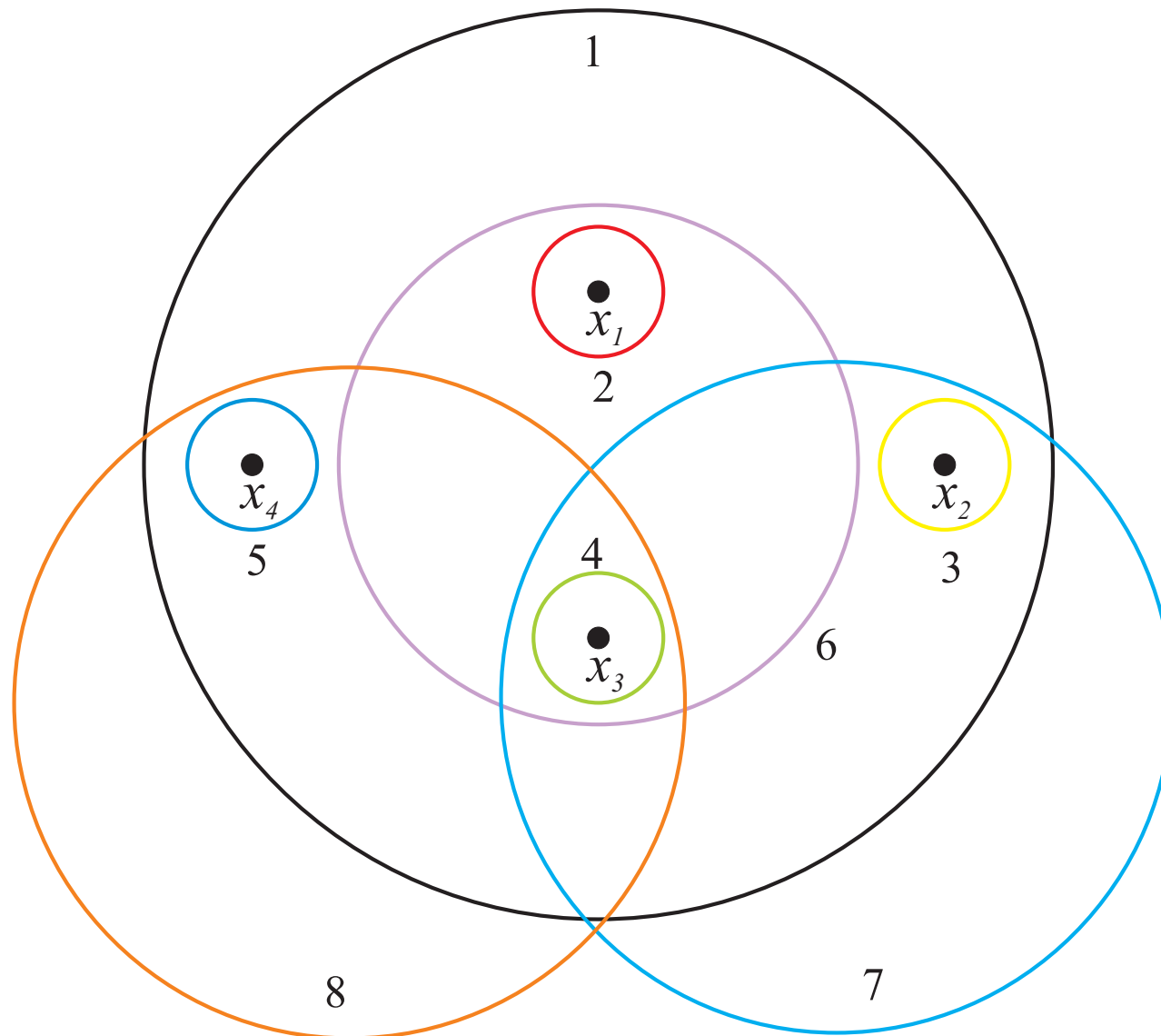
VC-dimension = 3.

Generalization to n -dimensions for linear classifiers

- ◆ A hyperplane in the space \mathbb{R}^n shatters any set of $h = n + 1$ linearly independent points.
- ◆ Consequently, VC-dimension of linear decision strategies is $h = n + 1$.

VC-DIMENSION IN A 2D SPACE FOR A CIRCULAR STRATEGY

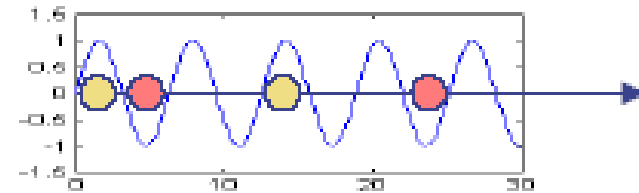
Maximally 4 data points in \mathbb{R}^2 can be shattered in 8 possible ways
 \Rightarrow VC-dimension $h = 4$.



VC-DIMENSION AND # OF PARAMETERS

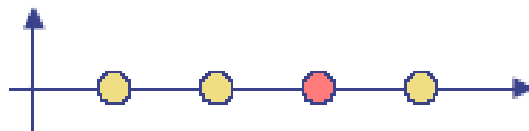
- ◆ Consider a sinusoidal classifier in a 1D feature space, $q(x, \Theta) = \text{sign}(\sin(\Theta)x)$, $x, \Theta \in \mathbb{R}$.
- ◆ For any given number $L \in \mathbb{N}$, the points x_i can be chosen as $x_i = 10^{-i}$, $i = 1, \dots, L$ the labels can be specified arbitrarily k_1, k_2, \dots, k_L , $k_i \in \{-1, 1\}$.
- ◆ Then $q(x, \Theta)$ a correct labelling if Θ is chosen as

$$\Theta = \pi \left(1 + \sum_{i=1}^L \frac{(1-k_i) 10^i}{2} \right).$$



- ◆ Thus the VC dimension of this decision strategy is infinite.
-

- ◆ There exists x_i which cannot shatter, e.g., equidistantly spaced ones.



STRUCTURAL RISK MINIMIZATION

- ◆ Minimize guaranteed risk $J(\Theta)$, that is the upper bound

$$R(\Theta) \leq J(\Theta) = R_{\text{emp}}(\Theta) + \sqrt{\frac{h \left(\log \left(\frac{2L}{h} \right) + 1 \right) - \log \left(\frac{\eta}{4} \right)}{L}}.$$

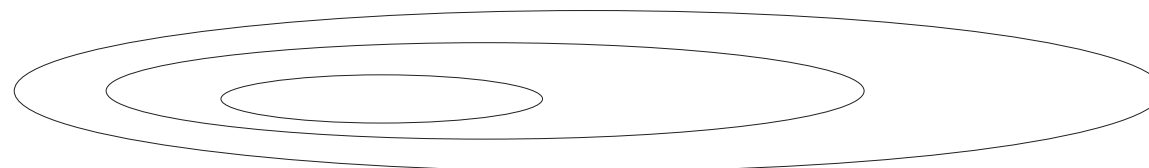
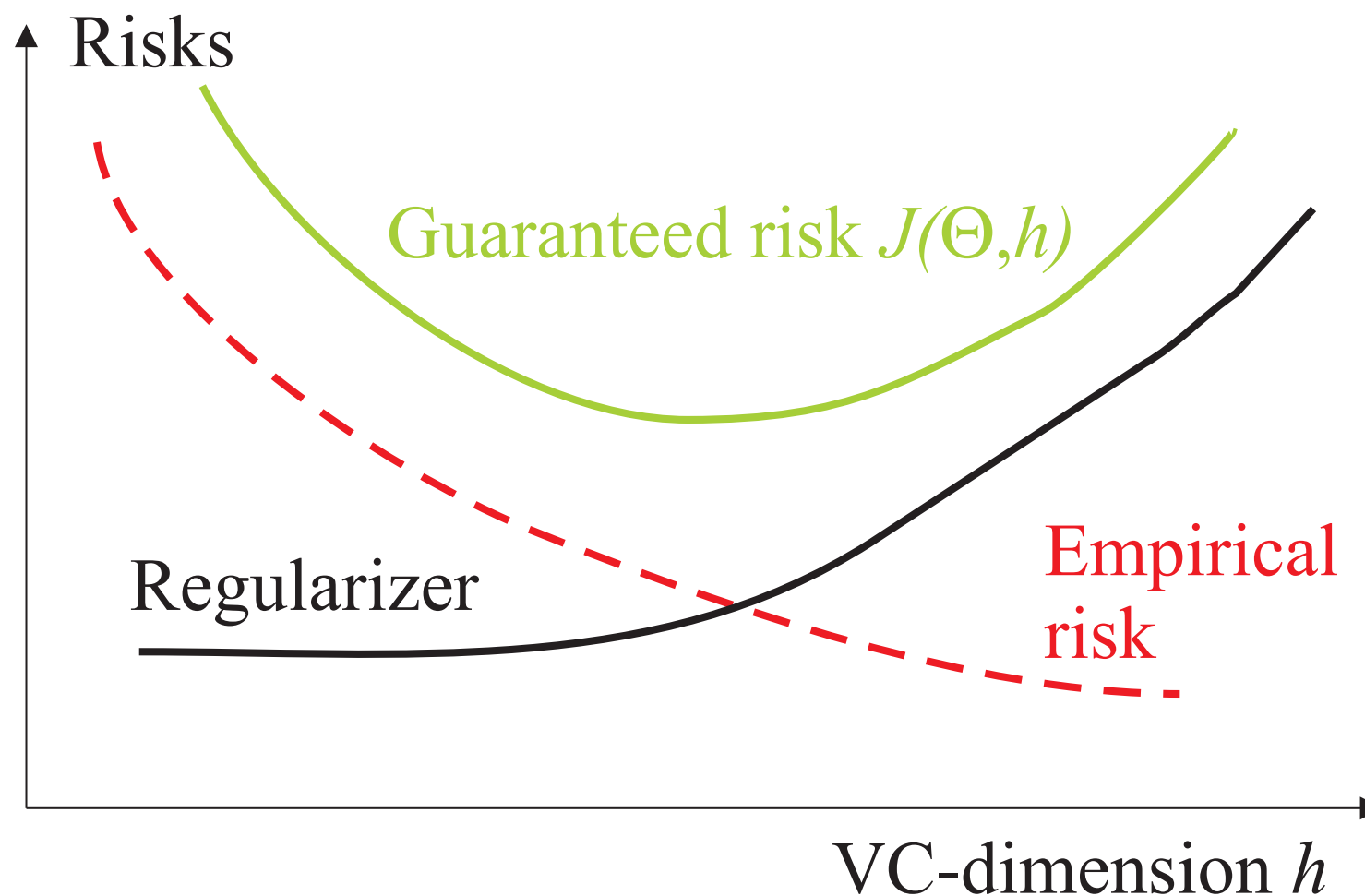
For each model i in the list of hypotheses

- Compute its VC-dimension h_i .
- $\Theta_i^* = \underset{\Theta_i}{\operatorname{argmin}} R_{\text{emp}}(\Theta_i)$.
- Compute $J_i(\Theta_i^*, h_i)$.

Choose the model with the lowest $J_i(\Theta_i^*, h_i)$.

- ◆ Preferably, optimize directly over both $(\Theta^*, h^*) = \underset{\Theta, h}{\operatorname{argmin}} J(\Theta, h)$.
- ◆ Gap tolerant linear classifiers minimize $R_{\text{emp}}(\Theta)$ while maximizing margin. Support Vector Machine does just that.

STRUCTURAL RISK MINIMIZATION PICTORIALLY



Space of nested hypotheses with decreasing h

VC-DIMENSION, A PRACTICAL VIEW

Bad news: Computing guaranteed risk is useless in many practical situations.

- ◆ VC dimension cannot be accurately estimated for non-linear models such as neural networks.
- ◆ Structural Risk Minimization may lead to a non-linear optimization problem.
- ◆ VC dimension may be infinite (e.g., for a nearest neighbor classifier), requiring infinite amount of training data.

Good news: Structural Risk Minimization can be applied for linear classifiers.

- ◆ Especially useful for Support Vector Machines.

Is then empirical risk minimization = minimization of training set error, e.g., neural networks with backpropagation, dead ? **No!**

– Guaranteed risk J may be so large that this upper bound becomes useless.

Find a tighter bound and you will be famous! It is not impossible!

- + Vapnik, Chervonenkis suggest learning with progressively more complex classes of the decision strategies Q .
- + Vapnik & Chervonenkis' theory justifies using empirical risk minimization on classes of functions with a reasonable VC dimension.
- + Empirical risk minimization is computationally hard (impossible for large L). Most classes of decision functions Q where empirical risk minimization (at least local) can be efficiently organized are often useful.

Where does the nearest neighbor classifier fit in the picture?