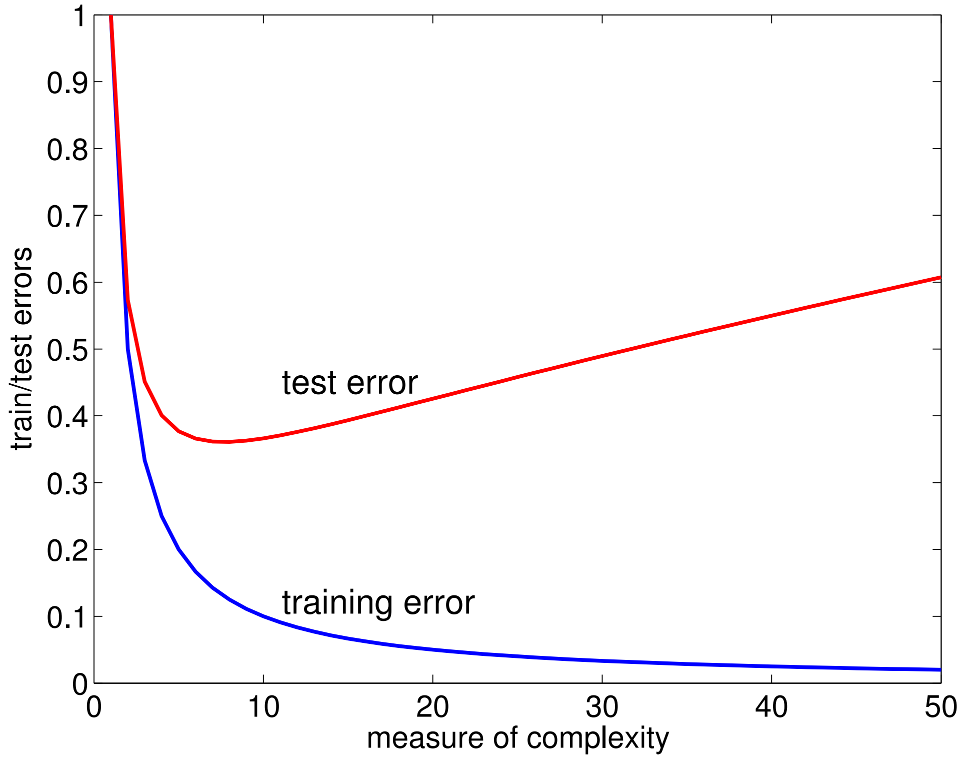
Machine learning: lecture 12

Tommi S. Jaakkola
MIT CSAIL
tommi@csail.mit.edu

Topics

- Complexity and model selection
  - Finite case
  - VC dimension
  - structural risk minimization

Why care about "complexity"?



- We need a quantitative measure of complexity in order to be able to relate the training error (which we can observe) and the test error (that we'd like to optimize)

Simple case: finite number of classifiers

- Suppose we consider only a finite number of classifiers, $h_1(\mathbf{x}), \ldots, h_m(\mathbf{x})$.
- How does the number of classifiers m affect the difference between training and test errors?

Simple case: finite number of classifiers

- Suppose we consider only a finite number of classifiers, $h_1(\mathbf{x}), \ldots, h_m(\mathbf{x})$.
- How does the number of classifiers m affect the difference between training and test errors?

  Let's start by defining

$$
\begin{aligned}
\hat{\mathcal{E}}_n(i) &= \frac{1}{n}\sum_{t=1}^{n} \overbrace{\text{Loss}(y_t, h_i(\mathbf{x}_t))}^{=0,1} = \text{ empirical error of } h_i(\mathbf{x}) \\
\mathcal{E}(i) &= E_{(\mathbf{x},y)\sim P}\left\{\, \text{Loss}(y, h_i(\mathbf{x}))\,\right\} = \text{ expected error of } h_i(\mathbf{x})
\end{aligned}
$$

Simple case cont'd

$$
\begin{aligned}
\hat{\mathcal{E}}_n(i) &= \frac{1}{n}\sum_{t=1}^{n} \text{Loss}(y_t, h_i(\mathbf{x}_t)) = \text{empirical error of } h_i(\mathbf{x}) \\
\mathcal{E}(i) &= E_{(\mathbf{x},y)\sim P}\{\,\text{Loss}(y, h_i(\mathbf{x}))\,\} = \text{expected error of } h_i(\mathbf{x})
\end{aligned}
$$

- If we choose the classifier that minimizes the training error, $\hat{i}_n = \arg\min_i \hat{\mathcal{E}}_n(i)$, then

$$
\begin{aligned}
\text{Training error} &= \hat{\mathcal{E}}_n(\hat{i}_n) \\
\text{Test error} &= \mathcal{E}(\hat{i}_n)
\end{aligned}
$$

Simple case cont'd

$$\begin{aligned}\hat{\mathcal{E}}_n(i) &= \frac{1}{n}\sum_{t=1}^{n} \text{Loss}(y_t, h_i(\mathbf{x}_t)) = \text{empirical error of } h_i(\mathbf{x}) \\ \mathcal{E}(i) &= E_{(\mathbf{x},y)\sim P}\,\{\,\text{Loss}(y, h_i(\mathbf{x}))\,\} = \text{expected error of } h_i(\mathbf{x})\end{aligned}$$

- If we choose the classifier that minimizes the training error, $\hat{i}_n = \arg\min_i \hat{\mathcal{E}}_n(i)$, then

$$\begin{aligned}\text{Training error} &= \hat{\mathcal{E}}_n(\hat{i}_n) \\ \text{Test error} &= \mathcal{E}(\hat{i}_n)\end{aligned}$$

- The training and test errors are necessarily close if

$$|\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| \le \epsilon, \;\; \text{for all } i = 1, \ldots, m$$

Simple case cont'd

- We'd like to evaluate the probability that the training error deviates more than ϵ from the corresponding test error:

$$P\left(\exists\, i : \; |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right)$$

  where the probability is over the choice of the training set.

Simple case cont'd

- We'd like to evaluate the probability that the training error deviates more than ϵ from the corresponding test error:

$$P\left(\exists\, i:\; |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right)$$

where the probability is over the choice of the training set.

By using the fact that $P(A\,\text{or}\,B) \leq P(A) + P(B)$ we get

$$P\left(\exists\, i:\; |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right) \;\leq\; \sum_{i=1}^{m} P\left(\,|\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\,\right)$$

Simple case cont'd

- We'd like to evaluate the probability that the training error deviates more than ϵ from the corresponding test error:

$$P\left(\exists\, i:\ |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right)$$

where the probability is over the choice of the training set.

By using the fact that $P(A \,\text{or}\, B) \leq P(A) + P(B)$ we get

$$\begin{aligned} P\left(\exists\, i:\ |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right) &\leq \sum_{i=1}^{m} P\left(|\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right) \\ &\leq \sum_{i=1}^{m} 2\exp(-2n\epsilon^2) \quad \text{(Chernoff)} \end{aligned}$$

Simple case cont'd

- We'd like to evaluate the probability that the training error deviates more than ϵ from the corresponding test error:

$$P\left(\exists\, i :\ |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right)$$

where the probability is over the choice of the training set.

By using the fact that $P(A \,\text{or}\, B) \leq P(A) + P(B)$ we get

$$\begin{aligned} P\left(\exists\, i :\ |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right) &\leq \sum_{i=1}^{m} P\left(|\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right) \\ &\leq \sum_{i=1}^{m} 2\exp(-2n\epsilon^2) \quad \text{(Chernoff)} \\ &= m \cdot 2\exp(-2n\epsilon^2) \end{aligned}$$

Simple case cont'd

- We'd like to evaluate the probability that the training error deviates more than ϵ from the corresponding test error:

$$P\left(\exists i: \, |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right)$$

where the probability is over the choice of the training set.

By using the fact that $P(A \,\text{or}\, B) \leq P(A) + P(B)$ we get

$$\begin{aligned} P\left(\exists i: \, |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right) &\leq \sum_{i=1}^{m} P\left(|\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon\right) \\ &\leq \sum_{i=1}^{m} 2\exp(-2n\epsilon^2) \quad \text{(Chernoff)} \\ &= m \cdot 2\exp(-2n\epsilon^2) = \delta \end{aligned}$$

where $(1-\delta)$ is our "confidence" that the errors are close.

Simple case cont'd

- We can restate our result in terms of a bound on the expected error of any classifier in our set.

$$m \cdot 2\exp(-2n\epsilon^2) = \delta, \quad \text{or} \quad \epsilon = \sqrt{\frac{1}{2n}(\log(2m) + \log(1/\delta))}$$

Theorem: With probability at least $1 - \delta$ over the choice of the training set, for all $i = 1, \ldots, m$

$$\mathcal{E}(i) \leq \hat{\mathcal{E}}_n(i) + \epsilon(n, m, \delta)$$

where $\epsilon = \epsilon(n, m, \delta)$ given above is a "complexity penalty".

Simple case cont'd

- We can restate our result in terms of a bound on the expected error of any classifier in our set.

$$m \cdot 2\exp(-2n\epsilon^2) = \delta, \quad \text{or} \quad \epsilon = \sqrt{\frac{1}{2n}(\log(2m) + \log(1/\delta))}$$

Theorem: With probability at least $1 - \delta$ over the choice of the training set, for all $i = 1, \ldots, m$

$$\mathcal{E}(i) \leq \hat{\mathcal{E}}_n(i) + \epsilon(n, m, \delta)$$

where $\epsilon = \epsilon(n, m, \delta)$ given above is a "complexity penalty".

- The complexity penalty
  - is an increasing function of m
  - increases as δ decreases
  - decreases as a function of n

Measures of complexity

- "Complexity" is a measure of a set of classifiers, not any specific (fixed) classifier
- Many possible measures
  - degrees of freedom
  - description length
  - Vapnik-Chervonenkis (VC) dimension

  etc.

VC-dimension: preliminaries

- **A set of classifiers $\mathbf{F}$:**
  For example, this could be the set of all possible linear separators, where $h \in F$ means that

$$h(\mathbf{x}) = \text{sign}\left(w_0 + \mathbf{w}^T\mathbf{x} \right)$$

  for some values of the parameters $\mathbf{w}, w_0$.

VC-dimension: preliminaries

- **Complexity:** how many different ways can we label n training points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ with classifiers $h \in F$?

  In other words, how many distinct binary vectors

  $$[h(\mathbf{x}_1)\; h(\mathbf{x}_2)\; \ldots\; h(\mathbf{x}_n)]$$

  do we get by trying each $h \in F$ in turn?

  $$\begin{array}{ll} [\;\; \text{-1} \quad 1 \quad \ldots \quad 1 \;\;] & h_1 \\ [\;\;\; 1 \quad \text{-1} \quad \ldots \quad 1 \;\;] & h_2 \\ & \ldots \end{array}$$

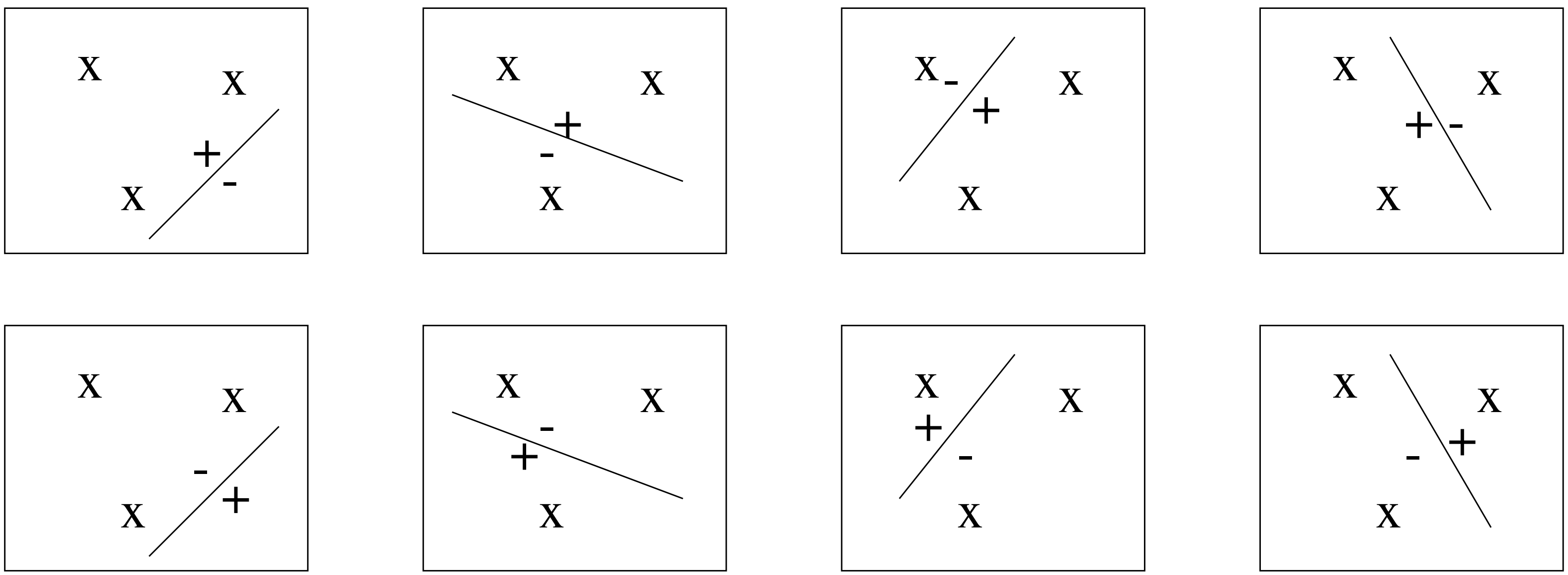
VC-dimension: shattering

- A set of classifiers F *shatters* n points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ if

$$[h(\mathbf{x}_1)\ h(\mathbf{x}_2)\ \ldots\ h(\mathbf{x}_n)],\ \ h \in F$$

generates all 2^n distinct labelings.

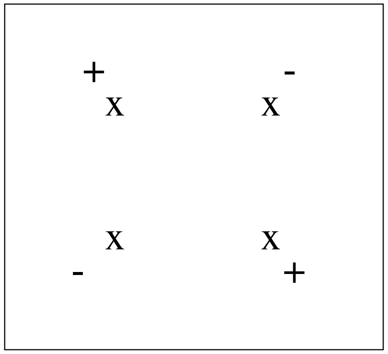
- Example: linear decision boundaries shatter (any) 3 points in 2D



but not any 4 points...

VC-dimension: shattering cont'd

- We cannot shatter 4 points in 2D with linear separators
  For example, the following labeling

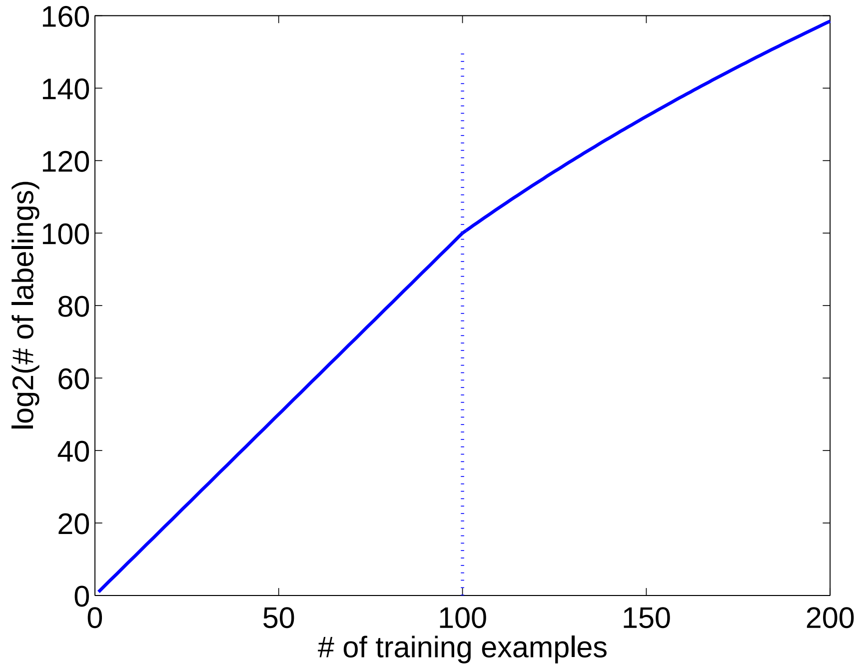


  cannot be produced with any linear separator

- More generally: the set of all d-dimensional linear separators can shatter exactly $d+1$ points
- **Definition:** The VC-dimension of a set of classifiers F is the number of points F can shatter

Learning and VC-dimension

- We don't really learn anything until after we have more than d_{VC} training examples



- The number of labelings that the set of classifiers can generate over n points increases sub-exponentially only after $n > d_{VC}$ (in this case $d_{VC} = 100$)

Learning and VC-dimension

- Let d_{VC} be the VC-dimension of our set of classifiers F.

  Theorem: With probability at least $1 - \delta$ over the choice of the training set, for all $h \in F$

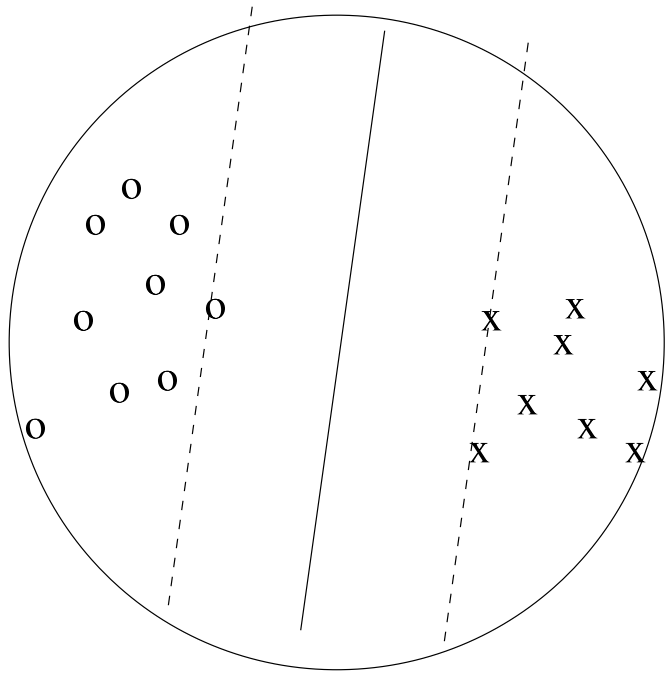
  $$\mathcal{E}(h) \leq \hat{\mathcal{E}}_n(h) + \epsilon(n, d_{VC}, \delta)$$

  where

  $$\epsilon(n, d_{VC}, \delta) = \sqrt{\frac{d_{VC}(\log(2n/d_{VC}) + 1) + \log(1/(4\delta))}{n}}$$

Complexity and margin

- The number of possible labelings of points with large margin can be dramatically less than the (basic) VC-dimension



- The set of separating hyperplaces which attain margin γ or better for examples within a sphere of radius R has VC-dimension bounded by $d_{VC}(\gamma) \leq R^2/\gamma^2$

Model selection

- We try to find the model with the best balance of complexity and the fit to the training data
- Ideally, we would select a model from a nested sequence of models of increasing complexity (VC-dimension)

  Model 1 d_1
  Model 2 d_2
  Model 3 d_3

  where $d_1 \leq d_2 \leq d_3 \leq \ldots$
- The model selection criterion is: find the model class that achieves the lowest upper *bound* on the expected loss

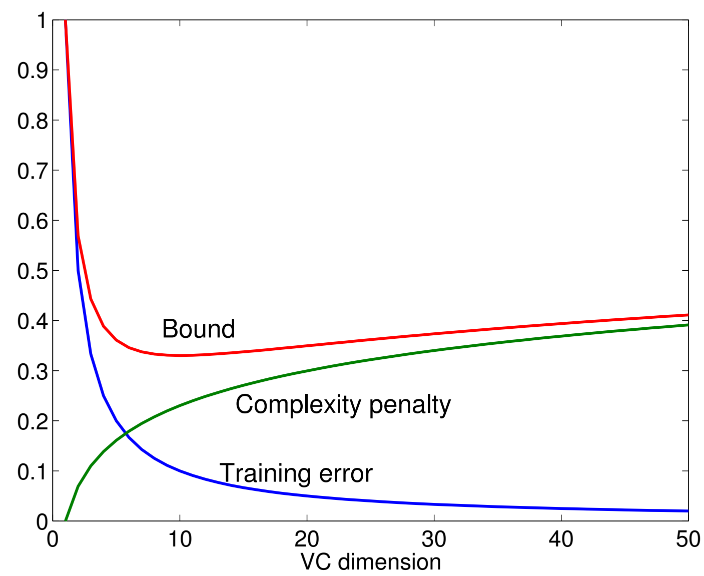
  Expected error $\leq$ Training error $+$ Complexity penalty

Structural risk minimization cont'd

- We choose the model class F_i that minimizes the upper bound on the expected error:

$$\mathcal{E}(\hat{h}_i) \leq \hat{\mathcal{E}}_n(\hat{h}_i) + \sqrt{\frac{d_i(\log(2n/d_i) + 1) + \log(1/(4\delta))}{n}}$$

where $\hat{h}_i$ is the best classifier from F_i selected on the basis of the training set.



Example

- Models of increasing complexity

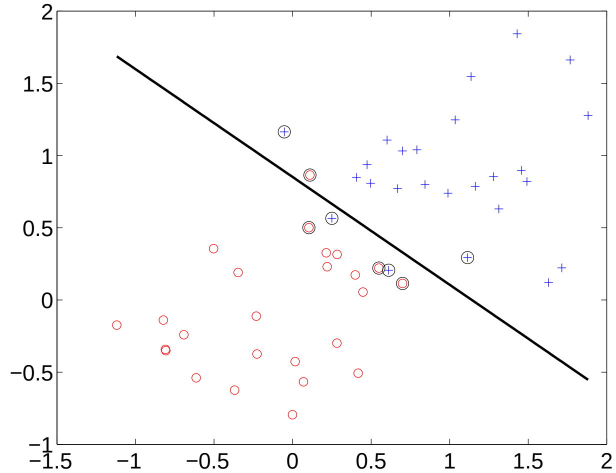
Model 1	$K(\mathbf{x}_1, \mathbf{x}_2) = (1 + (\mathbf{x}_1^T \mathbf{x}_2))$
Model 2	$K(\mathbf{x}_1, \mathbf{x}_2) = (1 + (\mathbf{x}_1^T \mathbf{x}_2))^2$
Model 3	$K(\mathbf{x}_1, \mathbf{x}_2) = (1 + (\mathbf{x}_1^T \mathbf{x}_2))^3$
...	...

- These are nested, i.e.,

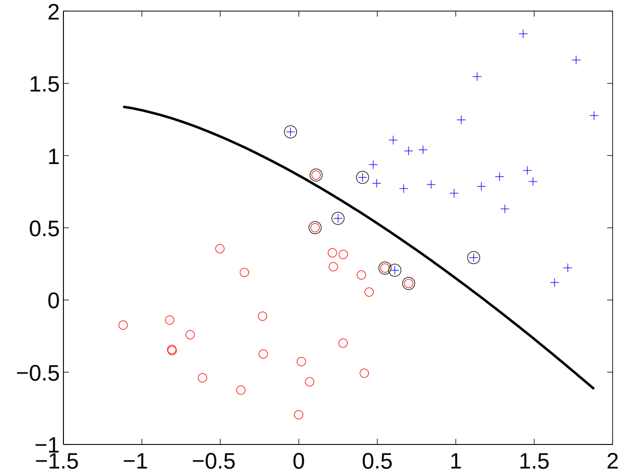
$$F_1 \subseteq F_2 \subseteq F_3 \subseteq \ldots$$

where F_k refers to the set of possible decision boundaries that the model k can represent.

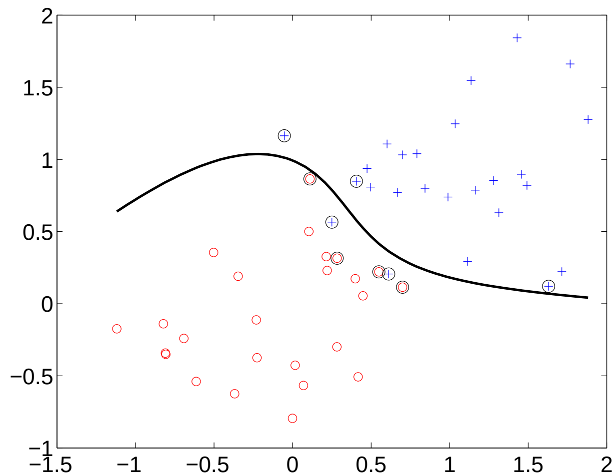
Structural risk minimization: example



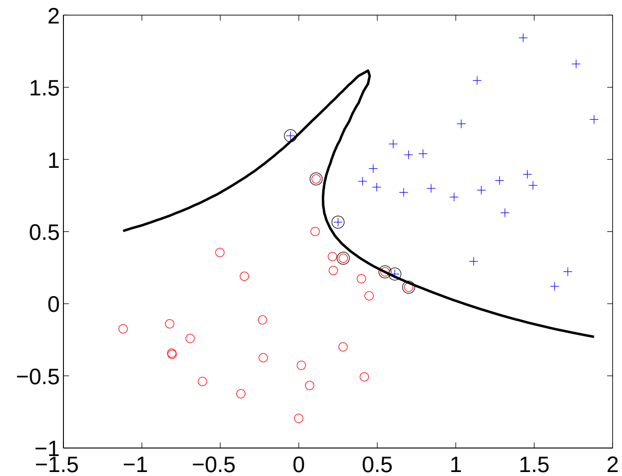
linear



2^{nd} order polynomial



4^{th} order polynomial



8^{th} order polynomial

Structural risk minimization: example cont'd

- Number of training examples $n = 50$, confidence parameter $\delta = 0.05$.

Model	d_{VC}	Empirical fit	$\epsilon(n, d_{VC}, \delta)$
1^{st} order	3	0.06	0.5501
2^{nd} order	6	0.06	0.6999
4^{th} order	15	0.04	0.9494
8^{th} order	45	0.02	1.2849

- Structural risk minimization would select the simplest (linear) model in this case.