COMP9444: Neural Networks

Vapnik Chervonenkis Dimension, PAC Learning and Structural Risk Minimization

How good a classifier does a learner produce?

- Training error is the precentage of incorrectly classified data among the training data.
- True error is the probability of misclassifying a randomly drawn data point from the domain (according to the underlying probability distribution).
- How does the training error relate to the true error?

How good a classifier does a learner produce?

- We can estimate the true error by using randomly drawn test data from the domain that was not shown to the classifier during training.
- Test error is the percentage of incorrectly classified data among the test data.
- Validation error is the percentage of incorrectly classified data among the validation data.

Calibrating MLPs for the learning task at hand

- Split the original training data set into two subsets:
  - Training data set
  - Validation data set
- Use the training data set for resampling to estimate which network configuration (e.g. how many hidden units to use) seems best.
- Once a network configuration is selected, train the network on the training set and estimate its true error using the validation set not shown to the network at any time before.

Resampling methods

- Resampling tries to use the same data multiple times to reduce estimation errors on the true error of a classifier being learned using a particular learning method.
- Frequently used resampling methods are
  - *n*-fold cross validation (with *n* being 5 or 10)
  - and bootstrapping.

Cross Validation

- Split available training data into n subsets $s_1, \ldots, s_n$.
- For each i set apart subset s_i for testing. Train the learner on all remaining $n-1$ subsets and test the result on s_i.
- Average the test errors over all n learning runs.

Bootstrapping

- Assume the available training data represents the actual distribution of data in the underlying domain. i.e. for k data points, assume each data point occurs with probability $1/k$.
- Draw a set of t training data points by randomly selecting a data point from the original collectionwith probability $1/k$ for each data point (if there are multiple occurrences of the same data point, the probability of that data point occurring is deemed to be proportionally higher.)
- Often t is chosen to be equal to k. This will usually result in having only about 63% of the original sample, while the remaining 37% $(1/e)$ are made up of duplicates

Bootstrapping

- Train learner on the randomly drawn training sample.
- Test the learned classifier on the items from the original data set which were not in the training set.
- Repeat the random selection, training and testing for a number of times and average the test error over all runs (typical number of runs is 30 to 50.)

Factors Determining the Difficulty of Learning

- The representation of domain objects, i.e. which attributes and how many.
- The number of training data points.
- The learner, i.e. the class of functions it can potentially learn.

PAC Learning

- We know a number of heuristics to ensure learning on the training set will carry to new data, such as using a minimally complex model, training on a representative dataset, with sufficient samples
- How can we address this issue in a principled way?
- Probably Approximately Correct learning, and Structured Risk Minimisation

PAC Learning

- Want to ensure the probability of poor learning is low

$$P[|E_{out} - E_{in}| > \varepsilon] < 1 - \delta$$

- If $|E_{out} - E_{in}|$ is low, our learner is “approximately correct”. This function provides a bounds on the probability that this is the case.
- Hoeffding inequality: probability that the sum of random variables deviates from expected values

$$P[|\nu - \mu| > \varepsilon] \leq 2e^{-2\varepsilon^2 N}$$

- For a given hypothesis h, we can say

$$P[|E_{out}(h) - E_{in}(h)| > \varepsilon] \leq 2e^{-2\varepsilon^2 N}$$

PAC Learning

- For a chosen hypothesis g, drawn from the set of hypotheses H:

$$P[|E_{out}(g) - E_{in}(g)| > \varepsilon] \leq 2Me^{-2\varepsilon^2 N}$$

- M represents the number of hypotheses, ∞ for a perceptron
- This is not very useful. Alternative, we can see how the value grows with the number of samples.

$$m_H(N) = \max_{x_1 \ldots x_N} |H(x_1 \ldots x_N)|$$

- The growth function represents the most dichotomies that H implements over all possible samples of size N

$$m_H(N) \leq 2^N$$

Vapnik-Chervonenkis dimension

- H *shatters* N points, if there exists an arrangement of N points such that for all arrangements of labels on the points, there is a hypothesis h that captures the labels
- Vapnik-Chervonenkis dimension $d_{VC}(H)$ is the largest N that H can shatter (and where $m_H(N) = 2^N$).
- If $N \leq d_{VC}$, H may be able to shatter the data. If $N > d_{VC}$ H *cannot* shatter the data.
- Bounds: it can be proved that $m_H(N) \leq N^{d_{VC}} + k$

$$P[|E_{out} - E_{in}| > \varepsilon] \leq 4m_H(2N)e^{\frac{-\varepsilon^2 N}{8}}$$

Vapnik-Chervonenkis dimension

- Definition
- Various function sets - relevant to learning systems - and their VC-dimension
- Bounds from probably approximately correct learning (PAC-learning).
- General bounds on the Risk.

The Vapnik-Chervonenkis dimension

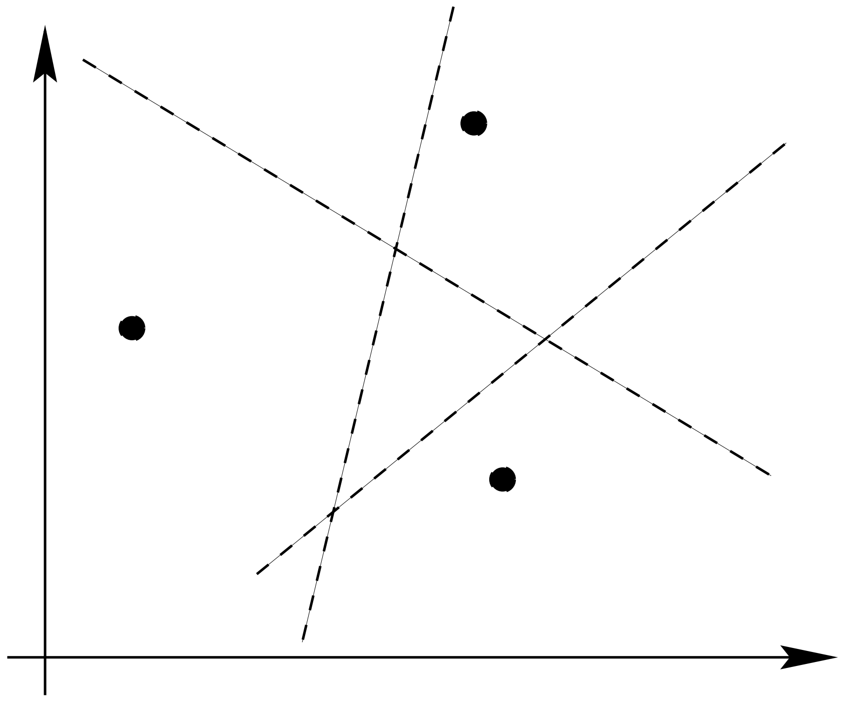
In the following, we consider a boolean classification function f on the input space X to be equivalent to the subset of X such that $\forall x \in X \ \ f(x) = 1$.

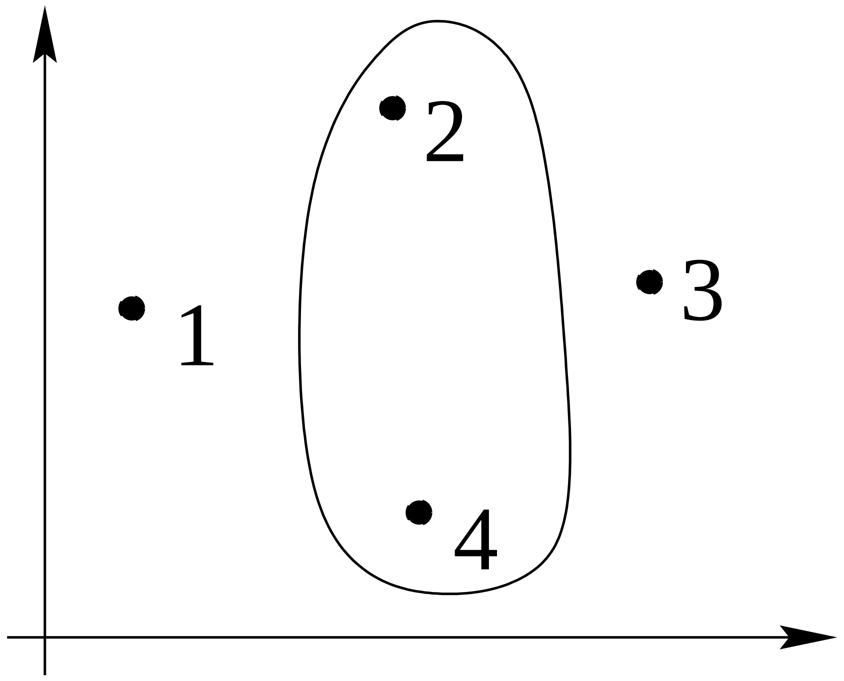
The VC-dimension is a useful combinatorial parameter on sets of subsets in the input space, e.g. on function classes H on the input space X.

Definition We say a set $S \subseteq X$ is **shattered** by H, if for all subsets $s \subseteq S$ there is at least one function $h \in H$ such that $s = h \cap S$.

Definition The Vapnik-Chervonenkis dimension of H, $d_{VC}(H)$ is the cardinality of the largest set $S \subseteq X$ shattered by H.

The Vapnik-Chervonenkis dimension





The VC-dimension of the set of linear decision functions in the 2-dimensional Euclidean space is 3.

The Vapnik-Chervonenkis dimension

The VC-dimension of the set of linear decision functions in the n-dimensional Euclidean space is equal to n+1.

VC-dimension and PAC-Learning

Theorem Upper Bound (Blumer et al., 1989)

Let L be a learning algorithm that uses H consistently, i.e. that finds an $h \in H$ that is consistent with all the data. For any $0 < \varepsilon, \delta < 1$ given

$$N = \frac{(4\log(\frac{2}{\delta}) + 8d_{VC}(H)\log(\frac{13}{\varepsilon}))}{\varepsilon}$$

random examples, L will with probability of at least $1 - \delta$

either produce a function $h \in H$ with error $\leq \varepsilon$

or indicate correctly, that the target function is not in H.

VC-dimension and PAC-Learning

Theorem Lower Bound (Ehrenfeucht et al., 1992)

Let L be a learning algorithm that uses H consistently. For any $0 < \varepsilon < \frac{1}{8}$, $0 < \delta < \frac{1}{100}$ given less than

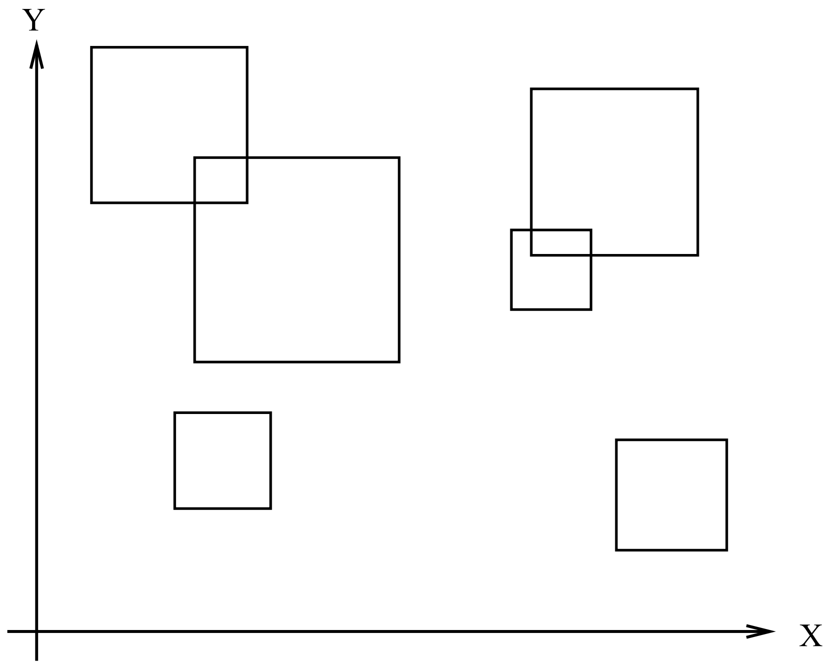
$$N = \frac{d_{VC}(H) - 1}{32\varepsilon}$$

random examples, there is some probability distribution for which L will **not** produce a function $h \in H$ with $error(h) \leq \varepsilon$ with probability $1 - \delta$.

VC-dimension bounds

ε	δ	d_{VC}	lower bound	upper bound
5%	5%	10	6	9192
10%	5%	10	3	4040
5%	5%	4	2	3860
10%	5%	4	1	1707
10%	10%	4	1	1677

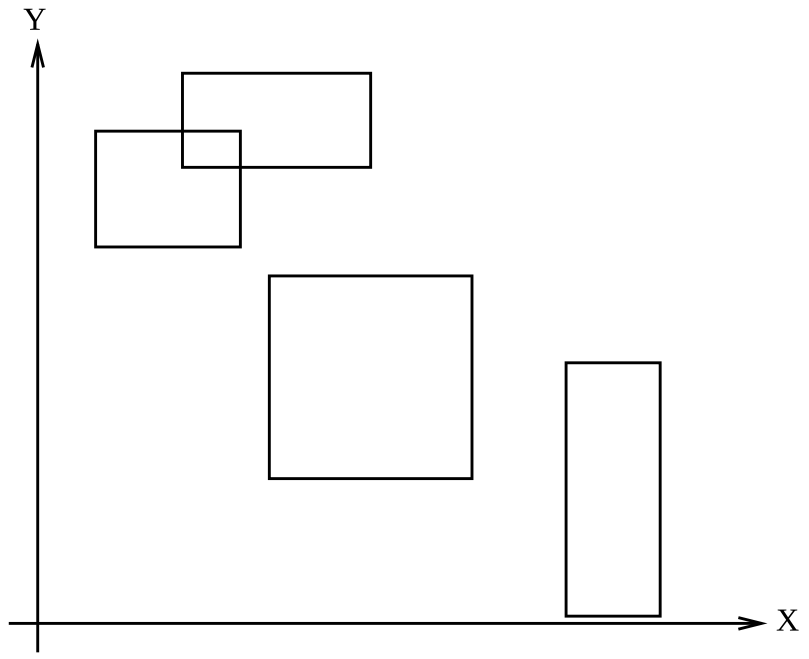
The Vapnik-Chervonenkis dimension



Let C be the set of all squares in the plane (parallel to axis).

VC-dim = . . .

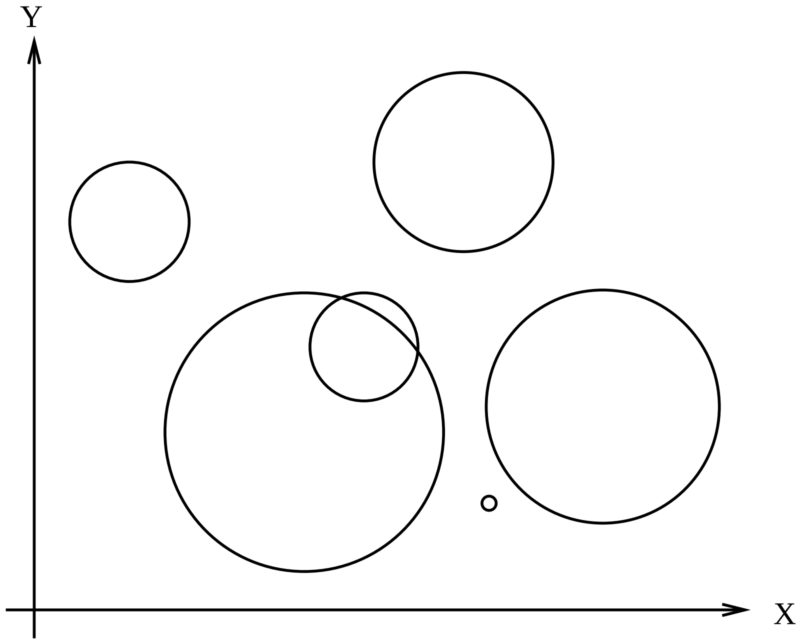
The Vapnik-Chervonenkis dimension



Let C be the set of all rectangles in the plane (parallel to axis).

VC-dim = . . .

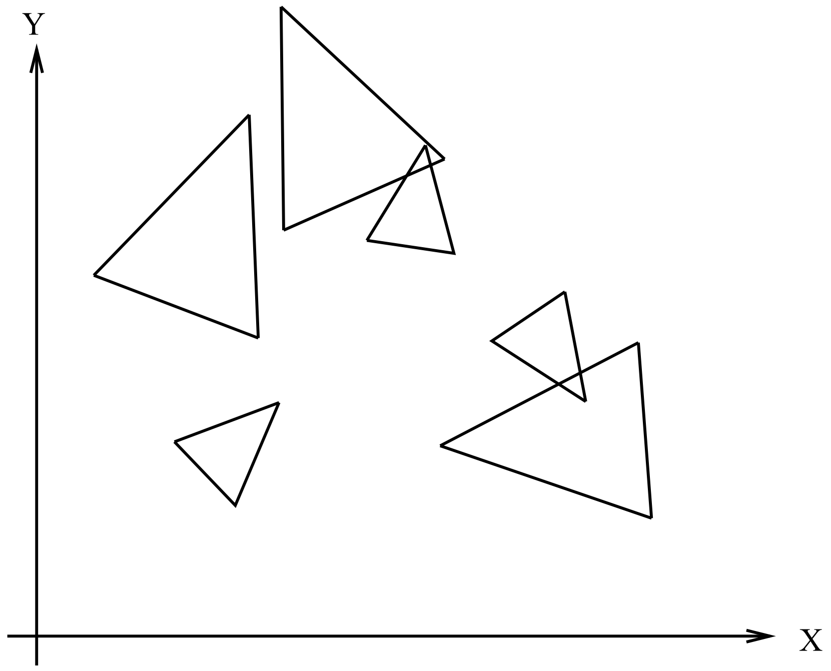
The Vapnik-Chervonenkis dimension



Let C be the set of all circles in the plane.

VC-dim = …

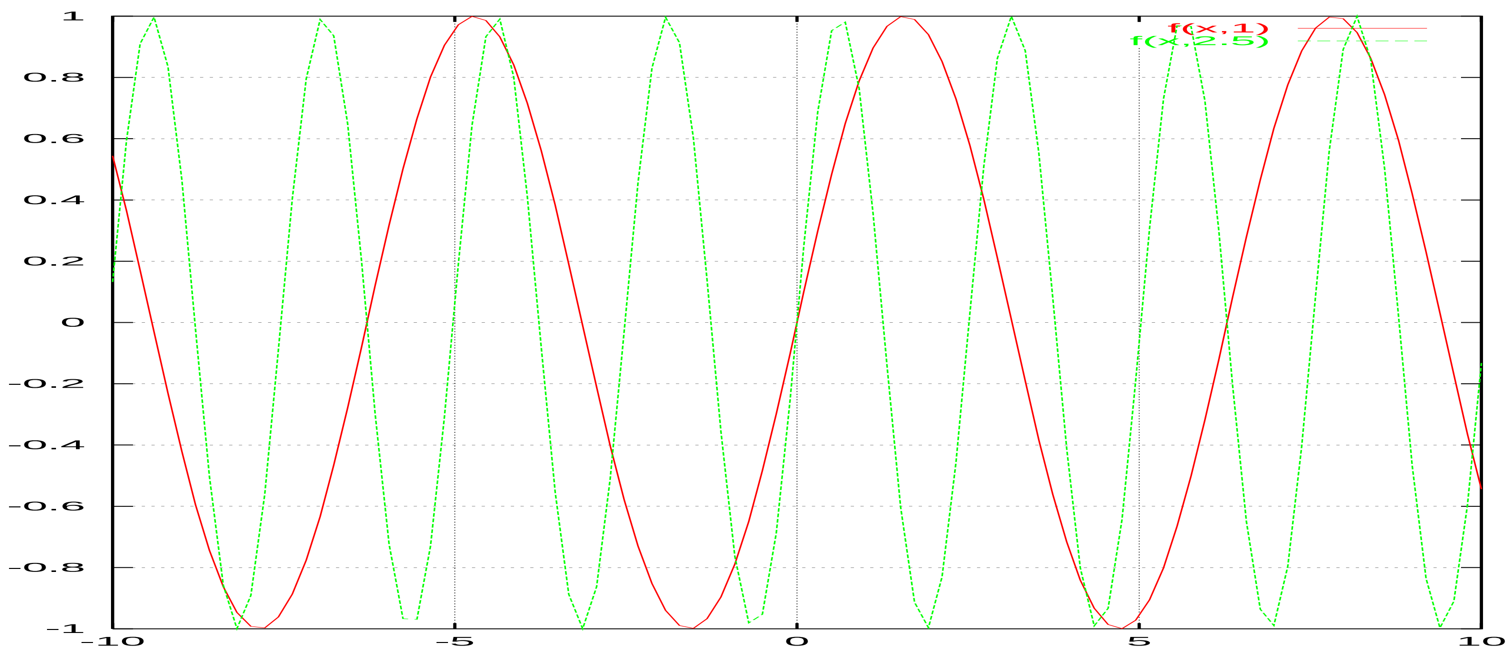
The Vapnik-Chervonenkis dimension



Let C be the set of all triangles in the plane, allowing rotation.

VC-dim = ...

The Vapnik-Chervonenkis dimension



The VC-dimension of all functions of the following form is inifinite!

$$f(x,\alpha) = sin(\alpha\, x),\ \alpha \in \Re.$$

The Vapnik-Chervonenkis dimension

The following points on the line

$$x_1 = 10^{-1}, \ldots, x_n = 10^{-n}$$

can be shattered by functions from this set.

To separate these data into two classes determined by the sequence

$$\delta_1, \ldots, \delta_n \in \{0, 1\}$$

it is sufficient to choose the value of parameter

$$\alpha = \pi(\sum_{i=1}^{n}(1 - \delta_i)10^i + 1)$$

VC-Dimensions of Neural Networks

As a good heuristic (Bartlett):

VC-dimension $\approx$ number of parameters

- For linear threshold functions on $\Re^n$, $d_{VC} = n+1$.
  (number of parameters is $n+1$.)
- For linear threshold networks, and fixed depth networks with piecewise polynomial squashing functions

$$c_1|\vec{W}| \leq d_{VC} \leq c_2|\vec{W}|\log|\vec{W}|$$

  where $|\vec{W}|$ is number of weights in the network.
- Some threshold networks have $d_{VC} \geq c|\vec{W}|\log|\vec{W}|$.
- $d_{VC}(\text{sigmoid net}) \leq \mathrm{c}|\vec{\mathrm{W}}|^4$

VC-Dimensions of Neural Networks

Any function class H that can be computed by a program that takes a real input vector $\vec{x}$ and k real parameters and involves no more than t of the following operations:

- $+, -, \times, /$ on real numbers
- $>, \geq, =, \leq, <, \neq$ on real numbers
- output value $y \in \{-1, +1\}$

has VC-dimension of $O(kt)$.

(See work of Peter Bartlett)

VC-Dimensions of Neural Networks

Any function class H that can be computed by a program that takes a real input vector $\vec{x}$ and k real parameters and involves no more than t of the following operations:

- $+, -, \times, /, e^{\alpha}$ on real numbers
- $>, \geq, =, \leq, <, \neq$ on real numbers
- output value $y \in \{-1, +1\}$

has VC-dimension of $O(k^2 t^2)$.

This includes sigmoid networks, RBF networks, mixture of experts, etc.

(See work of Peter Bartlett)

Structural Risk Minimisation (SRM)

The complexity (or capacity) of a function class from which the learner chooses a function that minimises the empirical risk (i.e. the error on the training data) determines the convergence rate of the learner to the optimal function.

For a given number of independently and identically distributed training examples, there is a trade-off between the degree to which the empirical risk can be minimised and to which the empirical risk will deviate from the true risk (i.e. the true error - error on unseen data according to the underlying distribution).

Structural Risk Minimisation (SRM)

The general SRM principle:

Choose complexity parameter d, e.g. the number of hidden units in a MLP, or the size of a decision tree, and function $g \in H$ such that the following is minimised:

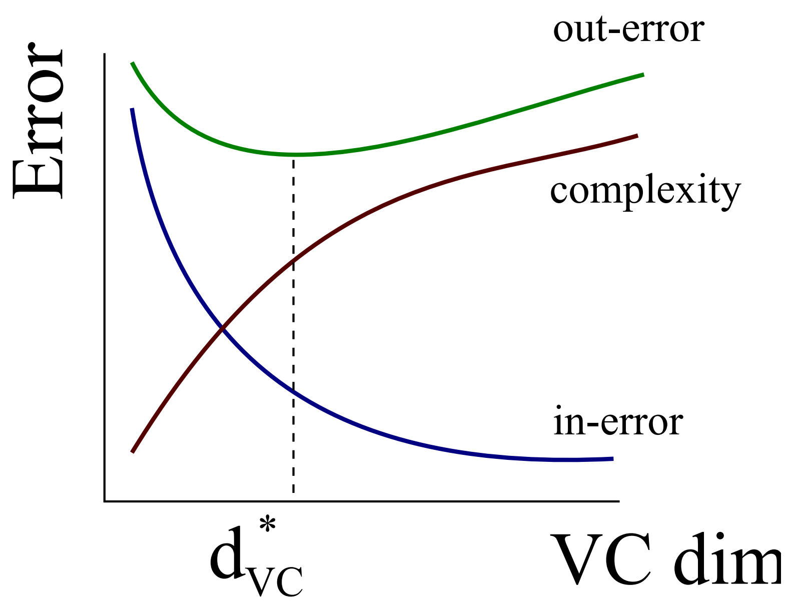
$$E_{out}(g) \leq E_{in}(g) + c\sqrt{\frac{d_{VC}(H)}{N}}$$

where N is the number of training examples.

The higher the VC dimension is the more likely will the empirical error be low.

Structural risk minimisation seeks the right balance.

Structural Risk Minimisation (SRM)



$$
\begin{aligned}
E_{out} &\leq E_{in} + \Omega \\
E_{out} &\leq E_{in} + \sqrt{\frac{8}{N} log \frac{4m_H(2N)}{1-\delta}} \\
m_H(N) &\leq N^{d_{VC}(H)} \\
E_{out} &\leq E_{in} + c\sqrt{\frac{d_{VC}(H)}{N}}
\end{aligned}
$$

VC-dimension Heuristic

For neural networks and decision trees:

VC-dimension $\approx$ size.

Hence, the order of the **misclassification probability** is no more than

$$\text{training error} + \sqrt{\frac{\text{size}}{\text{m}}}$$

where *m* is the number of training examples.

This suggests that the number of training examples should grow roughly linearly with the size of the hypothesis to be produced.

If the function to be produced is too complex for the amount of data available, it is likely that the learned function is not a near-optimal one. (See work of Peter Bartlett)

Summary

- The VC-dimension is a useful combinatorial parameter of sets of functions.
- It can be used to estimate the true risk on the basis of the empirical risk and the number of independently and identically distributed training examples.
- It can also be used to determine a sufficient number of training examples to learn probably approximately correctly.
- Applications of the VC-dimension for choosing the most suitable subset of functions for a given number of independently and identically distributed examples.
- Trading empirical risk against confidence in estimate.

Summary (cont.)

Limitations of the VC Learning Theory

- The probabilistic bounds on the required number of examples are worst case analysis.
- No preference relation on the functions of the learner are modelled.
- In practice, learning examples are not necessarily drawn from the same probability distribution as the test examples.