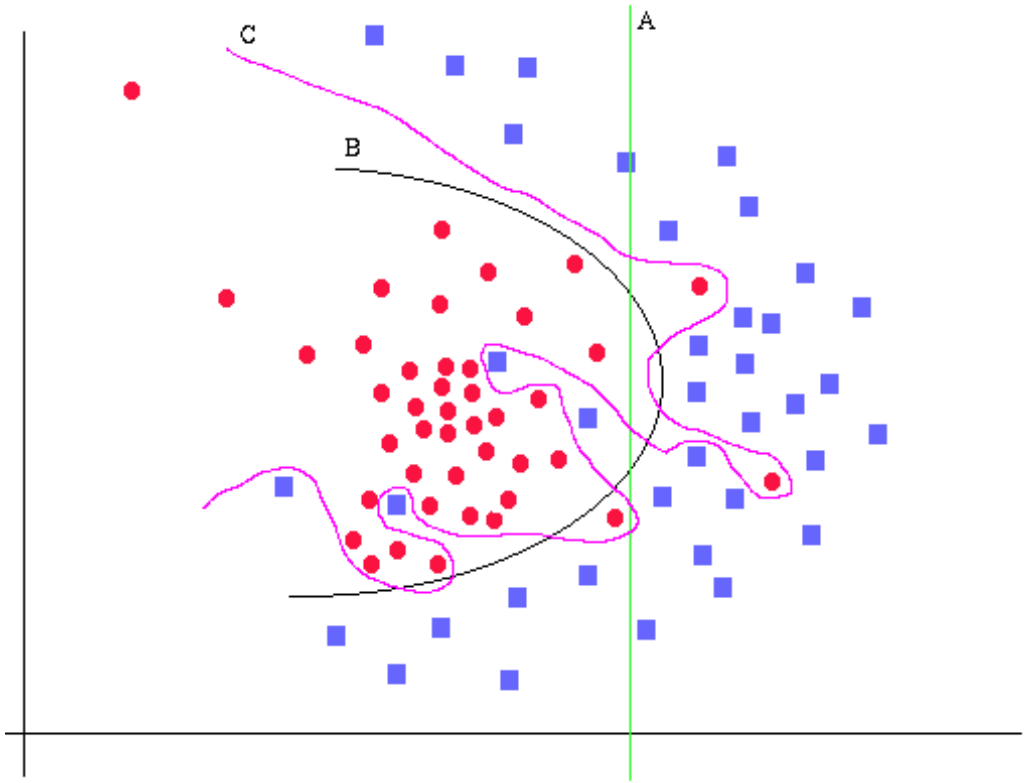
Overfitting

COMP4211

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

Example



which one is the best?

Training Error

given:

- target function f
- hypothesis h (e.g., a particular neural network)
- distribution $\mathcal{D}$ of the instances
- training set S (of size n) drawn from $\mathcal{D}$

training error

- proportion of examples in S that h misclassifies

$$error_S(h) \equiv \frac{1}{n} \sum_{x \in S}^{n} \delta(f(x) \neq h(x))$$

  - $\delta(f(x) \neq h(x))$ is 1 if $f(x) \neq h(x)$, and 0 otherwise
- what we can measure

Testing Error

testing error

- probability that h will misclassify an instance drawn at random according to $\mathcal{D}$

$$error_{\mathcal{D}}(h) \equiv Pr_{x \in \mathcal{D}}[f(x) \neq h(x)]$$

- what we wish to know

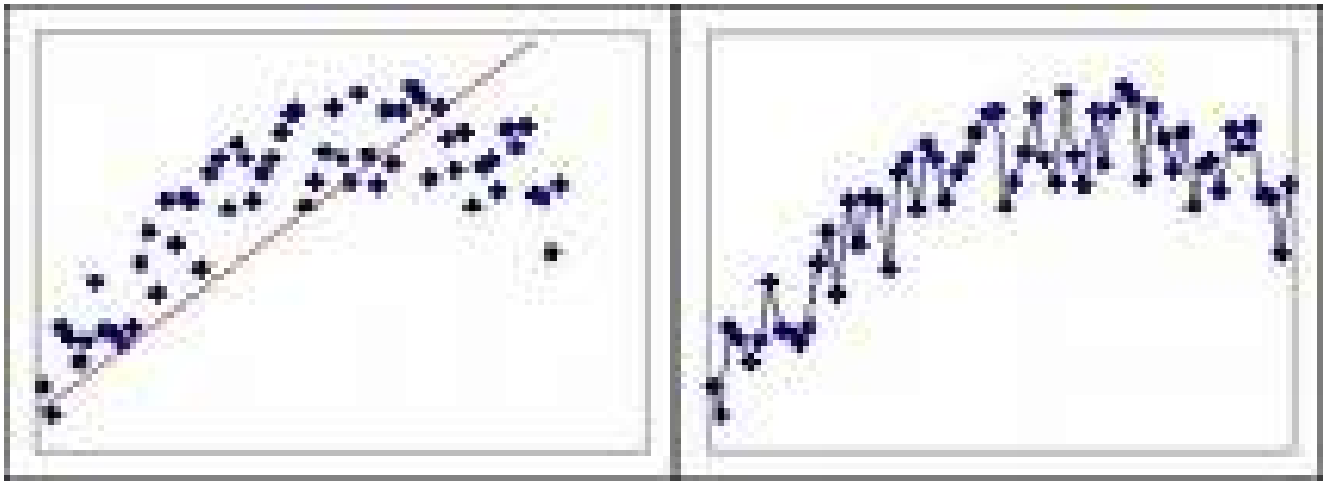
typically, use a test set to estimate the testing error

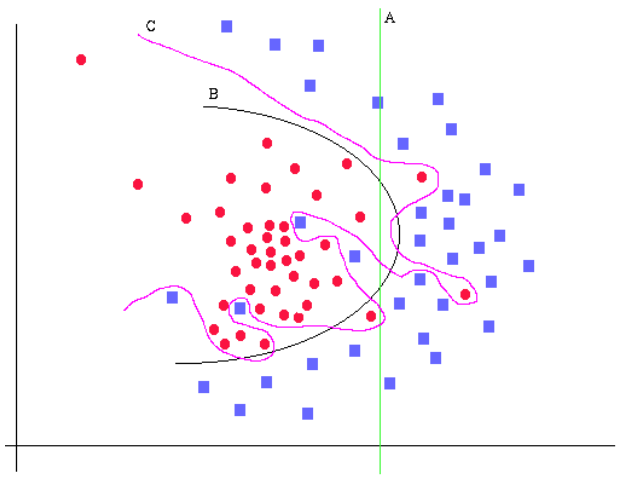
$$error_{test}(h) \equiv \frac{1}{|testSet|} \sum_{x \in testSet}^{n} \delta(f(x) \neq h(x))$$

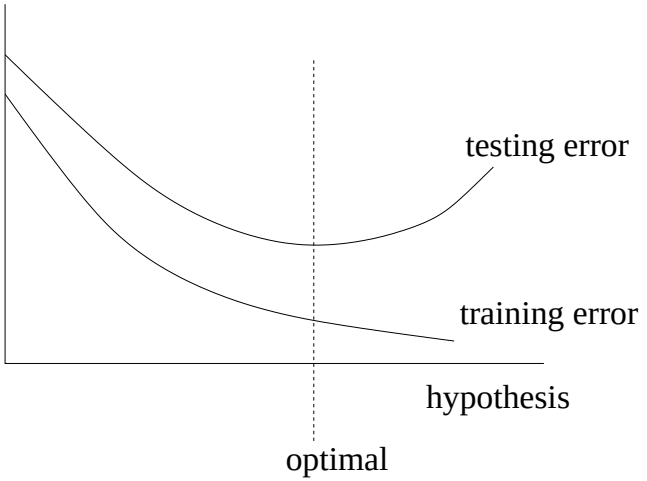
- the test patterns should be drawn independently from the training patterns
- the testing data should not be used in any way to learn the hypothesis

How to Select the "Best" Hypothesis?

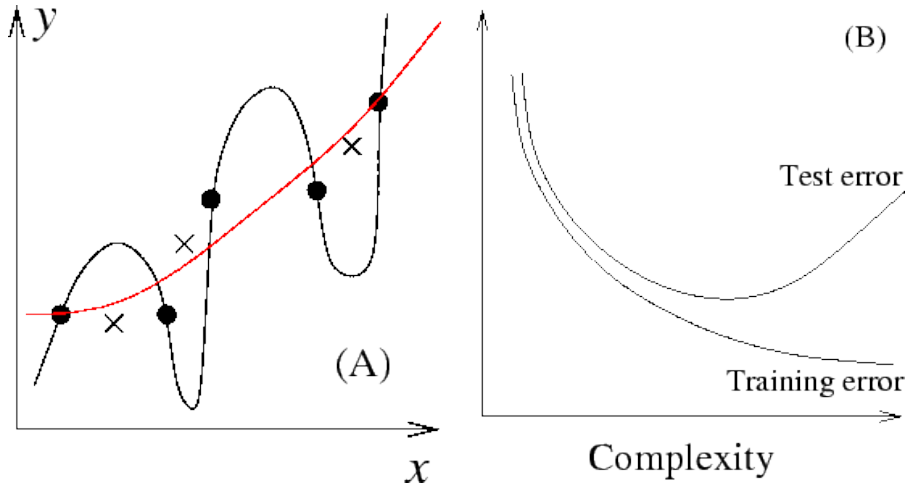
minimize the training error?







Overfitting

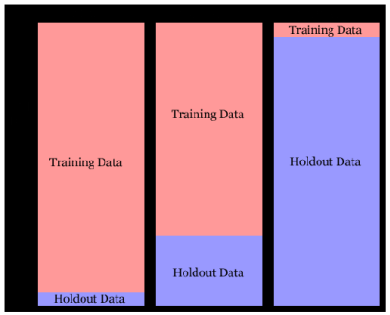


hypothesis $h \in H$ overfits the training data if there is an alternative hypothesis $h' \in H$ such that

- h has smaller error than h' over the training examples, but
- h has a larger error than h' over the entire distribution of instances

How to Select the "Best" Hypothesis?...

- the test data should not be used!
- partition the available (training) data into two sets
  - training set: used to form the learned hypothesis
  - validation set: used to estimate the accuracy of this hypothesis over subsequent data

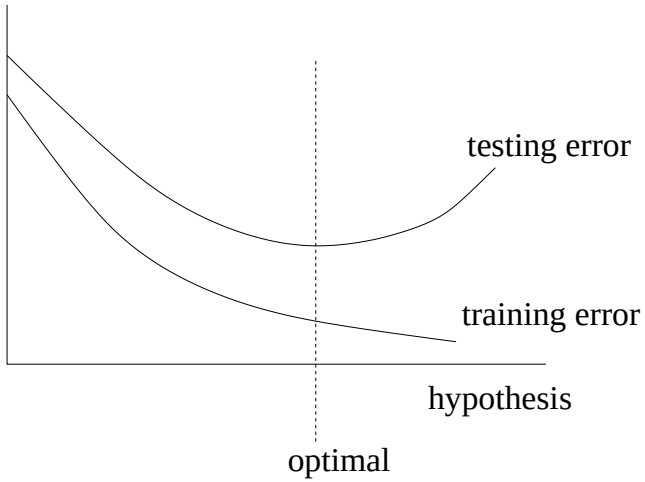


- once the evaluation is complete, all the data can be used to train the final hypothesis (optional)
- generally,
  - the larger the training set, the better the hypothesis
  - the larger the validation set, the more accurate the error estimation
- typically, withhold one-third of the available examples for the validation set, using the other two-thirds for training

How to Avoid Over-fitting?

early stopping

- stops before it reaches the point where it perfectly classifies the training data



pruning

Occam's Razor

- prefer simple hypotheses

why?

- fewer simple hypotheses than complicated hypotheses
- a simple hypothesis that fits data unlikely to be coincidence
- a complicated hypothesis that fits data might be coincidence

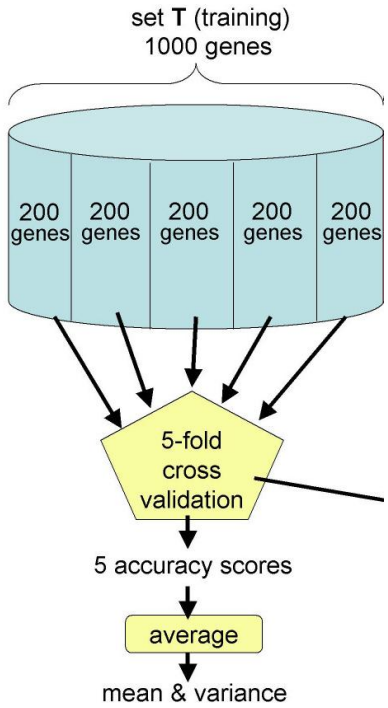
No Separate Validation Set?

when data is limited, withholding part of it for validation set reduces even further the number of examples available for training

how to make the error estimation more reliable?

- repeat the process with different subsamples

k-fold Cross-validation



1. the m available examples are partitioned into k disjoint subsets, each of size m/k
2. the learning procedure is then run k times, each time
   - using one of these subsets as the validation set, and
   - combining the other subsets for the training set
3. average the performance on the validation sets over the k runs

k-fold Cross-validation...

each example is used

- in the validation set once, and
- in the training set for the other $k-1$ folds

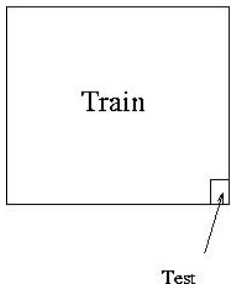
a high proportion of the available data $(1-\frac{1}{k})$ is used for training, while making use of all the data in computing the error

how many times do we need to perform training?

- typically, $k \sim 10$ is considered reasonable

leave-one-out cross-validation ($k = m$)

- train on $m-1$ examples and validate on 1 example
- useful for small data sets



k-fold Stratified Cross-validation

another problem:

- examples in the training set of each fold may not be representative
- e.g., all the examples of a certain class are missing
- $\rightarrow$ the classifier cannot learn to predict this class

how to ensure that each class is represented with approximately equal proportions in both the training and validation sets?

- partition the m examples into k folds such that each class is uniformly distributed among the k folds
- the class distribution in each fold is similar to that in the original data set
- typically, use 10-fold (stratified) cross-validation