# Computational Intelligence: Methods and Applications

Lecture 15

Model selection and tradeoffs.

Włodzisław Duch
SCE, NTU, Singapore
Google: Duch

## How to select a good model?

Optimal balance between variance and bias should be found.

Some *a priori* knowledge may be included in the design of the model, either specific knowledge (leading to parametric models), or quite general knowledge, like: try to increase smoothness of a function.

Regularization techniques are mathematical methods developed in the theory of function approximation to select bias and decrease variance of models using general *a priori* knowledge.

Many sophisticated model selection theories exist.

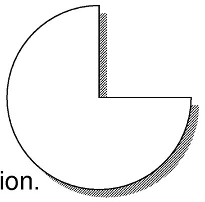
Overfitting on the training data leads to an increase of the MSE on the test or validation set. Therefore a simple approach is to learn from the training data, but evaluate and select model on separate validation data set. This leads to crossvalidation techniques.

## Crossvalidation

If the data is large one may learn from training set and evaluate results on validation set, but in practice rarely there is sufficient number of data samples. CV is general model selection/evaluation procedure.

CV uses training data for validation:

- randomize the training data set T
- divide it into k subsets $\{T_i\}$, i=1..k
- train using subsets $\{T-T_1\}$, use $T_1$ for validation
- repeat using subsets $\{T-T_i\}$, i=2 .. k, use $T_i$ for validation.



This procedure is called k-fold cross-validation (k-CV), or hold-out.

Select model parameters that minimize average CV hold-out error.

Error on the training data (MSE, or the number of errors) is called "resubstitution error" or in-sample error, and if the model has a proper bias and does not overfit the data it may be close to the error on the validation partitions, called out-of-sample or CV-test error.

## More CV

The training error rate is also called "in-sample" error.

The error on the validation partition is "out-of-sample" error.

If class proportions are kept in the $T_i$ subsets CV is called "stratified".

If k=N then the procedure is called leave-one-out CV (training).

Frequently 10-fold CV is used since leave-one-out is too expensive.

Statisticians recommend averaging to get a better estimation of generalization error, ex. 5 times 2-fold crossvalidation (5x2CV).

Assumption: test and training data should be representative samples of the underlying data distribution – this is sometimes violated, for example there may be regional differences in population data.

Final test should be done on data that **has not been used in any way** for parameter adaptation and model selection.

## CV use and properties.

How is the CV procedure used?
If a single validation data set is provided, than model parameters are learned from the training data, and the quality of the models is compared using the validation data.

If a number of numerical experiments using validation set is large the resulting model may overfit the validation data, even though this data is not directly used for training.

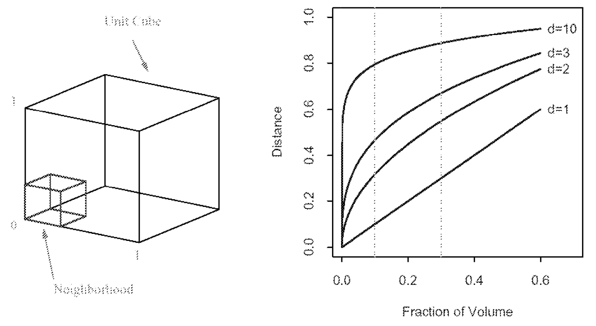
Crossvalidation selects the validation partitions randomizing the data, giving better estimation of generalization error than just validation.
CV is used to:

- compare quality of models by estimating expected statistical accuracy;
- find parameters (ex. pruning of trees) that give good generalization;

Final model is build using the whole training data with parameters determined from CV; this model is applied to the new (test) data.

## Curse of dimensionality

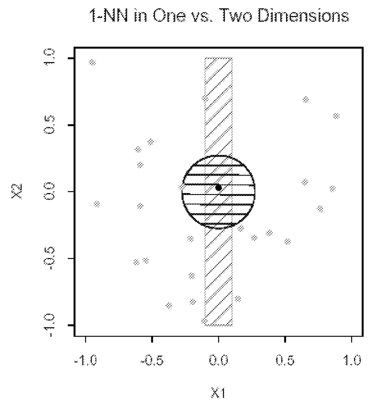
Neighborhood volumes and the number of samples shrink to zero when the number of dimensions is increased; for *d*=10 filling 10% of the unit cube volume requires already 0.8 side length!



High-D space is almost always empty! Hypercube is like a hedgehog!
See the paper: Mario Köppen, The curse of dimensionality.

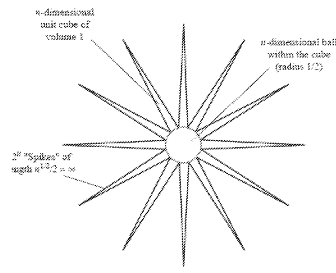
## 1D and 2D case

For any 1D projection distances between neighbors are quite small.
The average radius covered by a single training point rapidly grows with the number of dimensions, as shown in the figure below.



Ref: Fig. 2.7, from Friedman, Hasti & Tibshirani 2001.

A ball in hypercube in d dim, from:
R. Hecht-Nielsen, Neurocomputing

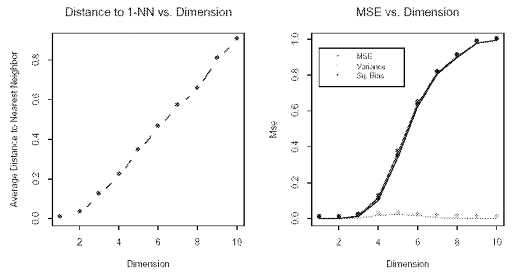

n-dimensional unit cube of volume 1
n-dimensional ball within the cube (radius 1/2)
$2^n$ "Spikes" of length $n^{1/2}/2 \to \infty$

## Curse and bias

The average distance grows approximately linearly with the number of dimensions. The MSE also grows due to the growing bias term – taking just one neighbor is not a good strategy in high dimensions.

Fig. 2.7 from

Friedman, Hasti & Tibshirani 2001.



Inputs are randomly distributed in $[-1,+1]^d$, for d=1 ..10, the task is to approximate the target functions $f(X)=\exp(-8||X||^2)$. MSE is the error of approximating f(0), and the $\text{Bias}^2$ show on the right side is calculated using the formula from Lecture 14.

## What can we do?

Try to reduce dimensionality:

- using information selection methods;
- using feature aggregations (PCA, ICA, FDA, or kernel-based);

or

- select model with appropriate bias for the data;
- try to maximize margins (SVM approach), not only minimize errors.

Select simplest solutions, smooth approximation surfaces, simple decision borders or simple rules.

Be careful! Do not believe what you find ...
It is probably an illusion created by lack of data.

## Model tradeoffs

There are two main tradeoffs in model selection:

- **Simplicity vs. accuracy**.

Simple model may be understandable and preferred by the user;

the highest achievable accuracy may require more complex model which may not be so transparent.

Models that are more complex than necessary to achieve best accuracy should not be used.

- **Confidence vs. rejection rate**.

Rejecting many cases close to decision border gives growing confidence in correctness of predictions.
Perhaps these cases require more tests or more specific evaluation.

## But ... what is better?

Overall accuracy does not tell us too much in practical applications.
What types of errors are made? What are their costs?

Predicted class $\omega_{M1}$ may not match the true class $\omega_2$,
or predicted class $\omega_{M2}$ may not match the true class $\omega_1$.
For example, predictions by tossing a coin are:

$$P(\omega_1 | \omega_{M1}) = 0.25; P(\omega_1 | \omega_{M2}) = 0.25;$$
$$P(\omega_2 | \omega_{M1}) = 0.25; P(\omega_2 | \omega_{M2}) = 0.25;$$

This is of course useless, with accuracy 0.5, but if:

$$P(\omega_1 | \omega_{M2}) = 0.00; P(\omega_2 | \omega_{M1}) = 0.50;$$

accuracy is still 0.5, but $\omega_1$ cases are never mistaken for $\omega_2$ while all $\omega_2$ cases are always assigned to $\omega_1$. So, in this case we know something.

Evaluation of how good the model is needs careful analysis.