Model complexity and Cross validation

Jukka Kohonen

Simple and complex models

In statistical modelling, there is always a tradeoff between simple and complex models.

Generally . . .

- a complex model has many free parameters / is very flexible
- a simple model has few free parameters / is very rigid

Examples of complex models . . .

- multiple linear regression from many explanatory variables
- ANOVA/regression with many interaction terms
- polynomial regression with high degree (this demo)
- piecewise regression with many pieces
- models with rich structure, many submodels
- . . .

Some extreme cases

Multiple linear regression, n observations from p explanatory variables: if $p \geq n - 1$, you can always fit a linear model "perfectly" to the data.

- $n = 2$, $p = 1$: Fitting a straight line to two points.
- $n = 3$, $p = 2$: Fitting a plane to three points in space.

Univariate polynomial regression: Quite similarly, you can always fit a $(n - 1)$-degree polynomial "perfectly" to n points.

The tradeoff

- Underfitting: simple model fail to capture the real-world phenomenon that we are trying to understand.
  (What if the drug effect is *not* quite linear?)
- Overfitting: a complex model can capture many things, including noise, and tries to explain even that.
  An extremely complex model can “explain” the observed data perfectly, but fails to generalize to situations outside the data.

Trouble: We don’t necessarily *know* what is the right complexity for our phenomenon.

Model selection

Trying to solve the simple–complex tradeoff leads to the problem of model selection.

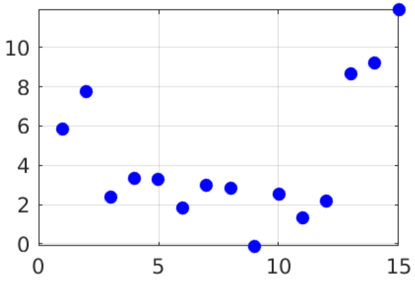
Typical solutions include . . .

- gather more data
- traditional rules of thumb ("should have fewer parameters than data points", $p \ll n$)
- penalize complex models (e.g. various "information criteria", AIC, BIC)
- cross-validation: keep some of the data "hidden" when fitting a model

Example: Polynomial regression

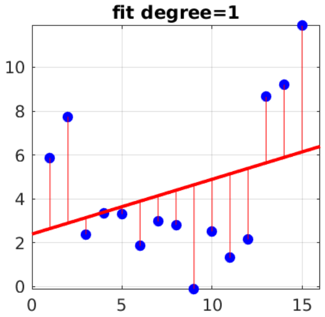
We have 15 points of univariate data that follows a polynomial $f(x)$ of unknown degree; and additive normal errors ε_i.

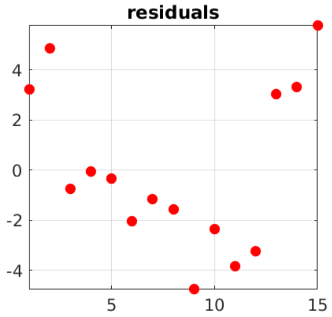
$$y_i = f(x_i) + \varepsilon_i$$



Example: Polynomial regression

We could try linear regression.

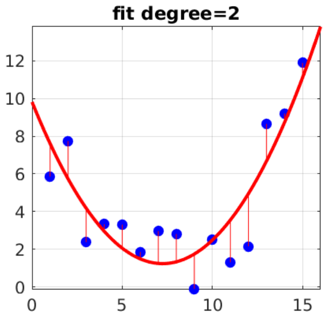


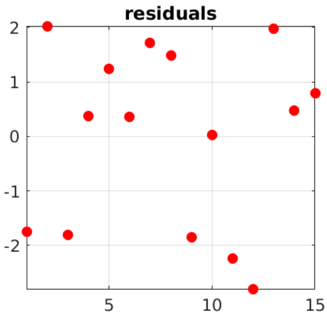


MSE = mean square residual = 3.1513

Example: Polynomial regression

We could try quadratic regression: use x_i and x_i^2 as explanatory variables.

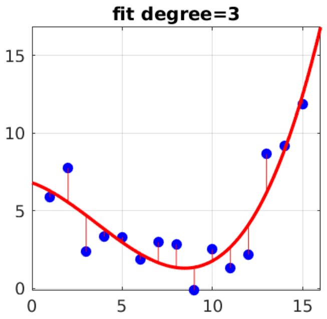


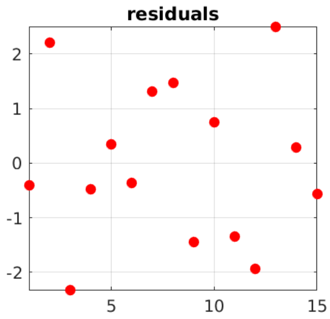


MSE = mean square residual = 1.5989 (smaller than before)

Example: Polynomial regression

We could try degree-3 regression: use x_i, x_i^2 and x_i^3 as explanatory variables.

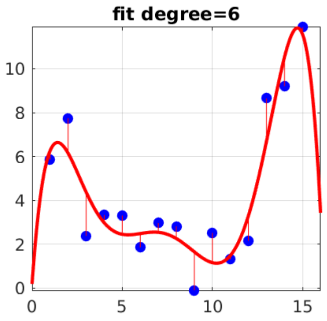


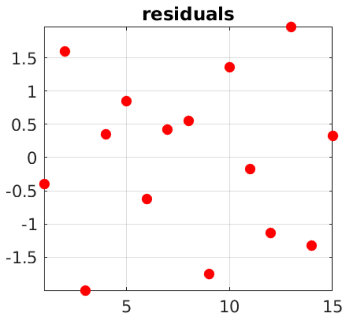


MSE = mean square residual = 1.4059 (smaller than before)

Example: Polynomial regression

We could try degree-6 regression: use $x_i, x_i^2, \ldots, x_i^6$ as explanatory variables.

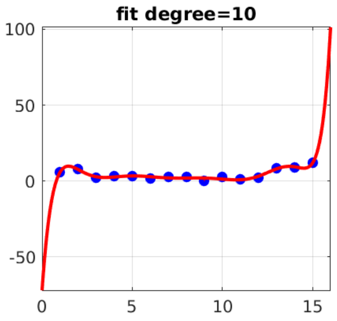


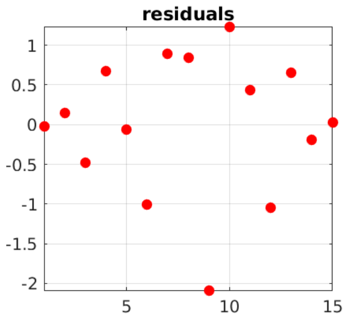


MSE = mean square residual = 1.1659 (smaller than before)

Example: Polynomial regression

We could try degree-10 regression: use $x_i, x_i^2, \ldots, x_i^{10}$ as explanatory variables.

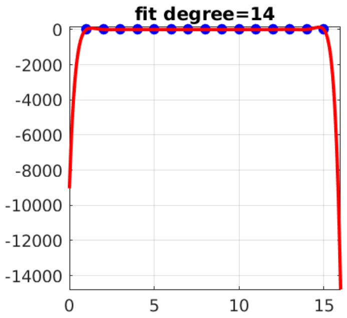


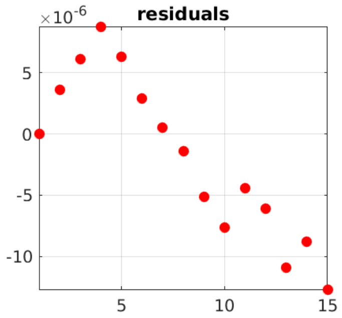


MSE = mean square residual = 0.8507 (smaller than before)

Example: Polynomial regression

We could try degree-14 regression: use $x_i, x_i^2, \ldots, x_i^{14}$ as explanatory variables.

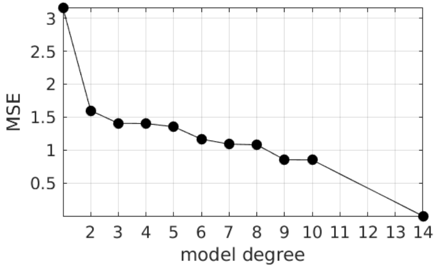




MSE = mean square residual = 0, curve fits data perfectly.
(The nonzero values in the residual plot are just artefacts of numerical accuracy.)

Example: Polynomial regression

Mean square residual (MSE) measures how well the model fits the observed data.



Can we deduce what is the best degree to use, or what was the real model that produced the data?

Will the model be useful for predicting *y* for other values of *x*?

Cross-validation

Idea: Let's split our observed data into two parts.

- Training set: We only use this part to fit the model.
- Test set: Only used for measuring the error.

Having our model (from fitting to the training set), create predictions on the test set, and measure the error there.

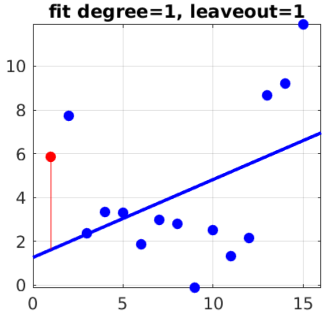
Reasoning: The model cannot possibly overfit to the data points that it does not even see!

Different possible choices on splitting. For example,

- random split to half and half
- random split to 75% training set and 25% test set
- leave-one-out: train on $n - 1$ samples, test on the remaining sample. Do this on all choices, and take average.

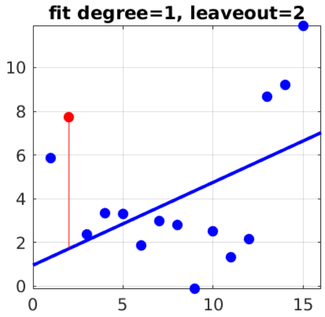
Linear regression, leave-one-out

Here we fit the model without data point #1.
Then we predict y_1 from x_1 with the model, and measure error.



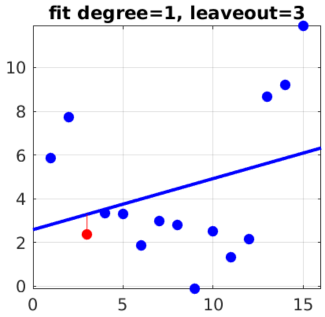
Linear regression, leave-one-out

Here we fit the model without data point #2.
Then we predict y_2 from x_2 with the model, and measure error.



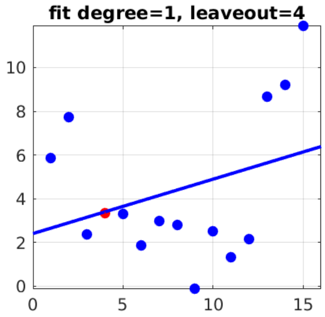
Linear regression, leave-one-out

Here we fit the model without data point #3.
Then we predict y_3 from x_3 with the model, and measure error.



Linear regression, leave-one-out

Here we fit the model without data point #4.
Then we predict y_4 from x_4 with the model, and measure error.

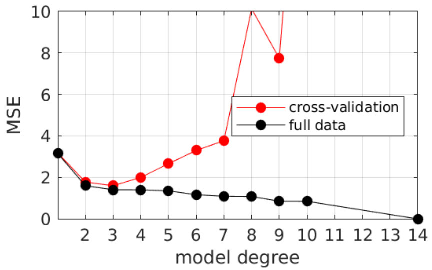


Results from the leave-one-out cross-validation

We make 15 different leave-out experiments, one for each data point left, and the *average* square residual is 3.1790.

This is a good measure of how the model can predict y from x, for values of x that it has not seen.

Do the same again with other polynomial fits.



Cross-validation in general

- The same framework can be used in many different situations.
- Generally, whatever modelling work you do, you use some part of the data for that.
- Then use another part to measure who well the model is performing.
- Can be applied to different kinds of predictions, different kinds of error measures, . . .
- E.g. also applies to classification tasks (measure the average classification error)
- Typical procedure in validating results from statistical inference / machine learning.