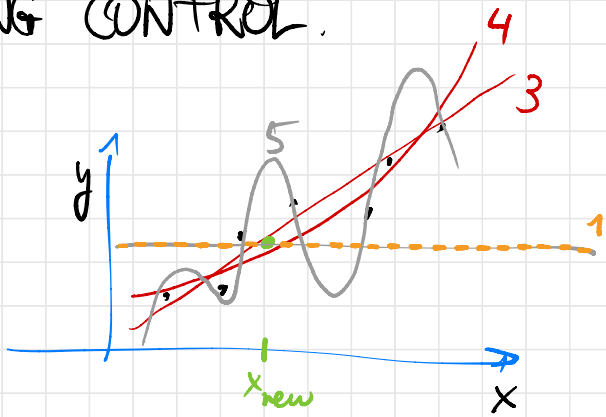
REGRESSION PROBLEMS.
OVERFITTING CONTROL.

Curve #1: clearly not fitting well. (underfitting)



Curves #3 and #4: Fit reasonably well the data

Curve #5: Fits perfectly the data since it passes through all the points!

However, clearly curve #5 doesn't seem to be the generating function of the data ⇒ it "overfits" the data

(a new input x_{new} would predict a value far from the rest of points)

↳ (expected value)

Lets work with the "linear model" of the previous lecture and find the analytical expression for $\vec{\theta}_{opt} = \vec{\theta}_{MLE}$

$$f(x; \vec{\theta}) = \vec{\theta}^T \cdot \vec{\phi}(x)$$

↳ assume this is how we parametrise the mean of the dependent variable y, and further assume that y is Gaussian, and i.i.d.

(independent & identically distributed)

$$y_i \sim N(y_i \mid f(x_i; \vec{\theta}), \sigma)$$

(& same σ for the time being)

$$\log p(\vec{y} \mid \vec{\theta}) = -\sum_{i=1}^{N} \frac{(y_i - f(x_i; \vec{\theta}))^2}{2\sigma^2} + \text{const}$$

So $\vec{\theta}_{MLE} = \underset{\vec{\theta}}{\operatorname{argmin}} \underbrace{\sum_i (y_i - f(x_i; \vec{\theta}))^2}_{C(\vec{\theta})}$

$$C(\vec{\theta}) = \sum_i (y_i - \vec{\theta}^T \cdot \vec{\phi}(x_i))^2$$

$$\frac{dC(\vec{\theta})}{d\vec{\theta}} = 0$$

$$\frac{dC(\vec{\theta})}{d\vec{\theta}} = -2 \sum_i \left(y_i - \vec{\theta}^T \cdot \vec{\phi}(x_i)\right) \cdot \vec{\phi}^T(x_i) = 0$$

$$\sum_i y_i \underbrace{\vec{\phi}^T(x_i)}_{1\times(M+1)} - \sum_i \underbrace{\vec{\theta}^T \cdot \vec{\phi}(x_i)}_{1\times 1} \underbrace{\vec{\phi}^T(x_i)}_{1\times(M+1)} = 0$$

$$\sum_i \underbrace{\vec{\phi}(x_i)}_{(M+1)\times 1} y_i - \sum_i \underbrace{\vec{\phi}(x_i)}_{(M+1)\times 1} \cdot \underbrace{\vec{\phi}^T(x_i)}_{1\times 1} \cdot \vec{\theta}$$

$$\underbrace{\Phi^T}_{(M+1)\times N} \cdot \underbrace{\vec{y}}_{N\times 1} - \underbrace{\Phi^T \Phi}_{(M+1)\times(M+1)} \cdot \vec{\theta}^{\,(M+1)\times 1} = 0$$

$$\boxed{\vec{\theta} = (\Phi^T \Phi)^{-1} \Phi^T \cdot \vec{y}} \Rightarrow \vec{\theta}_{opt} = \vec{\theta}_{MLE}$$

This solution is a generalization of the typical "linear least squares" solution

We could have decided to fit the data to a more complicated model (e.g. non-linear in $\vec{\theta}$). The MLE solution would not be analytical in general, so a numerical optimisation is required (more on that in the coming lectures)

- Let's work with a concrete example

Imagine a dataset of few points (1 dimension) whose generating function is $\sin(2\pi x)$, so

$$y = \sin(2\pi x) + \varepsilon$$

$$\hookrightarrow \varepsilon \sim \mathcal{N}(0, 0.3)$$ (mean = 0, σ = 0.3)

(Gaussian noise)

Let's consider a polynomial fit:

$$f(x, \vec{w}) = \sum_{m=0}^{M} w_m x^m$$

in this case our matrix Φ is:

$$\Phi = \begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^M \\ 1 & x_2 & x_2^2 & \cdots & x_2^M \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^M \end{pmatrix}$$

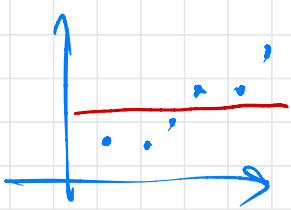
and lets compute the optimum parameters for degree $M = 2, 3, 6, 9$ polynomials

(see supplementary material)

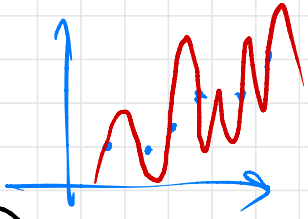
Model Selection

How many components shall we consider in our model? If it is a polynomial, which order? In general: which is the optimum model complexity?

- Roughly speaking, the optimum complexity is a trade-off between variance and bias



No variance, large bias



very little bias but large variance

$$y = f(x) + \varepsilon$$

↳ true generating function

we estimate $f(x)$ with a model $\hat{f}(x, \vec{\theta})$.

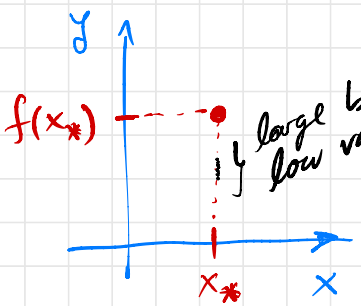
- Now, for a given (unseen before) input x_*:

$$\mathrm{Var}\left[\hat{f}(x_*)\right] = \mathbb{E}\left[\hat{f}(x_*)^2\right] - \left(\mathbb{E}\left[\hat{f}(x_*)\right]\right)^2$$

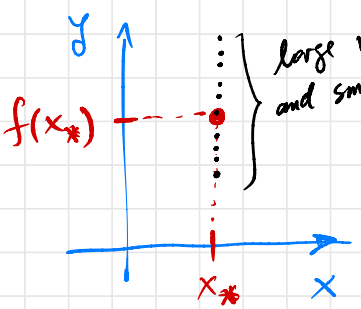
"$\mathbb{E}$" is w.r.t. different realisations of the dataset $D = \{x_i, y_i\}$ (i.e. if we had a different D, how would the results change?)

Analogously, the bias is defined as

$$\mathrm{Bias}\left[\hat{f}(x_*)\right] = \mathbb{E}\left[\hat{f}(x_*)\right] - f(x_*)$$



(different predictions come from different datasets D, each one being a noisy realisation of $f(x)$)



Actually the mean squared error

$$MSE = \mathbb{E}\left[(y_* - \hat{f}(x_*))^2\right]$$

$$y_* = f(x_*) + \varepsilon$$

$$\varepsilon \sim N(0, \sigma^2)$$

is a combination between Bias & Variance:

(This works point per point x_*, but also averaged across all points of the dataset)

The optimum model $\hat{f}(x)$ can be obtained by minimising the expected squared error:

$$\mathbb{E}[L] = \iint \{\hat{f}(x) - y\}^2 \underbrace{p(x,y)}_{\text{generative distribution of the dataset } D=\{x_i, y_i\}} dx\,dy$$

So in order to find the optimum $\hat{f}(x)$:

$$\frac{\delta \mathbb{E}[L]}{\delta \hat{f}(x)} = 0 \Rightarrow 2\iint \{\hat{f}(x) - y\}\, p(x,y)\, dx\,dy = 0$$

Solving for $\hat{f}(x)$:

$$\iint \hat{f}(x)\, p(x,y)\, dx\,dy = \iint y\, p(x,y)\, dx\,dy$$

$$\int \hat{f}(x)\, p(x)\, dx = \iint y\, p(y|x)\, p(x)\, dy\,dx$$

which, point per point, implies

$$\hat{f}(x)\, p(x) = \int y\, p(y|x)\, dy$$

$$\hat{f}_{opt}(x) = \frac{1}{p(x)} \mathbb{E}_{y|x}[y]$$

for any given x.

This solution can be used to simplify $\mathbb{E}[L]$:

$$\{\hat{f}(x) - y\}^2 = \{\hat{f}(x) - \mathbb{E}_{y|x}[y] + \mathbb{E}_{y|x}[y] - y\}^2$$

$$= \{\hat{f}(x) - \mathbb{E}_{y|x}[y]\}^2 + \{\mathbb{E}_{y|x}[y] - y\}^2$$

$$+ 2\{(\mathbb{E}_{y|x}[y] - y)(\hat{f}(x) - \mathbb{E}_{y|x}[y])\}$$

So $\int \{\hat{f}(x) - y\}^2 p(y|x)\, dy$

$$= \{\hat{f}(x) - \mathbb{E}_{y|x}[y]\}^2 + \mathrm{Var}[y|x]$$

$$+ 2\,\mathbb{E}_{y|x}[y]\,\hat{f}(x) - 2\left(\mathbb{E}_{y|x}[y]\right)^2$$

$$- 2\,\mathbb{E}_{y|x}[y]\,\hat{f}(x) + 2\left(\mathbb{E}_{y|x}[y]\right)^2$$

$\therefore$

$$\mathbb{E}[L] = \int \{\hat{f}(x) - \mathbb{E}_{y|x}[y]\}^2 p(x)\,dx + \int \mathrm{Var}[y|x]\, p(x)\,dx$$

$\int \mathrm{Var}[y|x]\, p(x)\,dx$ $\rightsquigarrow$ average noise! irreducible contribution

$= \mathbb{E}_x[\sigma^2]$

Now focusing on the 1st term

$$\{\hat{f}(x) - \mathbb{E}_{y|x}[y]\}^2 = \{\hat{f}(x) - \mathbb{E}_x[\hat{f}] + \mathbb{E}_x[\hat{f}] - \mathbb{E}_{y|x}[y]\}^2$$

$$= \{\hat{f} - \mathbb{E}_x[\hat{f}]\}^2 + \{\mathbb{E}_x[\hat{f}] - \mathbb{E}_{y|x}[y]\}^2 + 2\{\hat{f} - \mathbb{E}_x[\hat{f}]\}\{\mathbb{E}_x[\hat{f}] - \mathbb{E}_{y|x}[y]\}$$

so $\int \{\hat{f} - \mathbb{E}_x[\hat{f}]\}\{\mathbb{E}_x[\hat{f}] - \mathbb{E}_{y|x}[y]\}\, p(x)\,dx$

$= (\mathbb{E}_x[\hat{f}])^2 - \mathbb{E}_x[\hat{f}]\,\mathbb{E}_{y|x}[y]$

$$- \cancel{(\mathbb{E}_x[\hat{f}])^2} + \cancel{\mathbb{E}_x[\hat{f}]}\,\mathbb{E}_{y|x}[y] = 0$$

$$\mathbb{E}[L] = \int \{\hat{f}(x) - \mathbb{E}_{y|x}[y]\}^2 p(x)\,dx$$

$$= \int \{\hat{f} - \mathbb{E}_x[\hat{f}]\}^2 p(x)\,dx \rightarrow \mathrm{Var}[\hat{f}]$$

$$+ \int \{\mathbb{E}_x[\hat{f}] - \mathbb{E}_{y|x}[y]\}^2 p(x)\,dx$$

↳ $\mathrm{Bias}^2[\hat{f}]$

$$+ \int \mathrm{Var}[y|x]\, p(x)\,dx$$

↳ This proof is for MSE across all points in the dataset. The proof point per point is easier (see e.g. ESL book or Wikipedia)

{ Show notebook where N=20 points are used to fit polynomials of different orders P check them when P starts approaching N, overfitting occurs }

How do we avoid overfitting?

1) The "poor physicist" way:

check that $\chi^2/\text{d.o.f}$ is not $\ll 1$

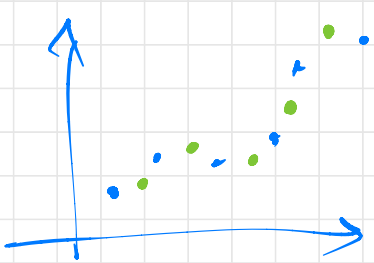
{watch out, maybe this is because of uncertainties are over-estimated}

2) If possible, increase the # of data points

Heuristic rule of thumb: $N \gtrsim (5\text{–}10) \times (\#\text{ of params.})$

(coming from old results in Computational Learning Theory)

3) Splitting data in "training" subset and "test" subset



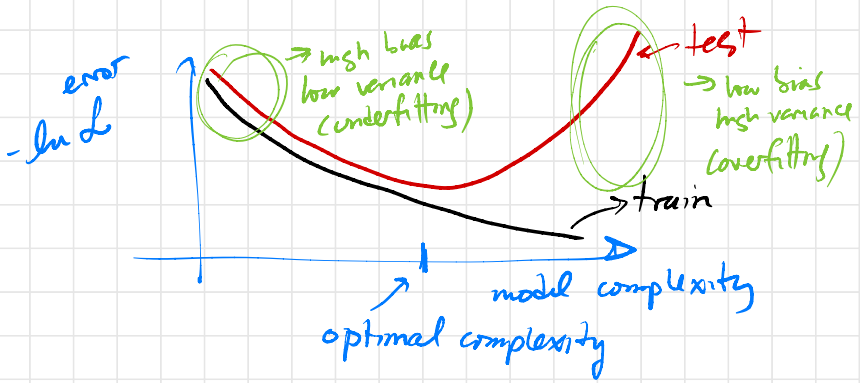
(training points) } both should cover the whole range,
(test points) }

their empirical distributions should be compatible

Motivation: fitting your function only with training points, and then check that test points

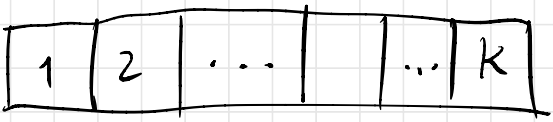
are predicted reasonably well

- Procedure:
  - Shuffle your data $\{x_i, y_i\}$ randomly, and reserve, say, 20% of it for testing.
  - Fit each model of your catalogue to the training data
  - Evaluate performance of all the models using the test data (e.g. MSE or other figure of merit)
  - The best model is then refitted to all the data, in order to get the definitive values of the optimum parameters



- Usually it is recommended to repeat the above procedure several times, for different partitions of the data: "k-fold cross-validation":

Randomly split the dataset in k partitions



Then loop over the partitions, taking the partition k^{th} as test data and the rest as training data

So the best model is going to be the one having the smallest error $\mathcal{E}^{(k)}_{test}$ on average; i.e. the one minimising $\frac{1}{K}\sum_k \mathcal{E}^{(k)}_{test}$

- Typical values of k are $k = 5, 10$

show notebook with training-test procedure

4) Regularization methods

We saw previously how the coefficients of overfitting polynomials got larger and larger. This is due to the fact that the matrix $(\Phi^T \Phi)$ is becoming singular.

In order to prevent that, Regularization methods modify the function to minimise, such that the procedure penalises large values of the coefficients:

e.g.

$$\mathcal{E}_{Ridge}(\vec{\theta}) = \frac{1}{2} \sum_{i=1}^{N} \left(y_i - f(x_i, \vec{\theta})\right)^2 + \frac{\alpha}{2} \vec{\theta}^T \cdot \vec{\theta}$$

regularization term

The minimisation is still analytical:

$$\frac{d\mathcal{E}_{Ridge}}{d\vec{\theta}} = -\Phi^T \cdot \vec{y} + \Phi^T \Phi \cdot \vec{\theta} + \alpha \vec{\theta} = 0$$

$$\vec{\theta}_{ridge} = \left(\Phi^T \Phi + \alpha \cdot \tilde{\mathbb{1}} \right)^{-1} \Phi^T \cdot \vec{y}$$

so when $\Phi^T \Phi$ becomes more and more singular, the inverse doesn't explode

$$\tilde{\mathbb{1}} = \begin{pmatrix} 0 & \vec{0}^T \\ \vec{0} & \mathbb{1} \end{pmatrix}$$

{ show notebook with the Ridge regression }