Statistical Machine Learning

Christian Walder

Machine Learning Research Group
CSIRO Data61

and

College of Engineering and Computer Science
The Australian National University

Canberra
Semester One, 2020.

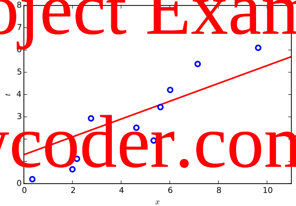
(Many figures from C. M. Bishop, "Pattern Recognition and Machine Learning")

Outlines

Part III

Linear Regression 1

$$N = 10$$
$$\mathbf{x} \equiv (x_1, \ldots, x_N)^T$$
$$\mathbf{t} \equiv (t_1, \ldots, t_N)^T$$
$$x_i \in \mathbb{R} \quad i = 1, \ldots, N$$
$$t_i \in \mathbb{R} \quad i = 1, \ldots, N$$



- Predictor $y(x, \mathbf{w})$?
- Performance measure?
- Optimal solution w^*?
- Recall: projection, inverse

- Gaussian Distribution
- Bayes Rule
- Expected Loss

Linear Curve Fitting - Least Squares

$$N = 10$$

$$\mathbf{x} \equiv (x_1, \dots, x_N)^T$$

$$\mathbf{t} \equiv (t_1, \dots, t_N)^T$$

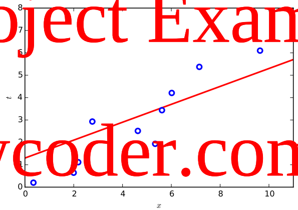
$$x_i \in \mathbb{R} \quad i = 1, \dots, N$$

$$t_i \in \mathbb{R} \quad i = 1, \dots, N$$

$$y(x, \mathbf{w}) = w_1 x + w_0$$

$$X \equiv [\mathbf{x} \quad 1]$$

$$w^* \equiv (X^T X)^{-1} X^T \mathbf{t}$$



We assume

$$t = \underbrace{y(\mathbf{x}, \mathbf{w})}_{\text{deterministic}} + \underbrace{\epsilon}_{\text{Gaussian noise}}$$

Curve fitting - revisited

- *a priori* belief about the parameter $\mathbf{w}$ captured in the prior probability $p(\mathbf{w})$
- observed data $\mathcal{D} = \{t_1, \dots, t_N\}$
- calculate the belief in $\mathbf{w}$ after the data $\mathcal{D}$ have been observed

$$p(\mathbf{w} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$$

- $p(\mathcal{D} \mid \mathbf{w})$ as a function of $\mathbf{w}$: likelihood function
- likelihood expresses how probable the data are for different values of $\mathbf{w}$ — it is not a probability density with respect $\mathbf{w}$ (but it is with respect to $\mathcal{D}$; prove it)

Maximum Likelihood

- Consider the linear regression problem, where we have random variables $\mathbf{x}_n$ and t_n.
- We assume a conditional model $\quad t_n | \mathbf{x}_n$
- We propose a distribution, parameterized by θ

$$t_n | \mathbf{x}_n \sim \text{density}(\theta)$$

For a given θ the density defines the probability of observing $t_n | \mathbf{x}_n$.

- We are interested in finding θ that maximises the probability (called the likelihood) of the data.

Likelihood function $p(\mathcal{D} \mid \mathbf{w})$

Frequentist Approach

- $\mathbf{w}$ considered fixed parameter
- value defined by some 'estimator'
- error bars on the estimated $\mathbf{w}$ obtained from the distribution of possible data sets $\mathcal{D}$

Bayesian Approach

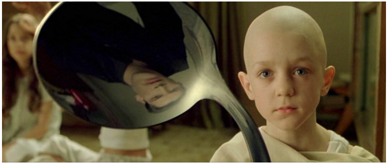
- only one single data set $\mathcal{D}$
- uncertainty in the parameters comes from a probability distribution over $\mathbf{w}$

Frequentist Estimator - Maximum Likelihood

- choose $\mathbf{w}$ for which the likelihood $p(\mathcal{D} \mid \mathbf{w})$ (the probability of the observed data) is maximal
- the most common heuristic for learning a single fixed $\mathbf{w}$
- equivalently: error function is negative log of likelihood function, to be minimised
- log is a monotonic function
- maximising the likelihood $\iff$ minimising the error
- Example: Fair-looking coin is tossed three times, always landing on heads.
- Maximum likelihood estimate of the probability of landing heads will give 1.

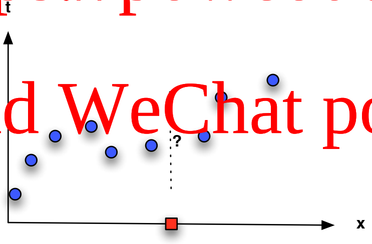
Bayesian Approach

- including prior knowledge easy (via prior $\mathbf{w}$)
- subjective choice of prior, allows better results by incorporating domain knowledge
- sometimes choice of prior motivated by convinient mathematical form
- prior irrelevant as $N \to \infty$, but helps for small N
- need to sum/integrate over the whole parameter space
  - advances in sampling (Markov Chain Monte Carlo methods)
  - advances in approximation schemes (Variational Bayes, Expectation Propagation)
- there is no true $\mathbf{w}$.

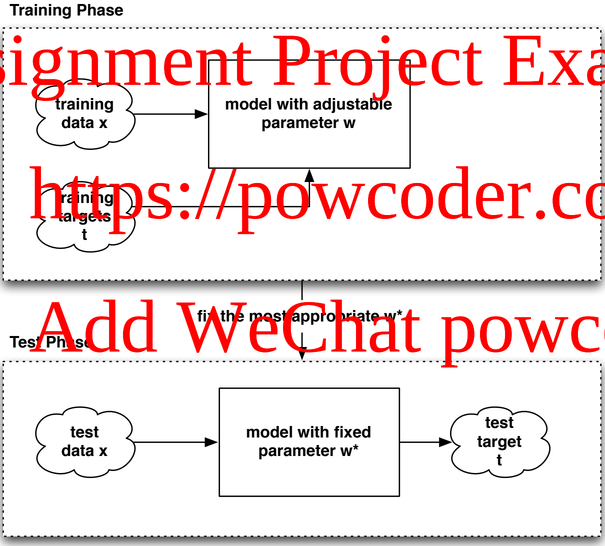


Regression

- Given a training data set of N observations $\{\mathbf{x}_n\}$ and target values t_n.
- Goal : Learn to predict the value of one ore more target values t given a new value of the input $\mathbf{x}$.
- Example: Polynomial curve fitting (see Introduction).

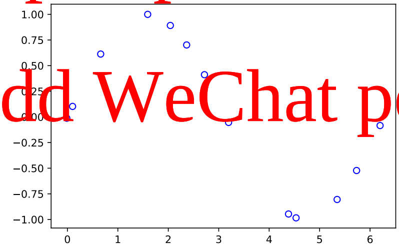


Supervised Learning: (non-Bayesian) Point Estimate



Why Linear Regression?

- Analytic solution when minimising sum of squared errors
- Well understood statistical behaviour
- Efficient algorithms exist for convex losses and regularizers
- But what if the relationship is non-linear?



Linear Basis Function Models

- Linear combination of fixed nonlinear basis functions

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$$

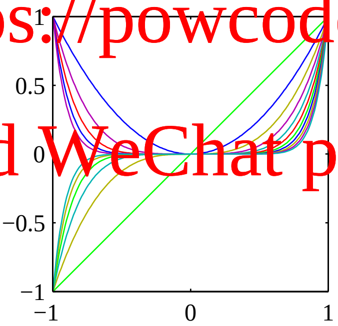
- parameter $\mathbf{w} = (w_0, \dots, w_{M-1})^T$
- basis functions $\boldsymbol{\phi}(\mathbf{x}) = (\phi_0(\mathbf{x}), \dots, \phi_{M-1}(\mathbf{x}))^T$
- convention $\phi_0(\mathbf{x}) = 1$
- w_0 is the bias parameter

Polynomial Basis Functions

- Scalar input variable x

$$\phi_j(x) = x^j$$

- Limitation : Polynomials are global functions of the input variable x so the learned function will extrapolate poorly

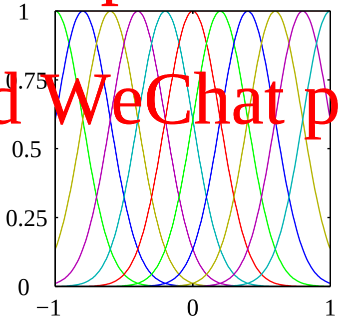


'Gaussian' Basis Functions

- Scalar input variable x

$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$

- Not a probability distribution.
- No normalisation required, taken care of by the model parameters w.
- Well behaved away from the data (though pulled to zero).



Sigmoidal Basis Functions

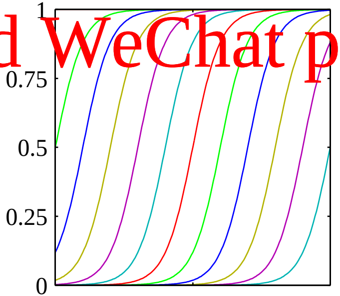
- Scalar input variable x

$$\phi_j(x) = \sigma\left(\frac{x - \mu_j}{s}\right)$$

where $\sigma(a)$ is the logistic sigmoid function defined by

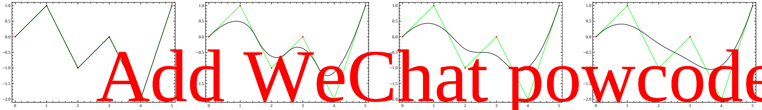
$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

- $\sigma(a)$ is related to the hyperbolic tangent $\tanh(a)$ by $\tanh(a) = 2\sigma(a) - 1$.



Other Basis Functions

- Fourier Basis : each basis function represents a specific frequency and has infinite spatial extent.
- Wavelets : localised in both space and frequency (also mutually orthogonal to simplify appliciation).
- Splines (piecewise polynomials restricted to regions of the input space; additional constraints where pieces meet, e.g. smoothness constrains $\rightarrow$ conditions on the derivatives).



Linear Splines | Quadratic Splines | Cubic Splines | Quartic Splines

Approximate the points $\{(0,0),(1,1),(2,-1),(3,0),(4,-2),(5,1)\}$ by different splines.

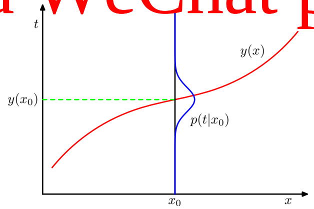
Maximum Likelihood and Least Squares

- No special assumption about the basis functions $\phi_j(\mathbf{x})$. In the simplest case, one can think of $\phi_j(\mathbf{x}) = x_j$, or $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$.
- Assume target t is given by

$$t = \underbrace{y(\mathbf{x}, \mathbf{w})}_{\text{deterministic}} + \underbrace{\epsilon}_{\text{noise}}$$

  where ϵ is a zero-mean Gaussian random variable with precision (inverse variance) β.
- Thus

$$p(t \mid \mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t \mid y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$



Maximum Likelihood and Least Squares

- Likelihood of one target t given the data $\mathbf{x}$ was

$$p(t \,|\, \mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t \,|\, y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

- Now, a set of inputs $\mathbf{X}$ with corresponding target values $\mathbf{t}$.
- Assume data are independent and identically distributed (i.i.d.) (means : data are drawn independent and from the same distribution). The likelihood of the target $\mathbf{t}$ is then

$$\begin{aligned} p(\mathbf{t} \,|\, \mathbf{X}, \mathbf{w}, \beta) &= \prod_{n=1}^{N} \mathcal{N}(t_n \,|\, y(\mathbf{x}_n, \mathbf{w}), \beta^{-1}) \\ &= \prod_{n=1}^{N} \mathcal{N}(t_n \,|\, \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) \end{aligned}$$

- From now on drop the conditioning variable $\mathbf{X}$ from the notation, as with supervised learning we do not seek to model the distribution of the input data.

Maximum Likelihood and Least Squares

- Consider the logarithm of the likelihood $p(\mathbf{t} \mid \mathbf{w}, \beta)$ (the logarithm is a monotone function!)

$$
\begin{aligned}
\ln p(\mathbf{t} \mid \mathbf{w}, \beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(t_n \mid \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) \\
&= \sum_{n=1}^{N} \ln \left(\sqrt{\frac{\beta}{2\pi}} \exp\left\{ -\frac{\beta}{2} (t_n - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n))^2 \right\} \right) \\
&= \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E_D(\mathbf{w})
\end{aligned}
$$

where the sum-of-squares error function is

$$
E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{t_n - \mathbf{w}^T \boldsymbol{\phi}(x_n)\}^2 .
$$

- $\arg\max_{\mathbf{w}} \ln p(\mathbf{t} \mid \mathbf{w}, \beta) \to \arg\min_{\mathbf{w}} E_D(\mathbf{w})$

- Goal: Find a more compact representation.
- Rewrite the error function

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\phi(x_n)\}^2 = \frac{1}{2}(\mathbf{t} - \mathbf{\Phi w})^T(\mathbf{t} - \mathbf{\Phi w})$$

where $\mathbf{t} = (t_1, \dots, t_N)^T$, and

$$\mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \dots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \dots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \dots & \phi_{M-1}(\mathbf{x}_N) \end{bmatrix}$$

- The log likelihood is now

$$\ln p(\mathbf{t} \mid \mathbf{w}, \beta) = \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E_D(\mathbf{w})$$
$$= \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta \frac{1}{2} (\mathbf{t} - \mathbf{\Phi w})^T (\mathbf{t} - \mathbf{\Phi w})$$

- Find critical points of $\ln p(\mathbf{t} \mid \mathbf{w}, \beta)$.
- The gradient with respect to $\mathbf{w}$ is

$$\nabla_{\mathbf{w}} \ln p(\mathbf{t} \mid \mathbf{w}, \beta) = \beta \mathbf{\Phi}^T (\mathbf{t} - \mathbf{\Phi w}).$$

Setting the gradient to zero gives

$$0 = \mathbf{\Phi}^T \mathbf{t} - \mathbf{\Phi}^T \mathbf{\Phi w},$$

- which results in

$$\mathbf{w}_{ML} = (\mathbf{\Phi}^T \mathbf{\Phi})^{-1} \mathbf{\Phi}^T \mathbf{t} = \mathbf{\Phi}^\dagger \mathbf{t}$$

where $\mathbf{\Phi}^\dagger$ is the Moore-Penrose pseudo-inverse of the matrix $\mathbf{\Phi}$.

Maximum Likelihood and Least Squares

- The log likelihood with the optimal $\mathbf{w}_{ML}$ is now

$$\ln p(\mathbf{t} \mid \mathbf{w}_{ML}, \beta)$$
$$= \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta \frac{1}{2} (\mathbf{t} - \mathbf{\Phi} \mathbf{w}_{ML})^T (\mathbf{t} - \mathbf{\Phi} \mathbf{w}_{ML})$$

- Find critical points of $\ln p(\mathbf{t} \mid \mathbf{w}, \beta)$ wrt β,

$$\frac{\partial \ln p(\mathbf{t} \mid \mathbf{w}_{ML}, \beta)}{\partial \beta} = 0$$

results in

$$\frac{1}{\beta_{ML}} = \frac{1}{N} (\mathbf{t} - \mathbf{\Phi} \mathbf{w}_{ML})^T (\mathbf{t} - \mathbf{\Phi} \mathbf{w}_{ML})$$

- Note: We can first find the maximum likelihood for $\mathbf{w}$ as this does not depend on β. Then we can use $\mathbf{w}_{ML}$ to find the maximum likelihood solution for β.
- Could we have chosen optimisation wrt β first, and then wrt to $\mathbf{w}$?

Sequential Learning - Stochastic Gradient Descent

- For large data sets, calculating the maximum likelihood parameters $\mathbf{w}_{ML}$ and β_{ML} may be costly.
- For online applications, never all data in memory.
- Use a sequential algorithms (online algorithm).
- If the error function is a sum over data points $E = \sum_n E_n$, then
  1. initialise $\mathbf{w}^{(0)}$ to some starting value
  2. update the parameter vector at iteration $\tau + 1$ by

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n,$$

where E_n is the error function after presenting the nth data set, and η is the learning rate.

Sequential Learning - Stochastic Gradient Descent

- For the sum-of-squares error function, stochastic gradient descent results in

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \eta \left(t_n - \mathbf{w}^{(\tau)T} \phi(\mathbf{x}_n) \right) \phi(\mathbf{x}_n)$$

- The value for the learning rate must be chosen carefully. A too large learning rate may prevent the algorithm from converging. A too small learning rate does follow the data too slowly.

Regularized Least Squares

- Add regularisation in order to prevent overfitting

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

with regularisation coefficient λ.

- Simple quadratic regulariser

$$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

- Maximum likelihood solution

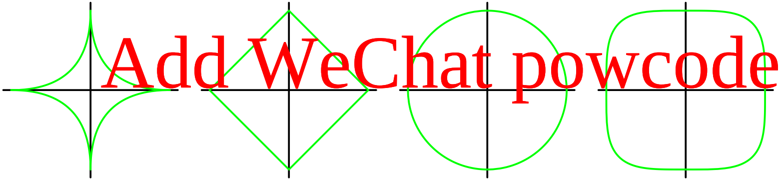
$$\mathbf{w} = \left(\lambda\mathbf{I} + \boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T\mathbf{t}$$

Regularized Least Squares

- More general regulariser

$$E_W(\mathbf{w}) = \frac{1}{2} \sum_{j=1}^{M} |w_j|^q$$

- $q = 1$ (lasso) leads to a sparse model if λ large enough.



Lagrangian Dual View of the Regulariser

- By the Lagrange multiplier method, minimization of the regularized error function

$$\frac{1}{2}\sum_{n=1}^{N}(t_n - \mathbf{w}^\top \phi(\mathbf{x}_n))^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q \,,$$

- is equivalent to minimizing the unregularized sum-of-squares error,

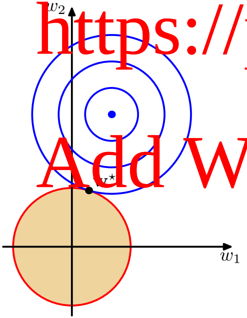
$$\frac{1}{2}\sum_{n=1}^{N}(t_n - \mathbf{w}^\top \phi(\mathbf{x}_n))^2 \quad \text{subject to} \quad \sum_{j=1}^{M}|w_j|^q \leq \eta.$$

- This yields the figures on the next slide.

Comparison of Quadratic and Lasso Regulariser

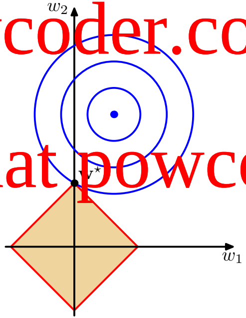
Quadratic regulariser

$$\frac{1}{2}\sum_{j=1}^{M} w_j^2$$



Lasso regulariser

$$\frac{1}{2}\sum_{j=1}^{M} |w_j|$$



Multiple Outputs

- More than 1 target variable per data point.
- $\mathbf{y}$ becomes a vector instead of a scalar. Each dimension can be treated with a different set of basis functions (and that may be necessary if the data in the different target dimensions represent very different types of information.)
- Here we restrict ourselves to the SAME basis functions

$$\mathbf{y}(\mathbf{x}, \mathbf{w}) = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x})$$

  where $\mathbf{y}$ is a K-dimensional column vector, $\mathbf{W}$ is an $M \times K$ matrix of model parameters, and $\boldsymbol{\phi}(\mathbf{x}) = \left(\phi_0(\mathbf{x}), \dots, \phi_{M-1}(\mathbf{x})\right)$, with $\phi_0(\mathbf{x}) = 1$, as before.

- Define target matrix $\mathbf{T}$ containing the target vector $\mathbf{t}_n^T$ in the n^{th} row.

Multiple Outputs

- Suppose the conditional distribution of the target vector is an isotropic Gaussian of the form

$$p(\mathbf{t}\,|\,\mathbf{x}, \mathbf{W}, \beta) = \mathcal{N}(\mathbf{t}\,|\,\mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}), \beta^{-1}\mathbf{I}).$$

- The log likelihood is then

$$\begin{aligned}\ln p(\mathbf{T}\,|\,\mathbf{X}, \mathbf{W}, \beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{t}_n\,|\,\mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}\mathbf{I}) \\ &= \frac{NK}{2} \ln\left(\frac{\beta}{2\pi}\right) - \frac{\beta}{2} \sum_{n=1}^{N} \|\mathbf{t}_n - \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}_n)\|^2\end{aligned}$$

Multiple Outputs

- Maximisation with respect to $\mathbf{W}$ results in

$$\mathbf{W}_{ML} = (\mathbf{\Phi}^T \mathbf{\Phi})^{-1} \mathbf{\Phi}^T \mathbf{T}.$$

- For each target variable $\mathbf{t}_k$, we get

$$\mathbf{w}_k = (\mathbf{\Phi}^T \mathbf{\Phi})^{-1} \mathbf{\Phi}^T \mathbf{t}_k = \mathbf{\Phi}^{\dagger} \mathbf{t}_k.$$

- The solution between the different target variables decouples.
- Holds also for a general Gaussian noise distribution with arbitrary covariance matrix.

- Why? $\mathbf{W}$ defines the mean of the Gaussian noise distribution. And the maximum likelihood solution for the mean of a multivariate Gaussian is independent of the covariance.

- Over-fitting results from a large number of basis functions and a relatively small training set.
- Regularisation can prevent overfitting, but how to find the correct value for the regularisation constant λ ?
- Frequentists viewpoint of the model complexity is the bias-variance trade-off.

Loss Function for Regression

- Choose an estimator $y(\mathbf{x})$ to estimate the target value t for each input $\mathbf{x}$.
- Choose a loss function $L(t, y(\mathbf{x}))$ which measures the difference between the target t and the estimate $y(\mathbf{x})$.
- The expected loss is then

$$\mathbb{E}[L] = \iint L(t, y(\mathbf{x}))\, p(\mathbf{x}, t)\, \mathrm{d}\mathbf{x}\, \mathrm{d}t$$

- Common choice: Squared Loss

$$L(t, y(\mathbf{x})) = \{y(\mathbf{x}) - t\}^2$$

- Expected loss for squared loss function

$$\mathbb{E}[L] = \iint \{y(\mathbf{x}) - t\}^2\, p(\mathbf{x}, t)\, \mathrm{d}\mathbf{x}\, \mathrm{d}t.$$

Loss Function for Regression

- Expected loss for squared loss function

$$\mathbb{E}[L] = \iint \{y(\mathbf{x}) - t\}^2 \, p(\mathbf{x}, t) \, \mathrm{d}\mathbf{x} \, \mathrm{d}t.$$

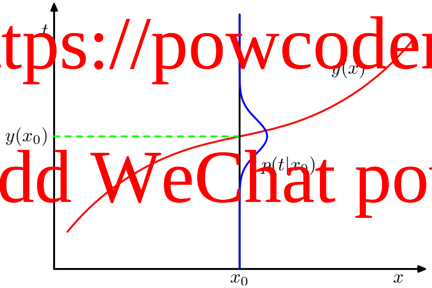
- Minimise $\mathbb{E}[L]$ by choosing the regression function

$$y(\mathbf{x}) = \frac{\int t \, p(\mathbf{x}, t) \, \mathrm{d}t}{p(\mathbf{x})} = \int t \, p(t \,|\, \mathbf{x}) \, \mathrm{d}t = \mathbb{E}_t[t \,|\, \mathbf{x}]$$

(calculus of variations is not required to derive this result ; we may work point-wise by fixing an $\mathbf{x}$ and using stationarity to solve for $y(\mathbf{x})$ — why is that sufficient?).

Optimal Predictor for Squared Loss

- The regression function which minimises the expected squared loss, is given by the mean of the conditional distribution $p(t \mid \mathbf{x})$.



Analysing the Squared Loss (1)

- Analyse the expected loss

$$\mathbb{E}[L] = \iint \{y(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) \, \mathrm{d}\mathbf{x} \, \mathrm{d}t.$$

- Rewrite the squared loss

$$\begin{aligned}
\{y(\mathbf{x}) - t\}^2 &= \{y(\mathbf{x}) - \mathbb{E}[t \mid \mathbf{x}] + \mathbb{E}[t \mid \mathbf{x}] - t\}^2 \\
&= \{y(\mathbf{x}) - \mathbb{E}[t \mid \mathbf{x}]\}^2 + \{\mathbb{E}[t \mid \mathbf{x}] - t\}^2 \\
&\quad + 2\{y(\mathbf{x}) - \mathbb{E}[t \mid \mathbf{x}]\}\{\mathbb{E}[t \mid \mathbf{x}] - t\}
\end{aligned}$$

- Claim

$$\iint \{y(\mathbf{x}) - \mathbb{E}[t \mid \mathbf{x}]\}\{\mathbb{E}[t \mid \mathbf{x}] - t\} \, p(\mathbf{x}, t) \, \mathrm{d}\mathbf{x} \, \mathrm{d}t = 0.$$

- Claim

$$\iint \{y(\mathbf{x}) - \mathbb{E}[t \,|\, \mathbf{x}]\}\{\mathbb{E}[t \,|\, \mathbf{x}] - t\} p(\mathbf{x}, t)\, \mathrm{d}\mathbf{x}\, \mathrm{d}t = 0$$

- Seperate functions depending on t from function depending on $\mathbf{x}$

$$\int \{y(\mathbf{x}) - \mathbb{E}[t \,|\, \mathbf{x}]\} \left(\int \{\mathbb{E}[t \,|\, \mathbf{x}] - t\} p(\mathbf{x}, t)\, \mathrm{d}t \right) \mathrm{d}\mathbf{x}$$

- Calculate the integral over t

$$\begin{aligned} \int \{\mathbb{E}[t \,|\, \mathbf{x}] - t\} p(\mathbf{x}, t)\, \mathrm{d}t &= \mathbb{E}[t \,|\, \mathbf{x}]\, p(\mathbf{x}) - p(\mathbf{x}) \int \frac{t\, p(\mathbf{x}, t)}{p(\mathbf{x})}\, \mathrm{d}t \\ &= \mathbb{E}[t \,|\, \mathbf{x}]\, p(\mathbf{x}) - p(\mathbf{x}) \mathbb{E}[t \,|\, \mathbf{x}] \\ &= 0 \end{aligned}$$

- The expected loss is now

$$\mathbb{E}[L] = \int \{y(\mathbf{x}) - \mathbb{E}[t \mid \mathbf{x}]\}^2 p(\mathbf{x}) \, \mathrm{d}\mathbf{x} + \int \operatorname{var}[t \mid \mathbf{x}] \, p(\mathbf{x}) \, \mathrm{d}\mathbf{x} \quad (1)$$

- Minimise first term by choosing $y(\mathbf{x}) = \mathbb{E}[t \mid \mathbf{x}]$ (as we saw already).
- Second term represents the intrinsic variability of the target data (can be regarded as noise). Independent of the choice $y(\mathbf{x})$, can not be reduced by learning a better $y(\mathbf{x})$.

The Bias-Variance Decomposition (1)

- Consider again squared loss for which the optimal prediction is given by the conditional expectation $h(x)$

$$h(\mathbf{x}) = \mathbb{E}\left[t \mid \mathbf{x}\right] = \int t\, p(t \mid \mathbf{x})\, \mathrm{d}t.$$

- Since $h(x)$ is unavailable to us, it must be estimated from a (finite) dataset $\mathcal{D}$.
- $\mathcal{D}$ is a finite sample from the unknown joint $p(\mathbf{x}, t)$
- Notate the dependency of the learned function on the data by $y(\mathbf{x}; \mathcal{D})$.
- Evaluate performance of algorithm by taking the expectation $\mathbb{E}_{\mathcal{D}}\left[L\right]$ over all data sets $\mathcal{D}$

- Taking the expectation over data sets $\mathcal{D}$, using Eqn 1, and interchanging the order of expectations for the first term:

$$\mathbb{E}_{\mathcal{D}}\left[\mathbb{E}\left[L\right]\right] = \int \mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - h(\mathbf{x})\}^2\right] p(\mathbf{x})\,\mathrm{d}\mathbf{x}$$

$$+ \iint \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t$$

- Again, add and subtract the expectation $\mathbb{E}_{\mathcal{D}}\left[y(\mathbf{x};\mathcal{D})\right]$

$$\{y(\mathbf{x};\mathcal{D}) - h(\mathbf{x})\}^2 = \{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}\left[y(\mathbf{x};\mathcal{D})\right] + \mathbb{E}_{\mathcal{D}}\left[y(\mathbf{x};\mathcal{D})\right] - h(\mathbf{x})\}^2$$

and show that the mixed term vanishes under the expectation $\mathbb{E}_{\mathcal{D}}$.

The Bias-Variance Decomposition (3)

- Expected loss $\mathbb{E}_{\mathcal{D}}[L]$ over all data sets $\mathcal{D}$

$$\text{expected loss} = (\text{bias})^2 + \text{variance} + \text{noise}.$$

where

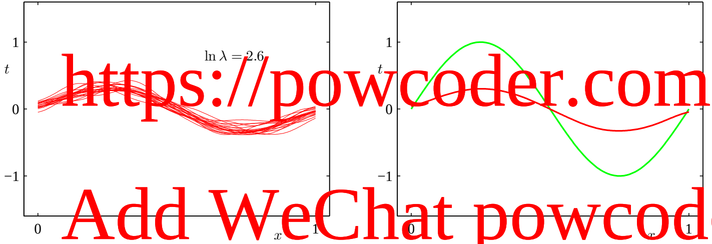
$$(\text{bias})^2 = \int \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 \, p(\mathbf{x}) \, \mathrm{d}\mathbf{x}$$

$$\text{variance} = \int \mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right] p(\mathbf{x}) \, \mathrm{d}\mathbf{x}$$

$$\text{noise} = \iint \{h(\mathbf{x}) - t\}^2 \, p(\mathbf{x}, t) \, \mathrm{d}\mathbf{x} \, \mathrm{d}t.$$

- $(\text{bias})^2$: How accurate is a model across different training sets? (How much does the average prediction over all data sets differ from the desired regression function ?)
- variance : How sensitive is the model to small changes in the training set? (How much do solutions for individual data sets vary around their average ?

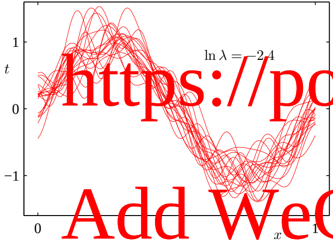
- Simple models have low variance and high bias.

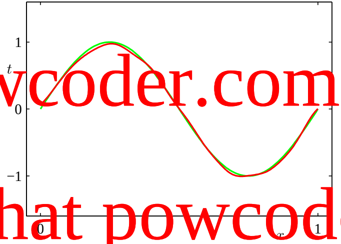


Left: Result of fitting the model to 100 data sets, only 25 shown.
Right: Average of the 100 fits in red, the sinusoidal function from where the data were created in green.

The Bias-Variance Decomposition

- Complex models have high variance and low bias

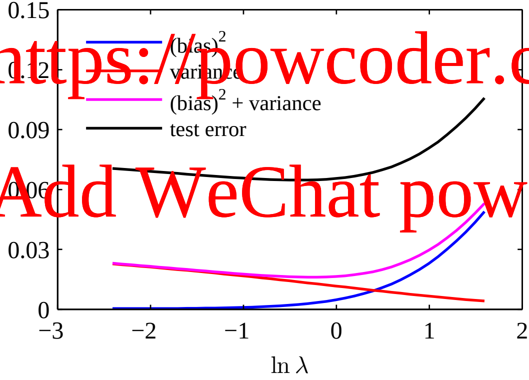




Left: Result of fitting the model to 100 data sets, only 25 shown.
Right: Average of the 100 fits in red, the sinusoidal function from where the data were created in green.

The Bias-Variance Decomposition

- Dependence of bias and variance on the model complexity
- Squared bias, variance, their sum, and test data
- The minimum for $(\text{bias})^2 + \text{variance}$ occurs close to the value that gives the minimum error



Unbiased Estimators

- You may have encountered *unbiased estimators*
- Why guarantee zero bias? To quote the pioneer of Bayesian inference, Edwin Jaynes, from his book *Probability Theory: The Logic of Science (2003)*:

> **Why do they do this?** Why do orthodoxians put such exaggerated emphasis on bias? We suspect that the main reason is simply that they are caught in a psycho–semantic trap of their own making. When we call the quantity $(\langle\beta\rangle - \alpha)$ the "bias", that makes it sound like something awfully reprehensible, which we must get rid of at all costs. If it had been called instead the "component of error orthogonal to the variance" as suggested by the Pythagorean form of (17–2), it would have been clear to all that these two contributions to the error are on an equal footing; it is folly to decrease one at the expense of increasing the other. This is just the price one pays for choosing a technical terminology that carries an emotional load, implying value judgments; orthodoxy falls constantly into this tactical error.

The Bias-Variance Decomposition

- Tradeoff between bias and variance
  - simple models have low variance and high bias
  - complex models have high variance and low bias
- The sum of bias and variance has a minimum at a certain model complexity.
- Expected loss $\mathbb{E}_{\mathcal{D}}[L]$ over all data sets $\mathcal{D}$

$$\text{expected loss} = (\text{bias})^2 + \text{variance} + \text{noise}.$$

- The noise comes from the data, and can not be removed from the expected loss.
- To analyse the bias-variance decomposition : many data sets needed, which are not always available.