University of BRISTOL

Computer Science Year 2

Algorithms & Data

Estimation, Regression, Classification

Prof Alin Achim

University of BRISTOL

Last time …

- **Classification**
- Bayesian classifiers are optimal with respect to minimising *classification error probability*.
- The classification error probability is not always the best criterion to be adopted for minimization as it assigns the same importance to all errors, while there are cases in which some wrong decisions may have more serious implications than others.
- It is sometimes more appropriate to assign a penalty term to weigh each error and hence minimise the average risk → *likelihood ratio test*.
- Yet another alternative is the *Neyman – Pearson criterion*: the error for one class is constrained to be fixed and equal to a chosen value.
  - Example of such classifier is the CFAR processor ubiquitously used in radar ATR/D.

Objectives

- LSE – Regression – Prediction
- Curve Fitting
- Best Constant Predictor
- Best Slope Predictor
- Straight Line Fitting
- Multivariate Regression
- Regularisation
- Nonlinear regression

Least Squares – The generic approach

- Consider the multiple observations:

$$x[n] = A + w[n], \text{ where } n = 0, 1, \ldots, N-1 \text{ and } w[n] \sim N(0, \sigma^2)$$

- Or more generally:

$x[n] = f(\theta_1, \theta_2, \ldots, \theta_M) + w[n]$, i.e. *f* is a function of M parameters

- The least square estimator (LSE) minimizes:

$$J = \sum_{n=0}^{N-1} \left(x[n] - f(\theta_1, \theta_2, \ldots, \theta_M) \right)^2$$

Least Squares - Properties

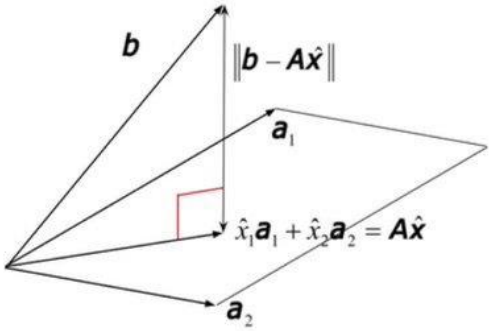
- LSE is widely used when estimating parameters for linear models
- No assumptions about the data are made
- If $w[n] \sim N(0, \sigma^2)$, LSE coincides with the MLE!
- Geometric interpretation: the LS estimate is an orthogonal projection of the data vector onto the space defined by the independent variable.

- Inverse problems formulation:
  - $\boldsymbol{b} = \boldsymbol{Ax} + \boldsymbol{n}, \boldsymbol{n} \sim N(0, \sigma^2)$
  - $\hat{\boldsymbol{x}} = min_{\boldsymbol{x}} \|\boldsymbol{b} - \boldsymbol{Ax}\|^2$

Geometric interpretation

- $\boldsymbol{A\hat{x}}$ is the orthogonal projection of $\boldsymbol{b}$ onto *range(A)*

$$\Leftrightarrow \boldsymbol{A}^T(\boldsymbol{b} - \boldsymbol{A\hat{x}}) = \boldsymbol{0} \Leftrightarrow \boldsymbol{A}^T\boldsymbol{A\hat{x}} = \boldsymbol{A}^T\boldsymbol{b}$$



Least Squares – A simple example of use

- Suppose we have a random sample $\{x_n\}, n = 0, \ldots, N-1$, drawn from a population with mean μ_x and standard deviation σ_x.
  - We can express x_n using a linear model:

$$x_n = \mu_x + \varepsilon_n, E[\varepsilon_n] = 0 \; and \; E[\varepsilon_n^2] = \sigma_x^2$$

  - Estimate μ_x via LSE, that is minimise:

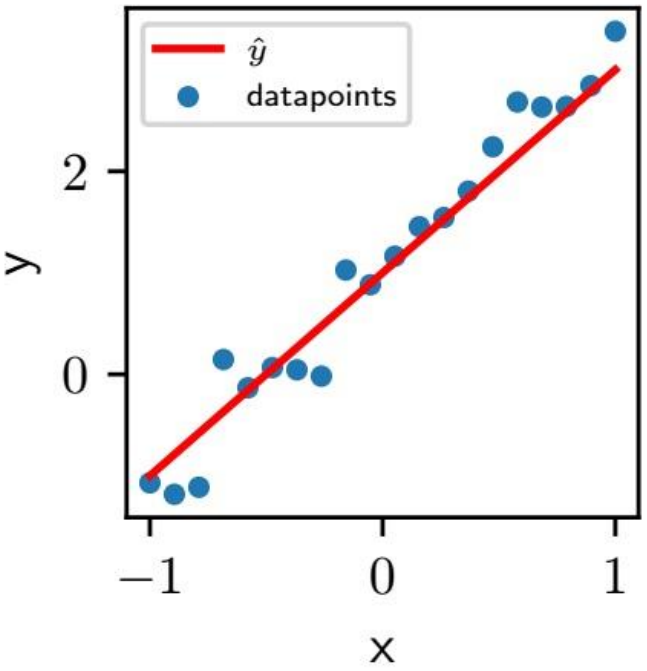
$$J(\mu_x) = \sum_{n=0}^{N-1} (x_n - \hat{\mu}_x)^2$$

$$\frac{\partial J(\mu_x)}{\partial \mu_x} = -2 \sum_{n=0}^{N-1} (x_n - \hat{\mu}_x) = 0$$

$$\Rightarrow \sum_{n=0}^{N-1} x_n = N\hat{\mu}_x \Rightarrow \hat{\mu}_x = \frac{1}{N} \sum_{n=0}^{N-1} x_n$$

Least Squares – Regression – Curve fitting

- Regression is essentially the word used to refer to curve fitting.
- Simplest form of regression is fitting a straight line to a dataset involving inputs, x_i, and targets (predictions), y_i.



- Red line is $\hat{y}(x) = 2x + 1$ and $\hat{y}(x_i)$ represents the prediction of y_i.
- This could have been guessed by looking at the scatterplot or estimated by regression.
- The sum of squared distances between the predictions and the true values is used as loss function:

$$\mathcal{L} = \sum_i (\hat{y}(x_i) - y_i)^2$$

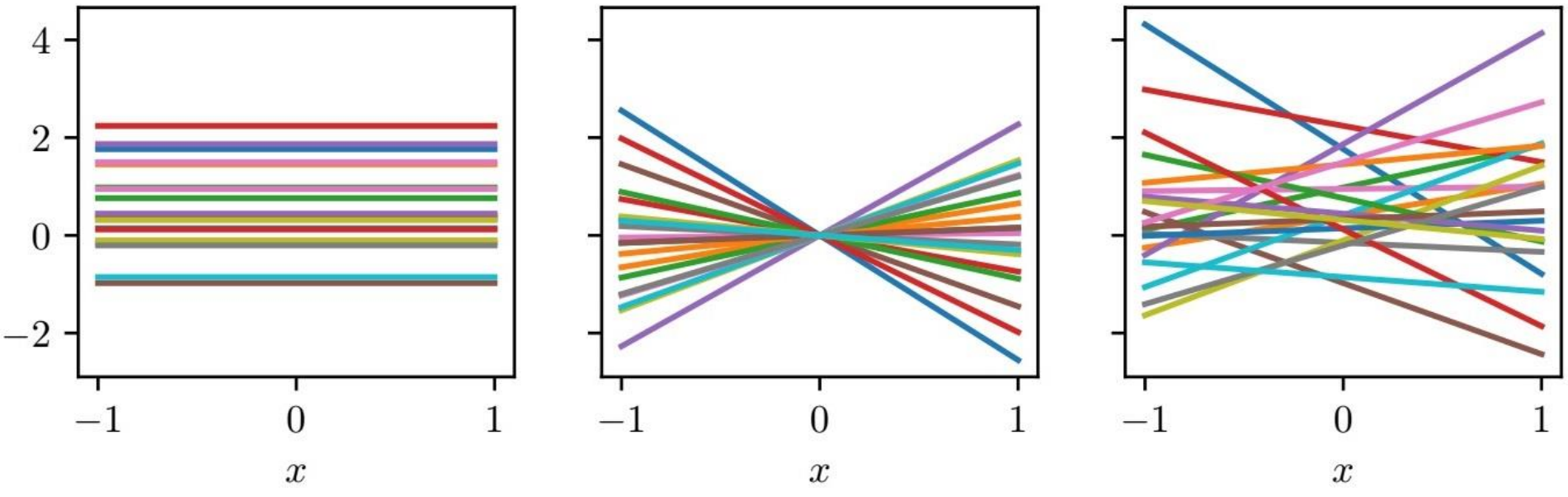
Prediction functions

- Loss function adopted is the sum of squared differences:

$$\mathcal{L} = \sum_i (\hat{y}(x_i) - y_i)^2$$

- One needs to choose good prediction functions $\hat{y}(x)$.
- Three parametric predictors considered: the constant predictor, the slope predictor and the straight-line predictor, i.e.

$\hat{y}_{const}(x) = w_1$; $\hat{y}_{slope}(x) = w_2 x$; $\hat{y}_{straight}(x) = w_1 + w_2 x$



Best constant predictor

$$\hat{y}_{const}(x) = w_1$$

- Loss corresponding to constant predictor is:

$$\mathcal{L}_{const}(w_1) = \sum_i (\hat{y}_{const}(x_i) - y_i)^2 = \sum_i (w_1 - y_i)^2$$

- Take derivative of loss w.r.t w_1:

$$\frac{\partial}{\partial w_1}[\mathcal{L}_{const}(w_1)] = \frac{\partial}{\partial w_1}\left[\sum_i (w_1 - y_i)^2\right]$$

$$= 2\sum_i (w_1 - y_i) = 2\sum_{i=1}^{N} w_1 - 2\sum_{i=1}^{N} y_i$$

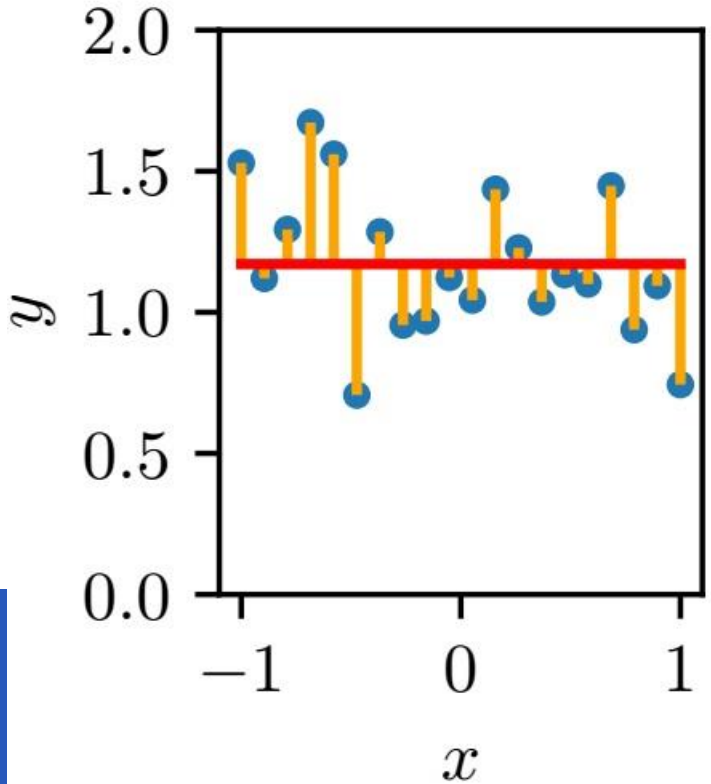
$$= 2\left(Nw_1 - \sum_{i=1}^{N} y_i\right)$$

Best constant predictor

$$\hat{y}_{const}(x) = w_1$$

- Setting equal to zero the result on the previous slide, unsurprisingly gives raise to the mean of y_i:

$$Nw_1 - \sum_{i=1}^{N} y_i = 0 \Rightarrow w_1 = \frac{1}{N}\sum_{i=1}^{N} y_i$$



Best slope predictor

$$\hat{y}_{slope}(x) = w_2 x_i$$

- Loss corresponding to slope predictor is:

$$\mathcal{L}_{slope}(w_2) = \sum_i \left(\hat{y}_{slope}(x_i) - y_i\right)^2 = \sum_i (w_2 x_i - y_i)^2$$

- Take derivative of loss w.r.t w_2:

$$\frac{\partial}{\partial w_2}\left[\mathcal{L}_{slope}(w_2)\right] = \frac{\partial}{\partial w_2}\left[\sum_{i=1}^{N}(w_2 x_i - y_i)^2\right] = \sum_{i=1}^{N}\frac{\partial}{\partial w_2}(w_2 x_i - y_i)^2$$

$$= \sum_{i=1}^{N} 2x_i(w_2 x_i - y_i) = 2w_2\sum_{i=1}^{N} x_i^2 - 2\sum_{i=1}^{N} x_i y_i$$

- Setting equal to 0 → $w_2 = \frac{\sum_{i=1}^{N} x_i y_i}{\sum_{i=1}^{N} x_i^2}$

Best straight-line predictor

$$\hat{y}_{straight}(x) = w_1 + w_2 x_i$$

- Loss corresponding to slope predictor is:

$$\mathcal{L}_{straight}(w_1, w_2) = \sum_i \left(\hat{y}_{straight}(x_i) - y_i\right)^2 = \sum_i (w_1 + w_2 x_i - y_i)^2$$

- w_1 and w_2 can be found by taking the derivatives of the loss w.r.t to both and setting them equal to 0:

$$\frac{\partial}{\partial w_1}\left[\mathcal{L}_{straight}(w_1, w_2)\right] = 0 \text{ and } \frac{\partial}{\partial w_2}\left[\mathcal{L}_{straight}(w_1, w_2)\right] = 0$$

- Without proof, the optimal value of w_2 can be found as:

$$w_2 = \frac{\frac{1}{N}\sum_{i=1}^{N} x_i y_i - \left(\frac{1}{N}\sum_{i=1}^{N} x_i\right)\left(\frac{1}{N}\sum_{i=1}^{N} y_i\right)}{\frac{1}{N}\sum_{i=1}^{N} x_i^2 - \left(\frac{1}{N}\sum_{i=1}^{N} x_i\right)^2}$$

Best straight-line predictor

$$w_2 = \frac{\frac{1}{N}\sum_{i=1}^{N} x_i y_i - \left(\frac{1}{N}\sum_{i=1}^{N} x_i\right)\left(\frac{1}{N}\sum_{i=1}^{N} y_i\right)}{\frac{1}{N}\sum_{i=1}^{N} x_i^2 - \left(\frac{1}{N}\sum_{i=1}^{N} x_i\right)^2}$$

- The above expression can be written more compactly as

$$w_2 = \frac{E\{xy\} - E\{x\}E\{y\}}{E\{x^2\} - (E\{x\})^2} \text{ or } w_2 = \frac{\sum_{i=1}^{N} x_i y_i - N\bar{x}\bar{y}}{\sum_{i=1}^{N} x_i^2 - N\bar{x}^2}$$

- Solving now for w_1, we have:

$$\frac{\partial}{\partial w_1}\left[\mathcal{L}_{straight}(w_1, w_2)\right] = 0$$

$$\frac{\partial}{\partial w_1}\left[\sum_{i=1}^{N}(w_1 + w_2 x_i - y_i)^2\right] = 0$$

Best straight-line predictor

$$\frac{\partial}{\partial w_1}\left[\sum_{i=1}^{N}(w_1 + w_2 x_i - y_i)^2\right] = 0$$

$$\sum_{i=1}^{N}\frac{\partial}{\partial w_1}(w_1 + w_2 x_i - y_i)^2 = 0$$

$$2\sum_{i=1}^{N}(w_1 + w_2 x_i - y_i) = 0$$

$$N w_1 = \sum_{i=1}^{N}(y_i - w_2 x_i)$$

$$w_1 = \frac{1}{N}\sum_{i=1}^{N}(y_i - w_2 x_i) = \bar{y} - w_2 \bar{x}$$

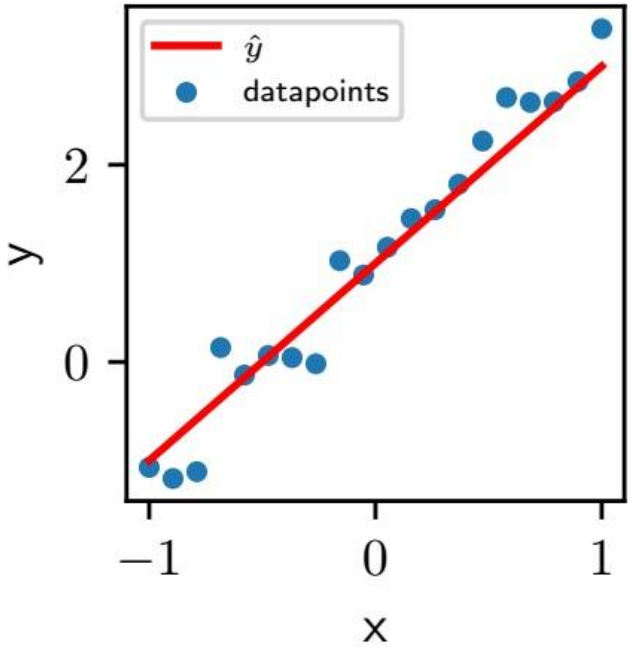
Best straight-line predictor

- The final result for straight line fitting is thus:

$$\hat{y}_{straight}(x) = w_1 + w_2 x$$

$$w_2 = \frac{\sum_{i=1}^{N} x_i y_i - N\bar{x}\bar{y}}{\sum_{i=1}^{N} x_i^2 - N\bar{x}^2}$$

$$w_1 = \frac{1}{N}\sum_{i=1}^{N}(y_i - w_2 x_i)$$



- The intercept, w_1, is the average error between y_i and $w_2 x_i$.
- If y_i is generally bigger than $w_2 x_i$, then the intercept is positive.

Best straight-line predictor - example

- Find the best least square fit by a linear function to the following data:

x	-1	0	1	2
y	0	1	3	9

$$\hat{y}_{straight}(x) = w_1 + w_2 x$$

$$w_2 = \frac{\sum_{i=1}^{N} x_i y_i - N\bar{x}\bar{y}}{\sum_{i=1}^{N} x_i^2 - N\bar{x}^2} = \frac{21 - 4 \times 0.5 \times 3.25}{6 - 4 \times 0.5^2} = 2.9$$

$$w_1 = \bar{y} - w_2 \bar{x} = 3.25 - 2.9 \times 0.5 = 1.8$$

$$y = 1.8 + 2.9x$$

Multivariable Regression

- So far only single-variable inputs, x, have been considered.
- In real-life, we might want to predict an outcome from many features, e.g. predict a product's sales from many different features (length, x_1, height, x_2, weight, x_3 etc.)
- For the sales example, the data is characterised by a number $D = 3$ of features and we assume that we have access to a number N of datapoints.
- All input features can thus be represented using an NxD matrix, $\boldsymbol{X}$:

$$\boldsymbol{X} = \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ \vdots & \vdots & \vdots \\ X_{N1} & X_{N2} & X_{N3} \end{bmatrix}$$ and the prediction is going to be a vector $y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$

- For each datapoint, there is thus an output y_i and a row vector of input features $\boldsymbol{x}_i^T = [X_{i1} \quad X_{i2} \quad X_{i3}]$
- The prediction becomes a function of the whole vector of input points:

$$\hat{y}(x_i) = \boldsymbol{x}_i^T \boldsymbol{w} = \sum_j X_{ij} w_j$$

Multivariable Regression

$$\hat{y}(x_i) = \boldsymbol{x}_i^T \boldsymbol{w} = \sum_j X_{ij} w_j$$

- In matrix form the prediction problem can be written:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w}$$

- The optimal weights can be found my minimising the L_2 norm (still LSE):

$$\boldsymbol{w}^* = arg\,\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2$$

- Taking the derivative w.r.t $\boldsymbol{w}$ and setting equal to 0:

$$-2\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}^*) = 0$$
$$\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}^*$$

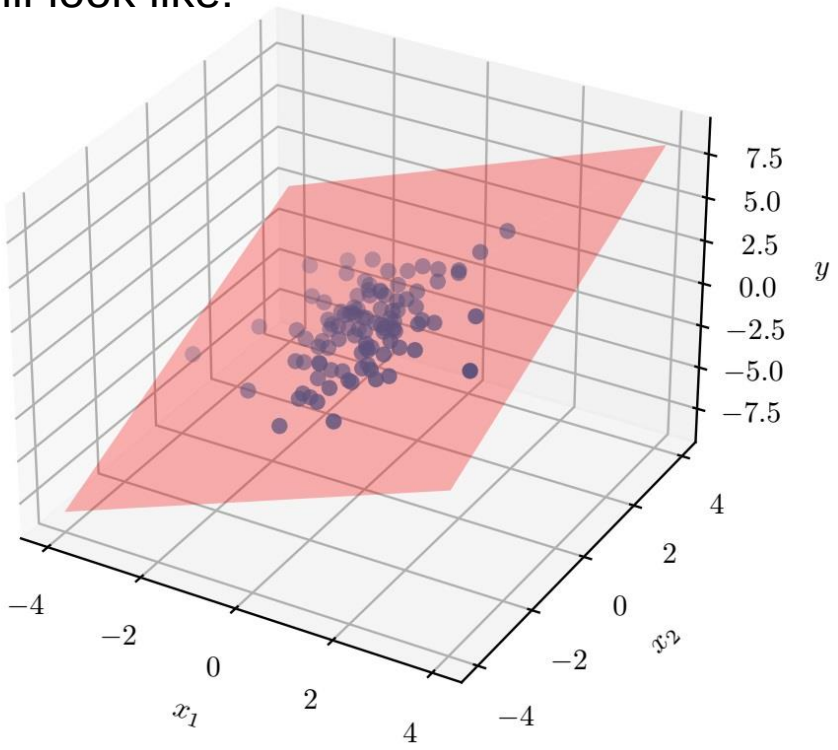
- Pre-multiplying by $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$:

$$\boldsymbol{w}^* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

Multivariable Regression

$$\boldsymbol{w}^* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

- When doing this by hand it's best to do $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$ and $\boldsymbol{X}^T\boldsymbol{y}$ separately, as the number of dimensions, D, is normally going to be smaller than the number of datapoints, N.
- A typical result will look like:



Regularisation

- So far, we have looked into minimising losses of the form

$$\mathcal{L} = \sum_{i} \left(y_i - \hat{y}(x_i)\right)^2 = \sum_{i=1}^{N} \left(y_i - \boldsymbol{x}_i^T \boldsymbol{w}\right)^2$$

- The above formulation doesn't allow for any information on what happens outside the training datapoints x_i. This leads to overfitting, which manifests by very high weights when the predictions go very big, away from the training points.
- Regularisation addresses the overfitting problem and consists in adding a term to the loss function which penalises very large weights:

$$\mathcal{L} = \sum_{i=1}^{N} \left(y_i - \boldsymbol{x}_i^T \boldsymbol{w}\right)^2 + \lambda \sum_{i=1}^{D} w_i^2$$

Regularisation

$$\mathcal{L} = \sum_{i=1}^{N} \left(y_i - \boldsymbol{x}_i^T \boldsymbol{w}\right)^2 + \lambda \sum_{i=1}^{D} w_i^2$$

- In matrix form, minimising the above expression amounts to:

$$\boldsymbol{w}^* = arg\,\min_{\boldsymbol{w}} (\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda \|\boldsymbol{w}\|_2^2)$$

- Taking the derivative w.r.t $\boldsymbol{w}$ and setting equal to 0:

$$-2\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}^*) + 2\lambda \boldsymbol{w}^* = 0$$
$$-\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}^* + \lambda \boldsymbol{w}^* = 0$$
$$(\boldsymbol{X}^T\boldsymbol{X} + \lambda \boldsymbol{I})\boldsymbol{w}^* = \boldsymbol{X}^T\boldsymbol{y}$$

- Pre-multiplying by $(\boldsymbol{X}^T\boldsymbol{X} + \lambda \boldsymbol{I})^{-1}$:

$$\boldsymbol{w}^* = (\boldsymbol{X}^T\boldsymbol{X} + \lambda \boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

- Compare this result to the original (with no regularization):

$$\boldsymbol{w}^* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

Regression with nonlinear features

- Assume now we want to fit a nonlinear model, such as

$$\hat{y} = w_1 + w_2 x + w_3 x^2$$

- This can be tackled similarly to multivariable regression:
  - Form multiple inputs by applying multiple functions to the actual inputs x_i:

$$\boldsymbol{x}^T(x) = [f_1(x) \quad f_2(x) \quad f_3(x)]$$

- The NxD (in general, but here $D = 3$) matrix, $\boldsymbol{X}$, is now:

$$\boldsymbol{X} = \begin{bmatrix} f_1(x_1) & f_2(x_1) & f_3(x_1) \\ f_1(x_2) & f_2(x_2) & f_3(x_2) \\ \vdots & \vdots & \vdots \\ f_1(x_N) & f_2(x_N) & f_3(x_N) \end{bmatrix}$$

and the prediction is going to be a vector

$$y = \boldsymbol{x}^T(x)\boldsymbol{w} = w_1 f_1(x) + w_2 f_2(x) + w_3 f_3(x)$$

Regression with nonlinear features

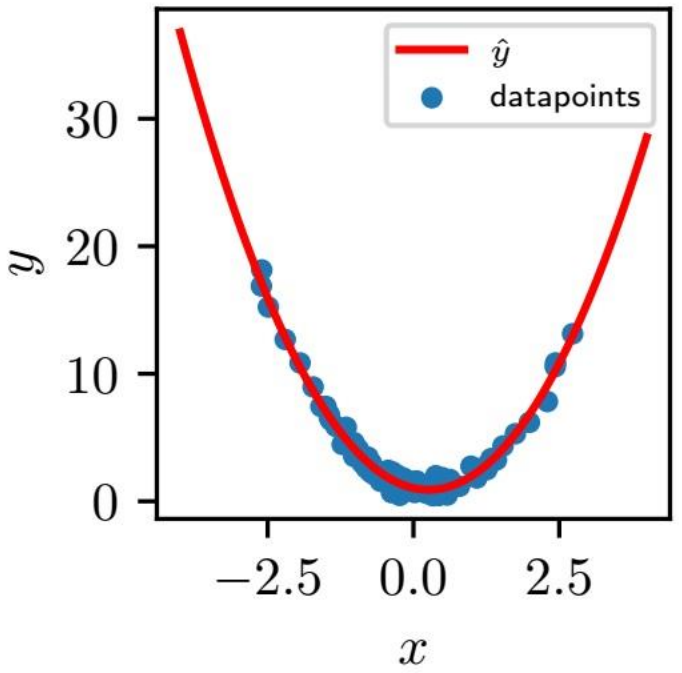
$$\hat{y} = w_1 + w_2 x + w_3 x^2$$

- For the given model, we now choose

$$f_1(x) = 1$$
$$f_2(x) = \mathrm{x}$$
$$f_3(x) = x^2$$

- Feature vector will be $\boldsymbol{x}^T(x) = [1 \quad x \quad x^2]$
- Matrix $\boldsymbol{X}$ will be:

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_N & x_N^2 \end{bmatrix}$$



- Then use again

$$\boldsymbol{w}^* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$