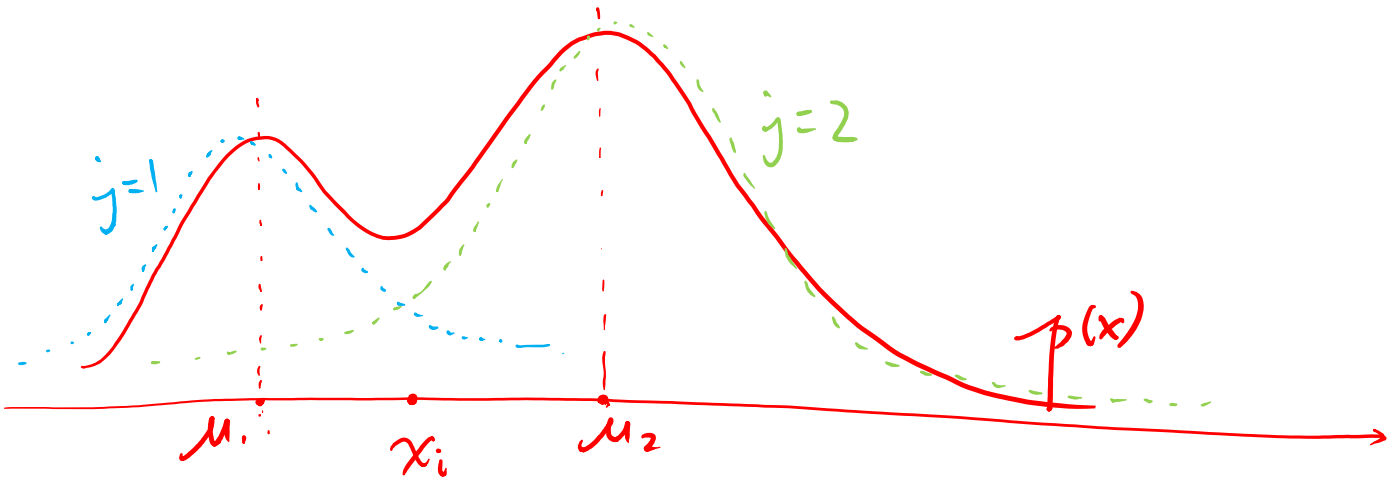
CSC345/M45:
Big Data & Machine Learning
(linear regression)

Prof. Xianghua Xie

x.xie@swansea.ac.uk
http://csvision.swan.ac.uk
224 Computational Foundry, Bay Campus



How to determine which component should data x_i belong to?

A. compare

$|\mu_1 - x_i|$

$|\mu_2 - x_i|$

B. compare

$p(x_i | j=1)$

$p(x_i | j=2)$

C. compare

$p(j=1 | x_i)$

$p(j=2 | x_i)$

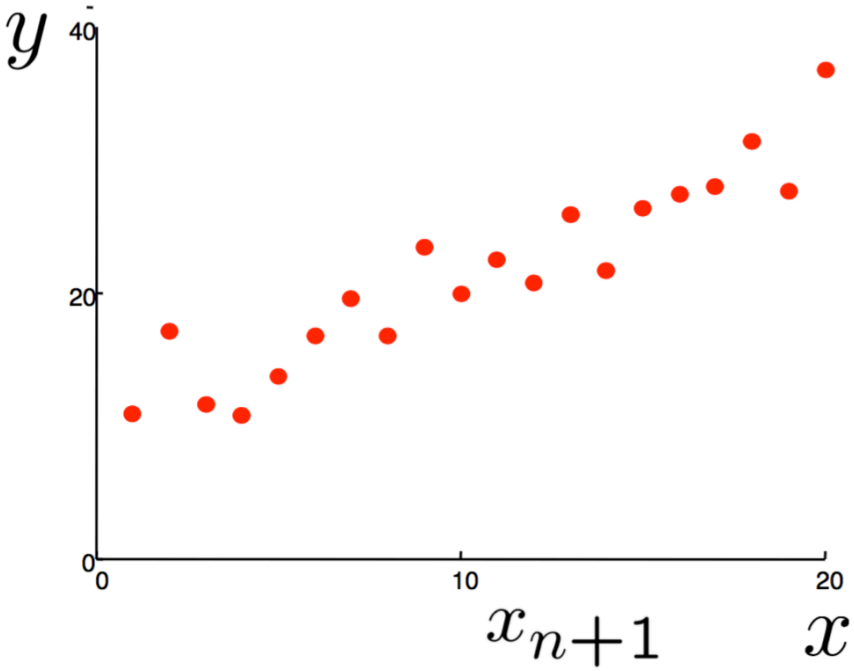
Regression

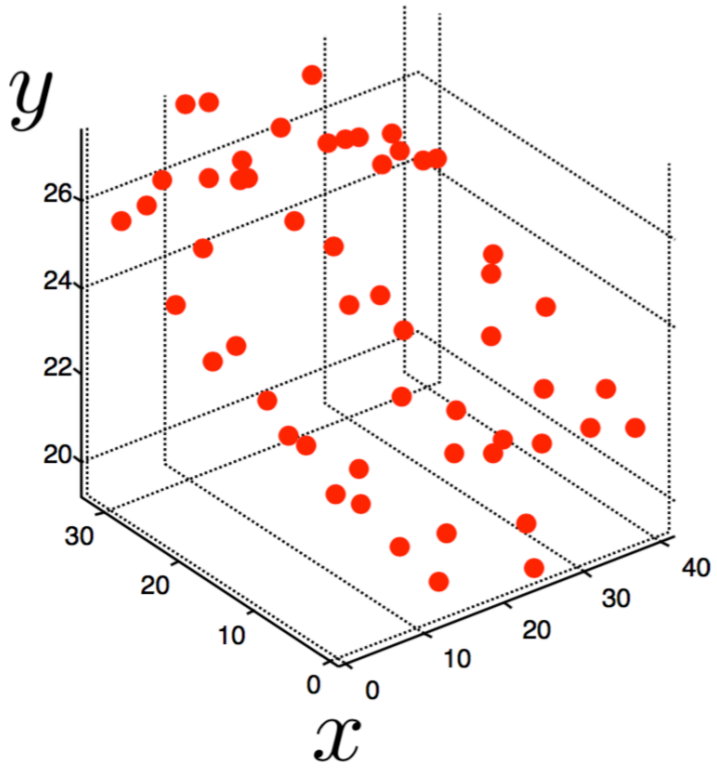
- Supervised Learning
  - The given data contains both the data themselves and also their correct answers
  - The task is to train the machine to predict answers for unseen data

- Regression
  - (Mostly) continuous output
  - We will be focusing on linear regression
    - However, this can fit nonlinear data as well
  - This technique is a foundation for many advanced algorithms
    - E.g. represent decision boundaries

Linear Regression

- Given examples $(x_i,y_i)_{i=1,\ldots,n}$
- Predict y_{n+1}, given a new point x_{n+1}



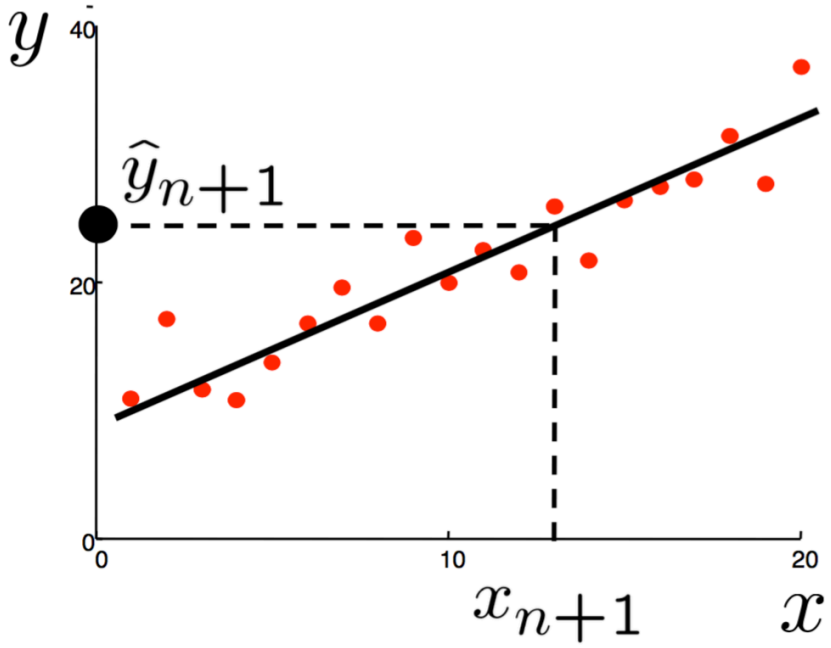


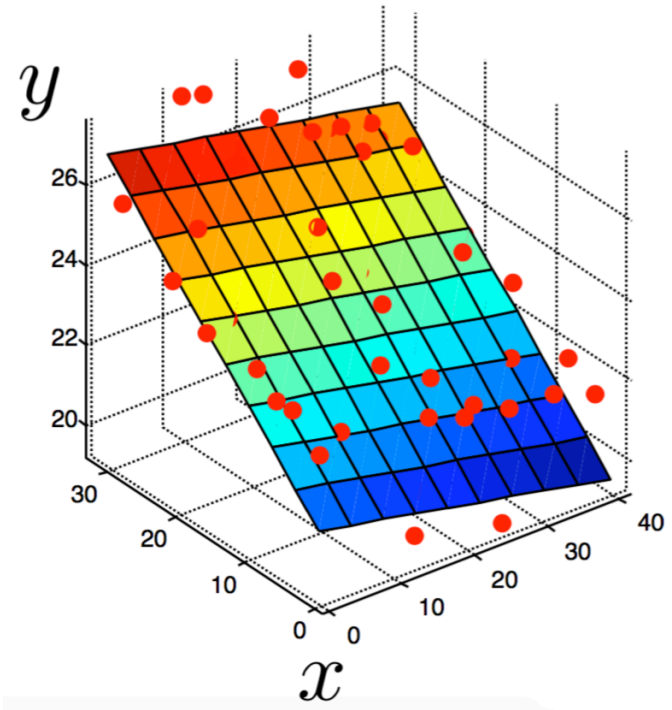
Linear Regression

- We wish to estimate $\hat{y}$ by a linear function of data x:

$$\hat{y}_{n+1} = w_0 + w_1 x_{n+1,1} + w_2 x_{n+1,2}$$

  - In this example, the input data is two-dimensional, e.g. $x_{n+1} = (x_{n+1,1}, x_{n+1,2})^T$
  - $\omega_0, \omega_1, \omega_2$ are the parameters to be estimated





Linear Regression

- Recap: $\hat{y}_{n+1} = w_0 + w_1 x_{n+1,1} + w_2 x_{n+1,2}$

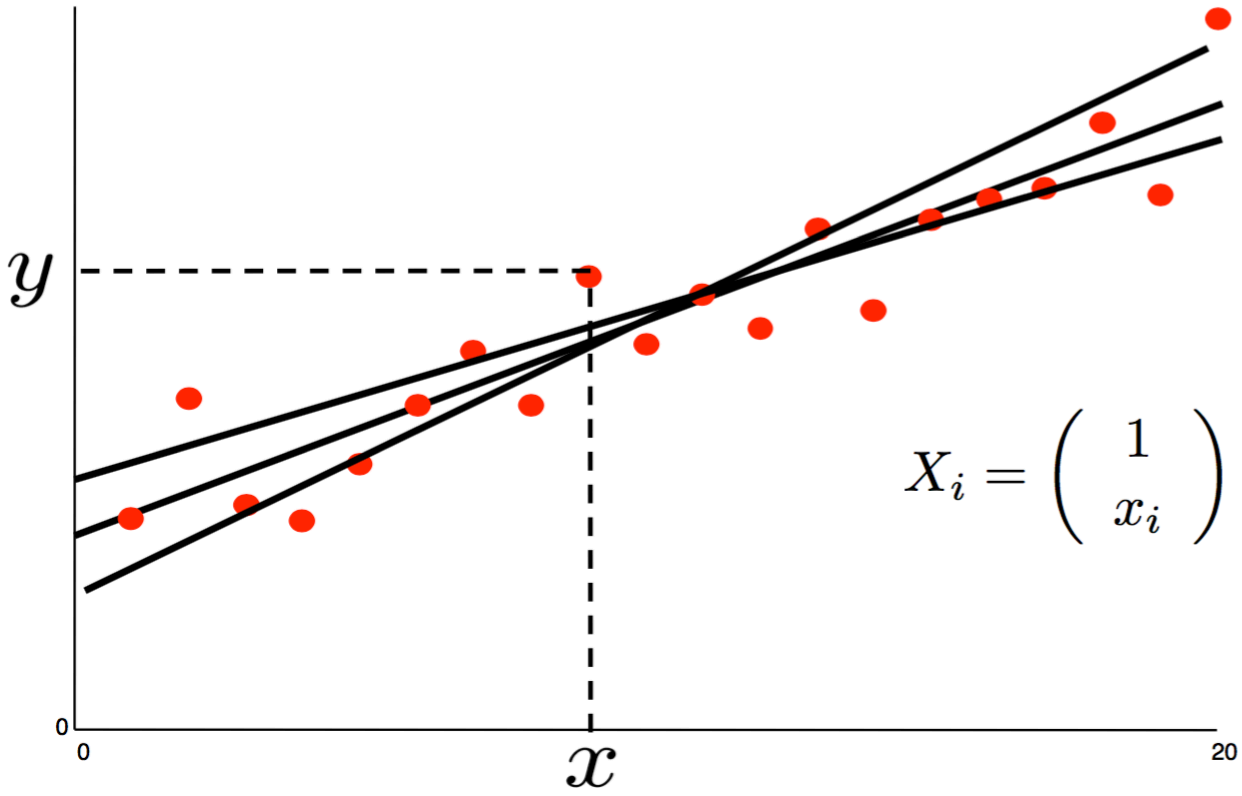
- A more general form of linear regression:

$$\hat{y}_{n+1} = \omega^T x_{n+1}$$

  - where $\omega = (\omega_0, \omega_1, \omega_2, \dots)^T$, $x_{n+1} = \left(1, x_{n+1,1}, x_{n+1,2}, \dots\right)^T$
    - T denotes transpose, that is a row vector becomes a column vector
    - We add 1 as the first component of data x so that we can then write the linear equation as a dot product between two vectors

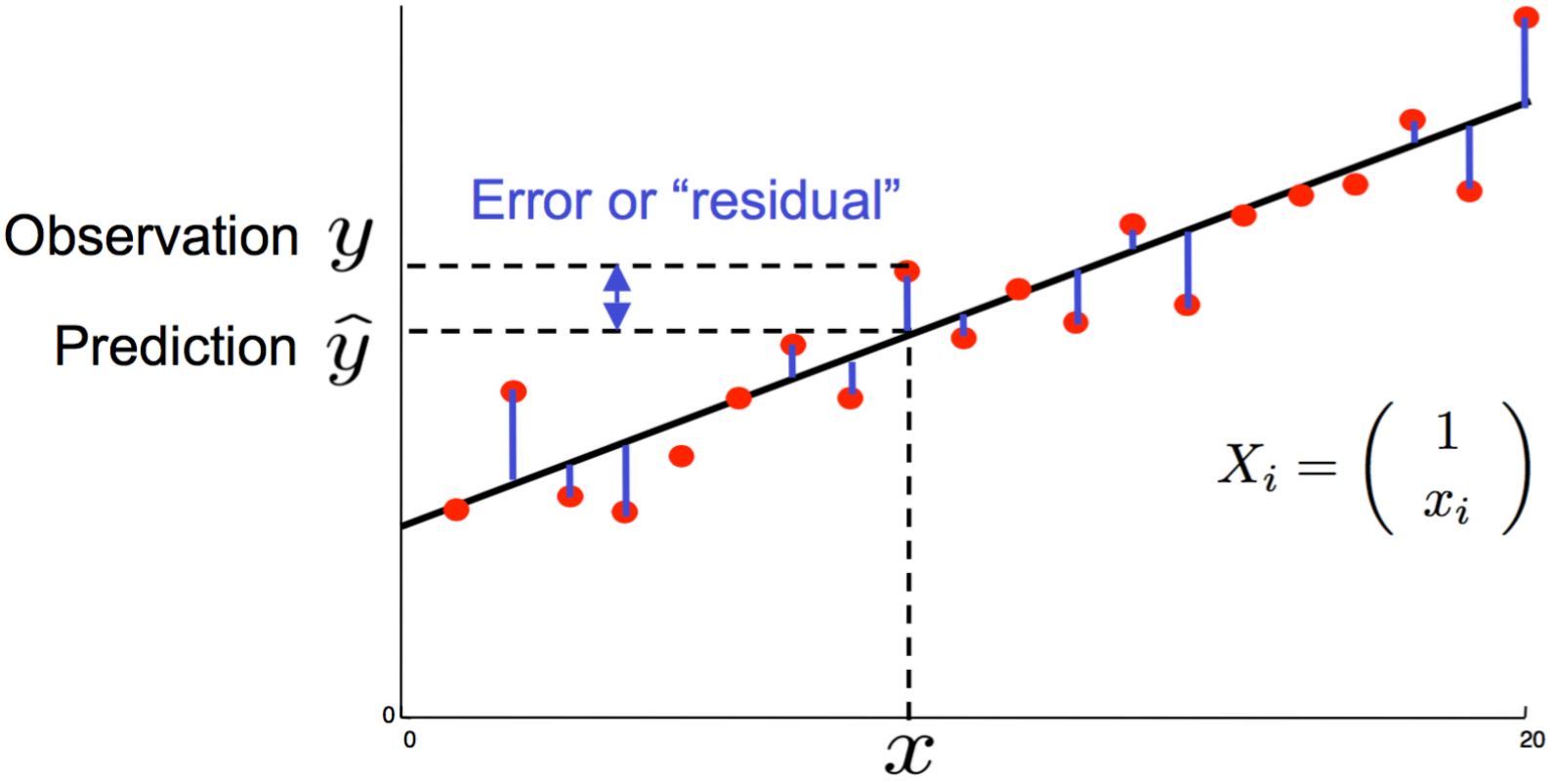
Linear Regression

- Choose the regressor
  - Of the many regression fits that approximate the data, which one is the “best” or “optimal”?



Linear Regression

- In order to evaluate the quality of the regression, we define a "cost function" to quantify the quality of fitting on the supervised training data.

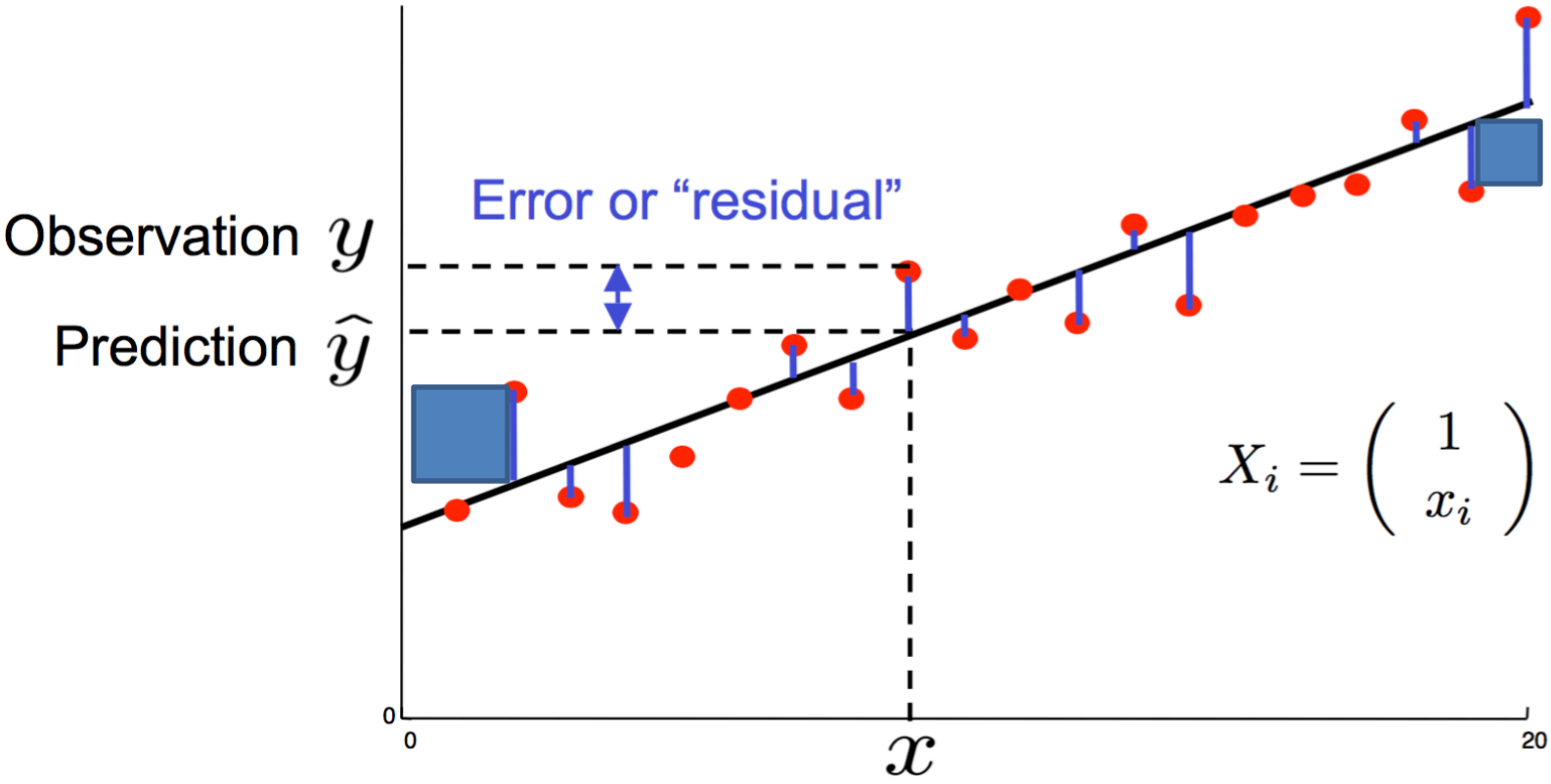


Linear Regression

$$\hat{y}_{n+1} = \omega^T x_{n+1}$$

- Cost function: Least Mean Squares (LMS)
  - Solution minimises sum of squared errors (SSE)

$$E = \frac{1}{2}\sum_{i=1}^{n}(w^\top x_i - y_i)^2 = \sum_{i=1}^{n} E_i$$



$$E = \frac{1}{2}\sum_{i=1}^{n}(w^\top x_i - y_i)^2 = \sum_{i=1}^{n} E_i$$

Linear Regression

- To find the solution to LMS, we once again employ gradient descent optimisation

$$E = \frac{1}{2}\sum_{i=1}^{n}(w^\top x_i - y_i)^2 = \sum_{i=1}^{n} E_i$$

$$w^{t+1} := w^t - \alpha \frac{\partial}{\partial w} E$$

  - where

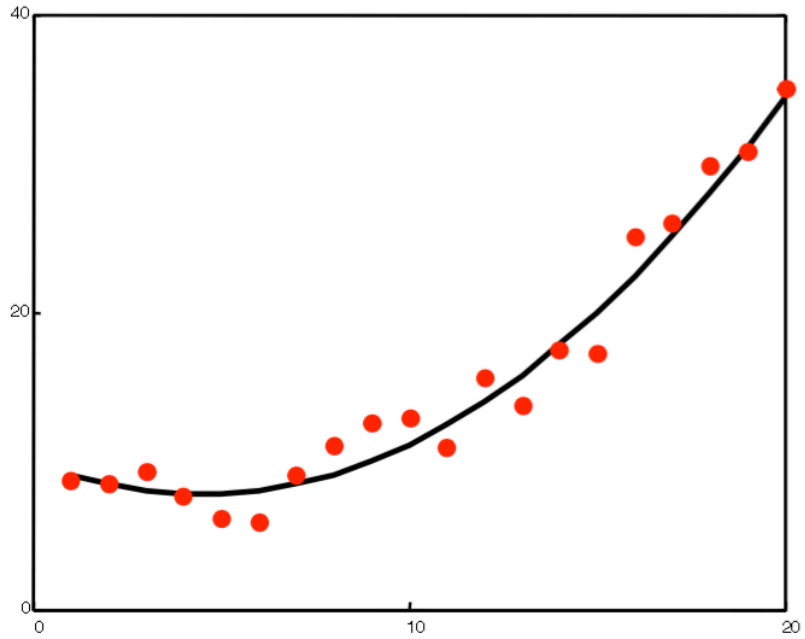
$$\frac{\partial}{\partial w}E = \sum_{i=1}^{n}\frac{\partial}{\partial w}E_i \qquad \frac{\partial}{\partial w}E_i = \frac{1}{2}\frac{\partial}{\partial w}(w^\top x_i - y_i)^2 = (w^\top x_i - y_i)x_i$$

  - Compute the partial derivatives: $\frac{\partial}{\partial w}E_i$
  - An iterative method
  - Alpha is a constant that controls the increments in each iteration

Linear Regression

- Nonlinear transformations
  - Linear linear models become powerful function approximators when we consider non-linear transformations

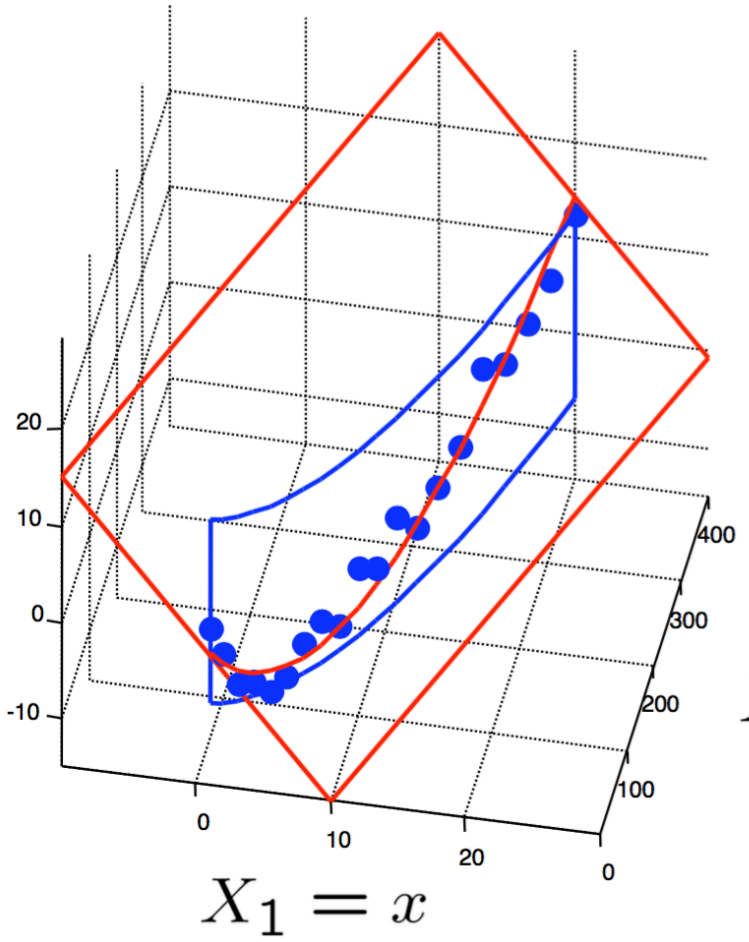
$$X_i = \begin{pmatrix} 1 \\ x_i \\ x_i^2 \end{pmatrix} \Longrightarrow \hat{y}_i = w_0 + w_1 x_i + w_2 x_i^2$$



- Predictions are still linear in the coefficients w;
- Hence, the math is exactly the same!

Linear Regression

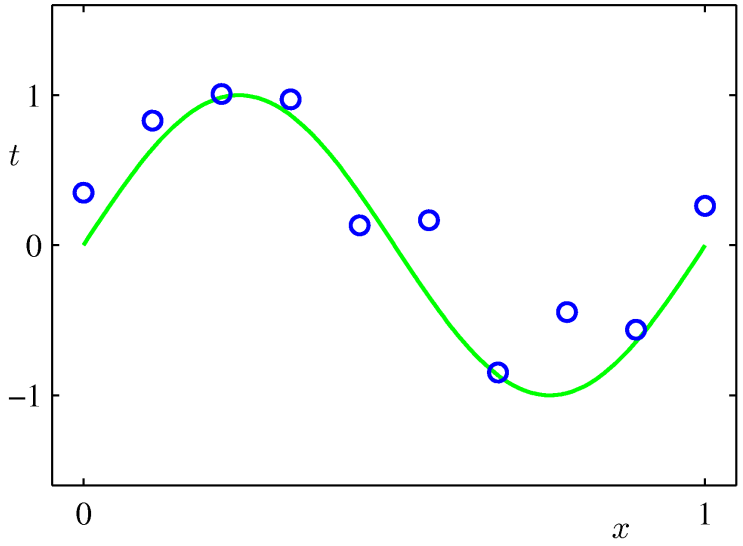
- Non-linear transformations



$$\hat{y}_i = w_0 + w_1 x_i + w_2 x_i^2$$

Linear Basis Function Models

- Polynomial fitting:



$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

- Generally,

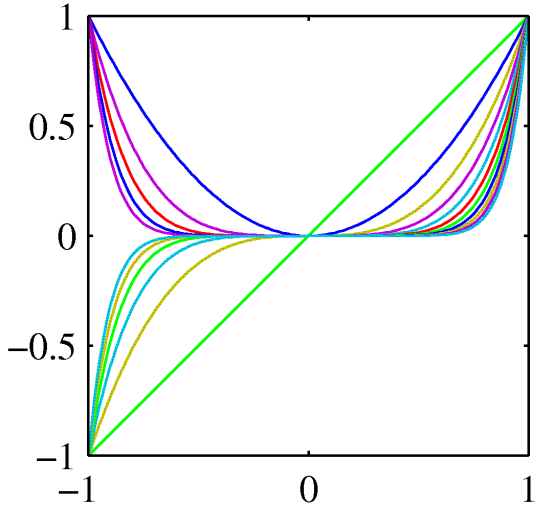
$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

where phi is known as the basis function

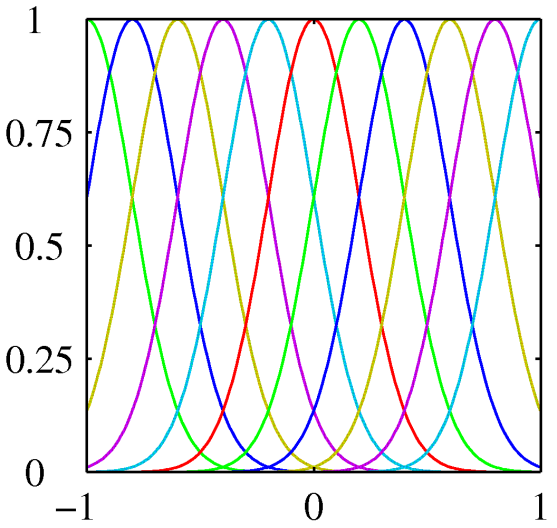
Note, the function is nonlinear in x, but the fitting is linear in terms of w.

Linear Basis Function Models

- Polynomial basis: $\phi_j(x) = x^j$.
- These are global; a small change in x affect all basis functions.

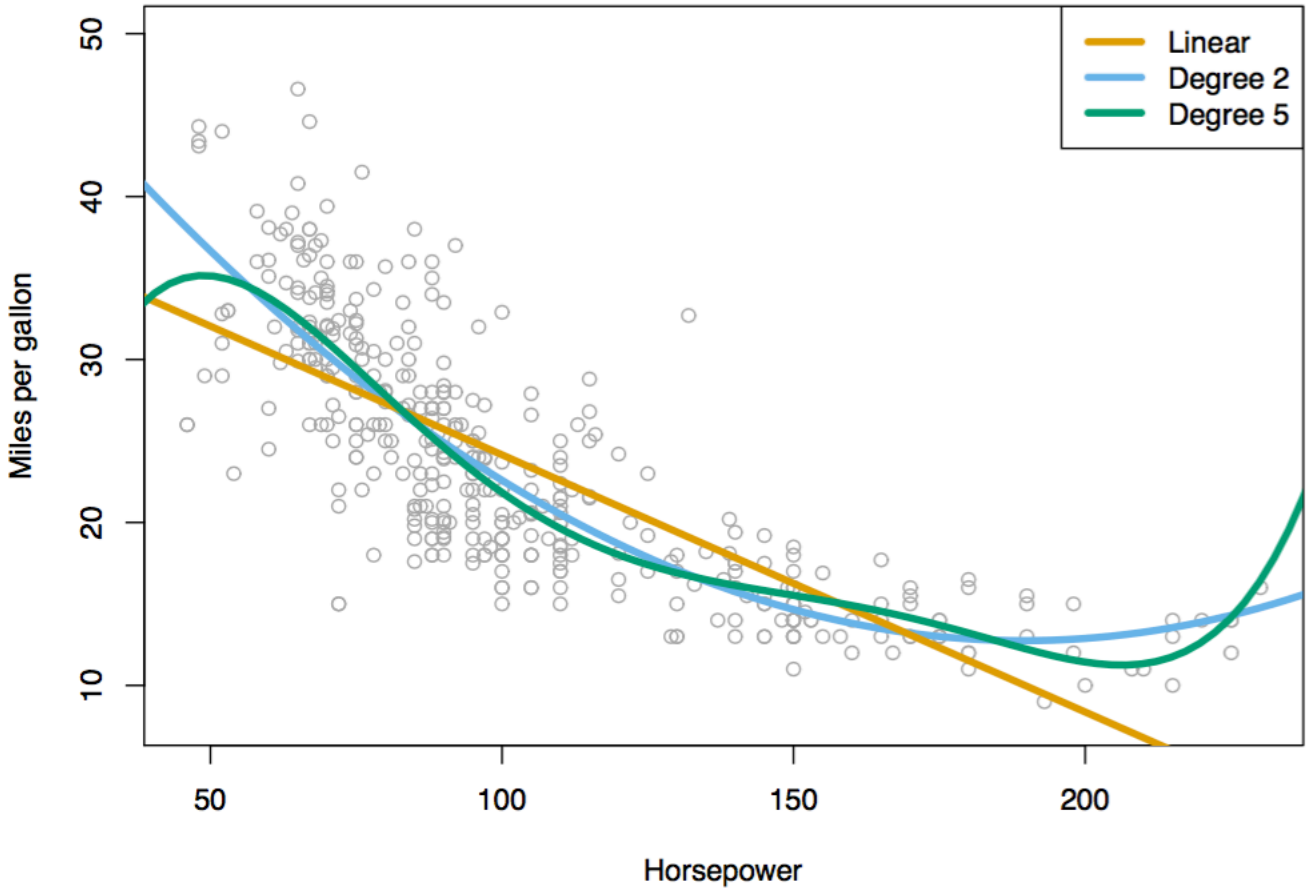


- Gaussian basis: $\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$
- These are local; a small change in x only affect nearby basis functions. mu and s control location and scale (width).



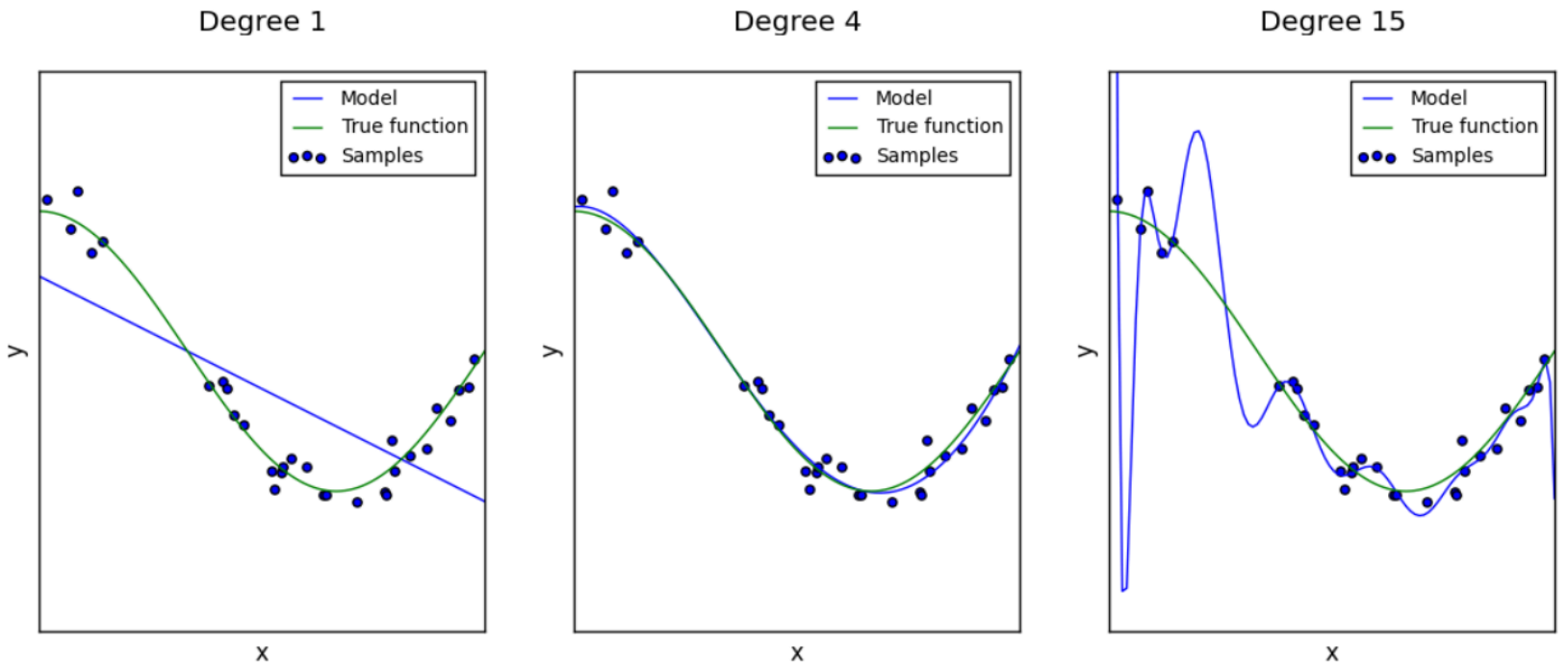
Linear Regression

- Example



Linear Regression

- Under-fitting and Over-fitting
  - E.g. fitting a polynomial function to data with varying degrees of complexity
    - SSE will go lower for the training data as the complexity increases, but the model does not generalise well



Linear Regression

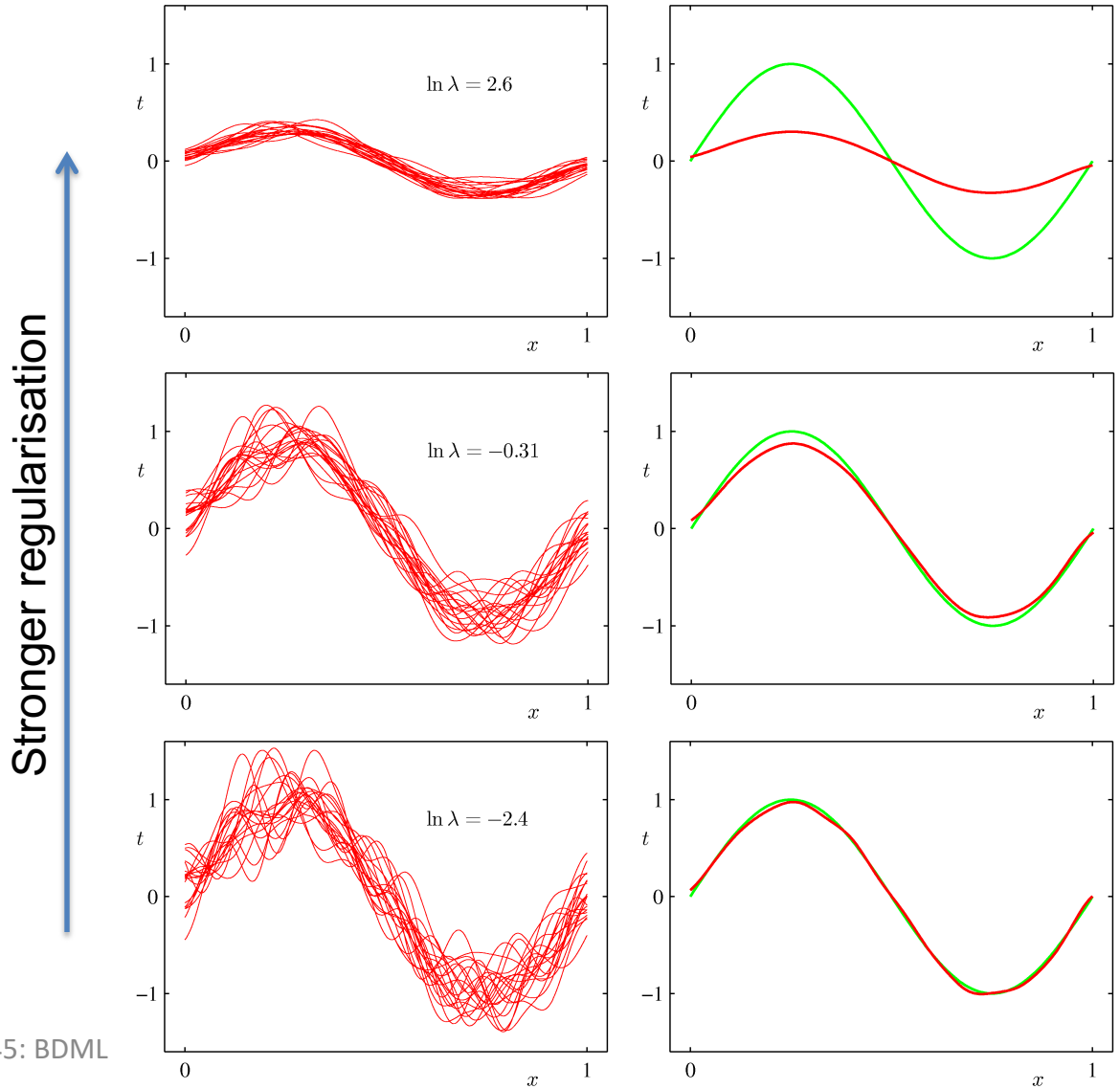
- Regularisation is often necessary to minimise the risk of overfitting
  - E.g. quadratic regularisation (add an extra term to the cost function):

$$E = \frac{1}{2}\sum_{i=1}^{n}(w^\top x_i - y_i)^2 \quad \boxed{+\frac{\lambda}{2}\omega^T\omega}$$

    - Lambda λ is called the regularisation coefficient. It is a constant that controls the amount of regularisation.

Linear Regression

- Example: varying degree of regularisation

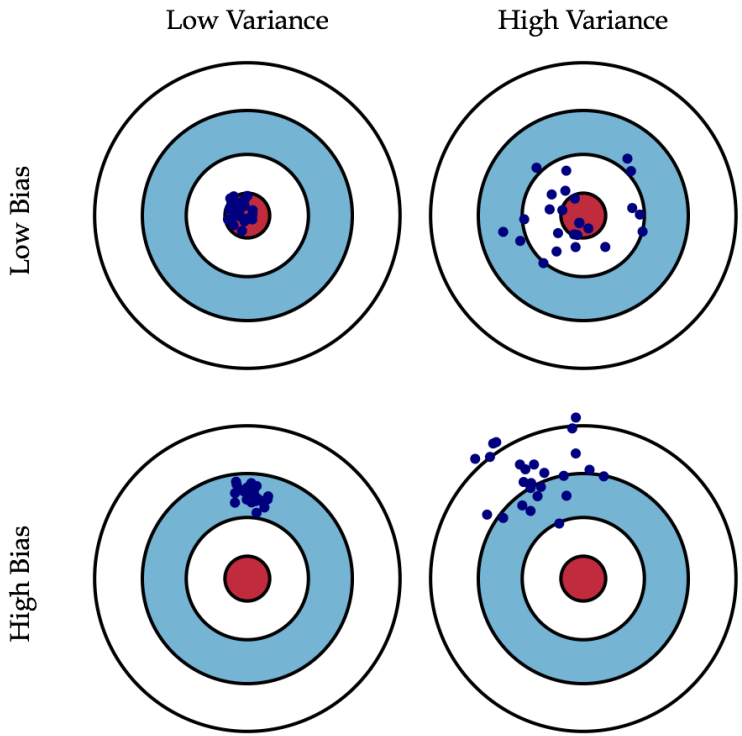


- Green: groundtruth
- Red on the left column: results from multiple runs
- Red on the right column: average result

Bias and Variance

- Bias: the difference between the expected (or average) prediction of the model and the correct value
- Variance: the variability of a model prediction for a given data point

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



Bias and Variance

- From these plots, we note that an over-regularised model (large lambda) will have a high bias, while an under-regularised model (small lambda) will have a high variance.