COMP24112: Machine Learning

Chapter 5: Loss Functions I

Dr. Tingting Mu

Email: tingting.mu@manchester.ac.uk

Content

- Typical approaches of constructing losses for **regression**
  – Non-probabilistic losses
  – Probabilistic losses

"Data + Model" in Supervised Learning

- "Data + Model" Strategy:

Training Data: D_{tr} ⇒ Parametric Model: $f(\theta, x)$

parameters θ

Parameters to be optimised.

Hyper-parameters to be selected.

"Data + Model" in Supervised Learning

- After the learning (or called training) is finished, the final product is:

A trained model:

$$f\left(\theta\left(D_{tr}\right), x\right)$$

- The process of using the trained model on unseen data (or called query data, test data) is called inference.

$$answer = f\left(\theta\left(D_{tr}\right), x_{query}\right)$$

Loss Function

- Loss function is essential in training, computed using the training data.
- It decides how good the model parameters are, how well the model fits your training data.
- Other names: error function, cost function, or more general, objective function.
- To train a machine learning model, you pick a loss function $O(\theta)$. Then:
  - Minimise it, if *O* evaluates how bad the model is.
  - Maximise it, if *O* evaluates how good the model is.
- Supervised learning: $O\left(\theta, D_{tr}\right)$

Losses for Training Regression Models

Training Data: $D_{tr} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$

Feature vector: $\mathbf{x}_i \in R^d$

Target output: $\mathbf{y}_i \in R^c$ where $\mathbf{y}_i = \left[y_{i1}, y_{i2}, \ldots y_{ic} \right]$

Non-probabilistic Regression Losses

Regression Error based Loss

- Recall RMSE?

- Some regression losses are simplified versions of RMSE computed using training samples.

The prediction for each training sample is computed by $\hat{\mathbf{y}}_i = f\left(\theta, \mathbf{x}_i\right)$.

– **Sum-of-squares error loss:** $O\left(\theta\right) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{c}\left(\hat{y}_{ij} - y_{ij}\right)^2$

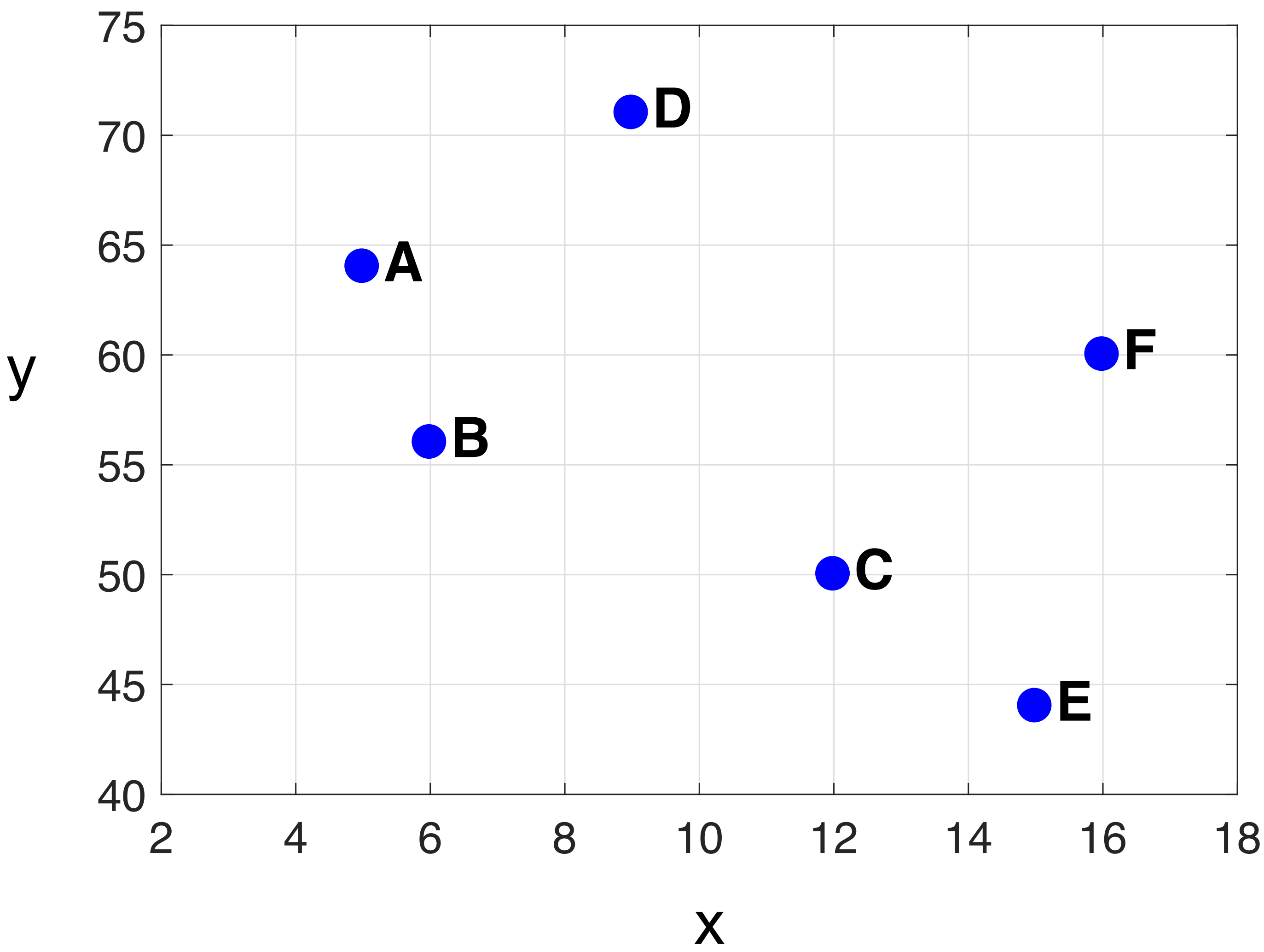
– **Mean-squared error loss:**

A single-output example (*c=1*): $\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_i - y_i\right)^2$

A Regression Example

MANCHESTER 1824
The University of Manchester

- Fit a **linear model** using the following six training data samples.



x	y
5	64
6	56
12	50
9	71
15	44

$$\hat{y} = f(x) = w_0 + w_1 x$$

A Regression Example

- Sum-of-squares error loss for training the linear model (finding the best w_0 and w_1):

$$O = \frac{1}{2}\left[(\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2 + (\hat{y}_3 - y_3)^2 + (\hat{y}_4 - y_4)^2 + (\hat{y}_5 - y_5)^2 + (\hat{y}_6 - y_6)^2\right]$$

$$= \frac{1}{2}\sum_{i=1}^{6}(\hat{y}_i - y_i)^2$$

$$= \frac{1}{2}\sum_{i=1}^{6}(w_1 x_i + w_0 - y_i)^2$$

A Regression Example

MANCHESTER 1824
The University of Manchester

- Incorporate the values of x_i and y_i to the error function:

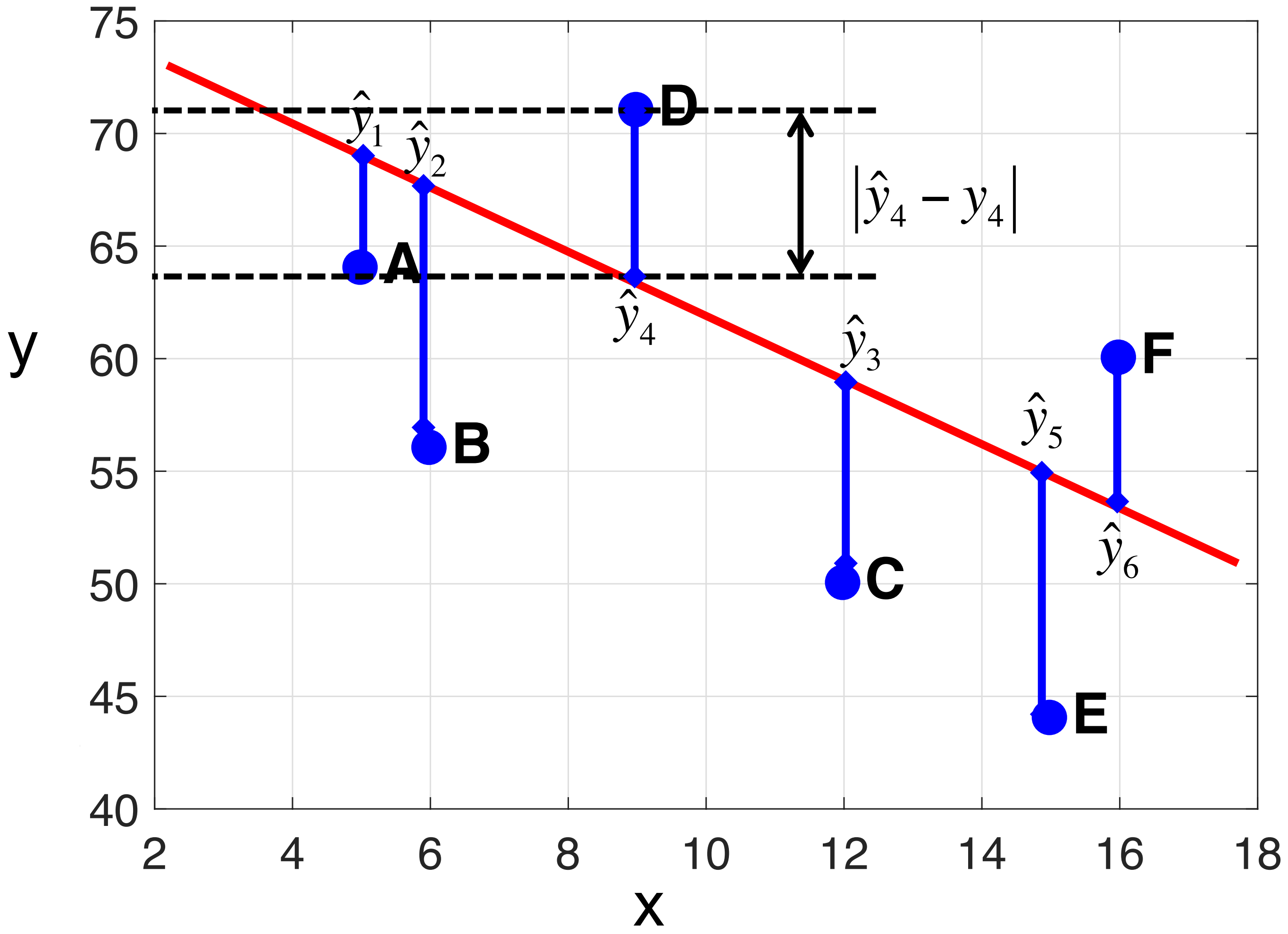
$$\boxed{\begin{aligned} &x_1 = 5,\ y_1 = 64 \\ &x_2 = 6,\ y_2 = 56 \\ &x_3 = 12,\ y_3 = 50 \\ &x_4 = 9,\ y_4 = 71 \\ &x_5 = 15,\ y_5 = 44 \\ &x_6 = 16,\ y_6 = 60 \end{aligned}}$$

$$O\left(w_1, w_0\right) = \frac{1}{2}\Big[\left(5w_1 + w_0 - 64\right)^2 + \left(6w_1 + w_0 - 56\right)^2 + \left(12w_1 + w_0 - 50\right)^2 + \left(9w_1 + w_0 - 71\right)^2 + \left(15w_1 + w_0 - 44\right)^2 + \left(16w_1 + w_0 - 60\right)^2\Big]$$

- We want to minimise this training loss: $\boxed{\min O\left(w_1, w_0\right)}$

A Regression Example

- Geometrically, to minimise this regression error loss enables you to find the best red line to have the shortest blue distances on average.



Linear Least Squares (LLS)

- Linear least squares: To train a linear model by minimising the sum-of-squares error.
  - Single-output case:

$$\min O(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \mathbf{w}^T\tilde{\mathbf{x}}_i\right)^2$$

  - Multi-output case:

$$\min O(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{c}\left(y_{ij} - \mathbf{w}_j^T\tilde{\mathbf{x}}_i\right)^2$$

Regularised Linear Least Squares

- A regularisation term can be added to the error function. For instance, in the single-output case, we have

$$\min O_\lambda(\mathbf{w}) = \text{sum of suqares error } + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 = \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \mathbf{w}^T\tilde{\mathbf{x}}_i\right)^2 + \boxed{\frac{\lambda}{2}\mathbf{w}^T\mathbf{w}}$$

$$\mathbf{w}^T\mathbf{w} = \sum_{j=1}^{d+1} w_j^2$$

λ is a positive real-valued number set by the user.

- Other type of regulariser:

$$\min O_\lambda(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \mathbf{w}^T\tilde{\mathbf{x}}_i\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{d+1}\left|w_j\right|^q$$

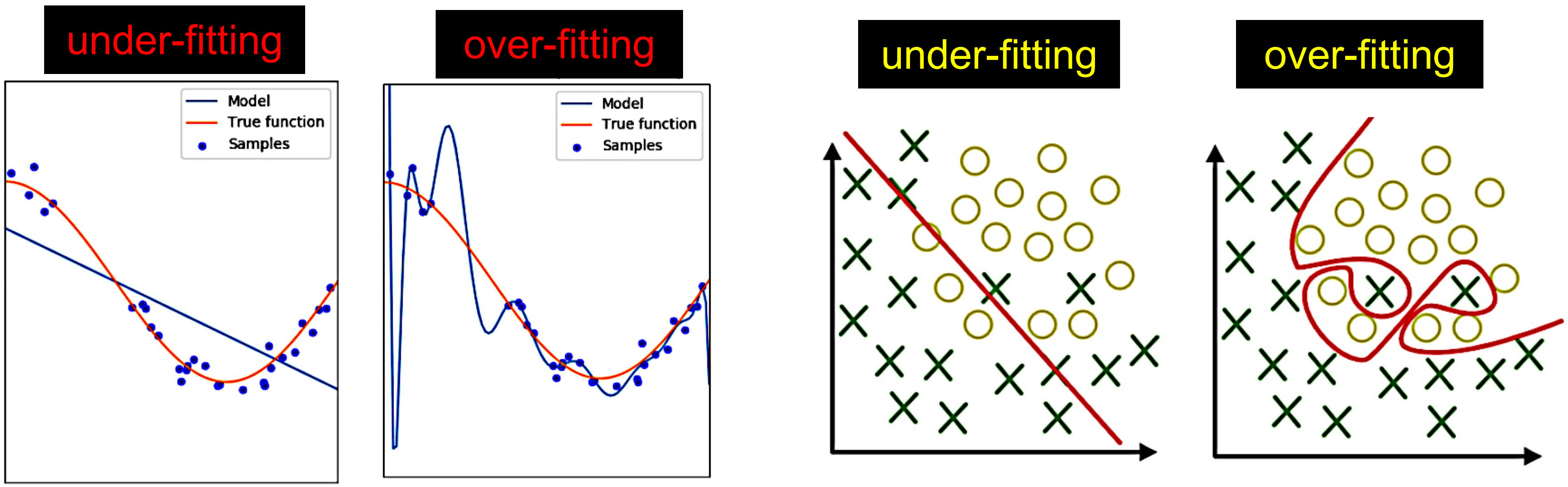
Here *q* is a positive integer set by the user.
- ❖ The case of *q=1* (l_1-regularisation) for regression is known as lasso.
- ❖ The case of *q=2* (l_2-regularisation) for regression is known as ridge regression.

- Regularisation prevents the model from over-fitting to training data.

- When λ is too large, it will lead to under-fitting though.

Why Regularisation?

- Over-fitting: Fit too closely to a particular set of data (e.g., training data), and may therefore fail to fit new data.
- Under-fitting: Cannot capture the underlying trend of the training data.
- Prevent over-fitting, e.g., $O_\lambda(\mathbf{w})$ gives less emphasis to the sum of squares error of the training data.



Probabilistic Regression Losses

Likelihood

- **Likelihood:** Given the observed data, it is the conditional probability assumed for the observed data given some parameter values.

$$\text{Likelihood}(\theta \mid \text{data}) = p(\text{data} \mid \theta)$$

- **Log likelihood**: Take the natural logarithm of the likelihood.

Likelihood Maximization

- A model can be trained by **maximising** the likelihood (or log likelihood) function of the training samples.

- Assume independence between samples.

  – **Maximum likelihood estimator** (MLE):

$$\max_{\theta} = \prod_{i=1}^{N} p(\mathbf{x}_i, y_i \mid \theta)$$

  – **Log likelihood maximisation**:

$$\max_{\theta} = \sum_{i=1}^{N} \log p(\mathbf{x}_i, y_i \mid \theta)$$

Example: MLE for Linear Regression

MANCHESTER 1824
The University of Manchester

- Likelihood of N training samples:

$$L = \prod_{i=1}^{N} N\left(y_i \middle| \mathbf{w}^T \tilde{\mathbf{x}}_i, \sigma^2\right)$$

Assume a Gaussian distribution for likelihood estimation.

$$p(\text{data}_i \,|\, \theta) = p(y_i \,|\, \theta) = N\left(y_i \,|\, \mathbf{w}^T \tilde{\mathbf{x}}_i, \sigma^2\right)$$

- Log-likelihood:

$$O = \ln(L) = \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) - \beta \boxed{\frac{1}{2}\sum_{i=1}^{N}\left(y_i - \mathbf{w}^T \tilde{\mathbf{x}}_i\right)^2}$$

where $\beta^{-1} = \sigma^2$

Sum-of-squares error function!

- In this case, an MLE is equivalent to a sum-of-squares error minimizer.