1. Linear regression is a supervised machine learning algorithm use to predict a continuous variable by using the training data that is fed into the model.

Linear regression assumes a linear relation ship betw the input features $x \in R^d$ and output y, goal is to find parameter β such that $\hat{y}$ (prediction) are closest to y (actual outputs)

* squaring penalizes large errors more than small ones
* ensure the fn is differentiable, hence able to take gradient
* Direct summation may ~~lead to~~ allow +ve & -ve errors to cancel each other. squaring answers all errors are +ve

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}_{n\times 1} \qquad X = R^{n \times (d+1)} \qquad \beta = R^{(d+1)\times 1}$$

(b). $J = \frac{1}{2} \|y - X\beta\|^2$

$$= \frac{1}{2}(y - X\beta)^T (y - X\beta)$$

$$= \frac{1}{2}(y^T - \beta^T X^T)(y - X\beta)$$

$$= \frac{1}{2}(y^T y - y^T X\beta - \beta^T X^T y - \beta^T X^T X\beta)$$

$$\because (y^T X\beta)^T = \beta^T X^T y = \text{scalar}$$

$$\nabla J(\beta) = \nabla\left(\frac{1}{2}(y^T y - 2\beta^T X^T y + \beta^T X^T X\beta)\right)$$

$$\because \frac{\partial (x^T a)}{\partial x} = a$$

$$= \frac{1}{2}(-2X^T y + 2X^T X\beta)$$

$$\Rightarrow -X^T y + X^T X\beta = 0$$

$$X^T X\beta = X^T y$$

$$\beta = (X^T X)^{-1} X^T y$$

b). Direct conversion of X^TX in the normal equation can be problematic as taking inverse of a high dimensional matrix is computationally extensive as it takes both long time & memory.

Gradient descent takes small steps rather than take inverse of whole matrix at once, updates the parameters of model constantly without starting calculation from start.

7.

(a). Backpropagation is an algorithm use to calculate gradients by propagating the error backward from output to the input layer.

• chain rule help us to break down complex derivative of loss fn into simpler steps.

(b).
$$\frac{\partial L}{\partial z_2} = a_2 - y$$

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial z_2} \cdot \frac{\partial z_2}{\partial w_2}$$

$\because z_2 = w_2 a_1 + b_2$

$$\frac{\partial z_2}{\partial w_2} = a_1$$

$$= (a_2 - y) a_1$$

$$\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial z_2} \cdot \frac{\partial z_2}{\partial b_2}$$

$z_2 = w_2 a_1 + b_2$

$$\frac{\partial z_2}{\partial b_2} = 1$$

$$\frac{\partial L}{\partial b_1} = a_2 - y$$

$$\frac{\partial L}{\partial z_1} = \frac{\partial L}{\partial z_2} \frac{d z_2}{d a_1} \frac{\partial a_1}{\partial z_1}$$

$$\frac{\partial a_1}{\partial z_1} = a_1(1 - a_1)$$

$$= (a_2 - y) w_2$$

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial z_1}\frac{\partial z_1}{\partial w}$$

$$= (a_2 - y)\, w_2\, a_1(1-a_1)x$$

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial z_1}\frac{\partial z_1}{\partial b_1}$$

$$= (a_2 - y)\, w_2\, a_1(1-a_1)$$

©. To update the weights we subtract a fraction of the gradient from the current value

$$w_1' = w_1 - \alpha \frac{\partial L}{\partial w_1}$$

$$w_2' = w_2 - \alpha \frac{\partial L}{\partial w_2}$$

$$b_1' = b_1 - \alpha \frac{\partial L}{\partial b_1}$$

$$b_2' = b_2 - \alpha \frac{\partial L}{\partial b_2}$$

$\alpha \rightarrow$ controls the step size

If α ~~is~~ too small
↳ convergence very slow

If α too large
↳ may overshoot the minimum

Q3.

① (a). RNN → RNN processes input sequentially it maintains a hidden state (memory) the output at step t depends both on current input & step $t-1$.

b). ANN → process inputs independently

②. Simple RNN's struggle with long term dependencies due to the problem of vanishing gradient.

(b). LSTMs uses gates to regulate information flow unlike RNN

Forget gate → Decides what irrelevant info to throw away

Input gate → Decides what new important info to store

Output gate → Decides what info is relevant of immediate prediction (ii). LSTM uses cell state to address the problem of vanishing gradient

Q3. • ANN → House price prediction
RNN → Time Series forecasting
LSTM → Machine Translation

Q4.

Example of long range dependency

(a). The boys who play soccar in the rain were tires

The model must connect the plural subject boys at the start the plural verb were at the end

A Standard RNN would struggle; by the time it processes the sentence the vanishing gradient problem would cause it to forget that the subject was "boys". It might incorrectly predict "is".

(b). LSTM use a memory cell that carries relevant context across time steps. Gates control info flow, input gate decides what new info to store, forget gate decides what old info to erase the output gate decide what to give as output. This gating prevents vanishing gradients & enables long term dependency learning