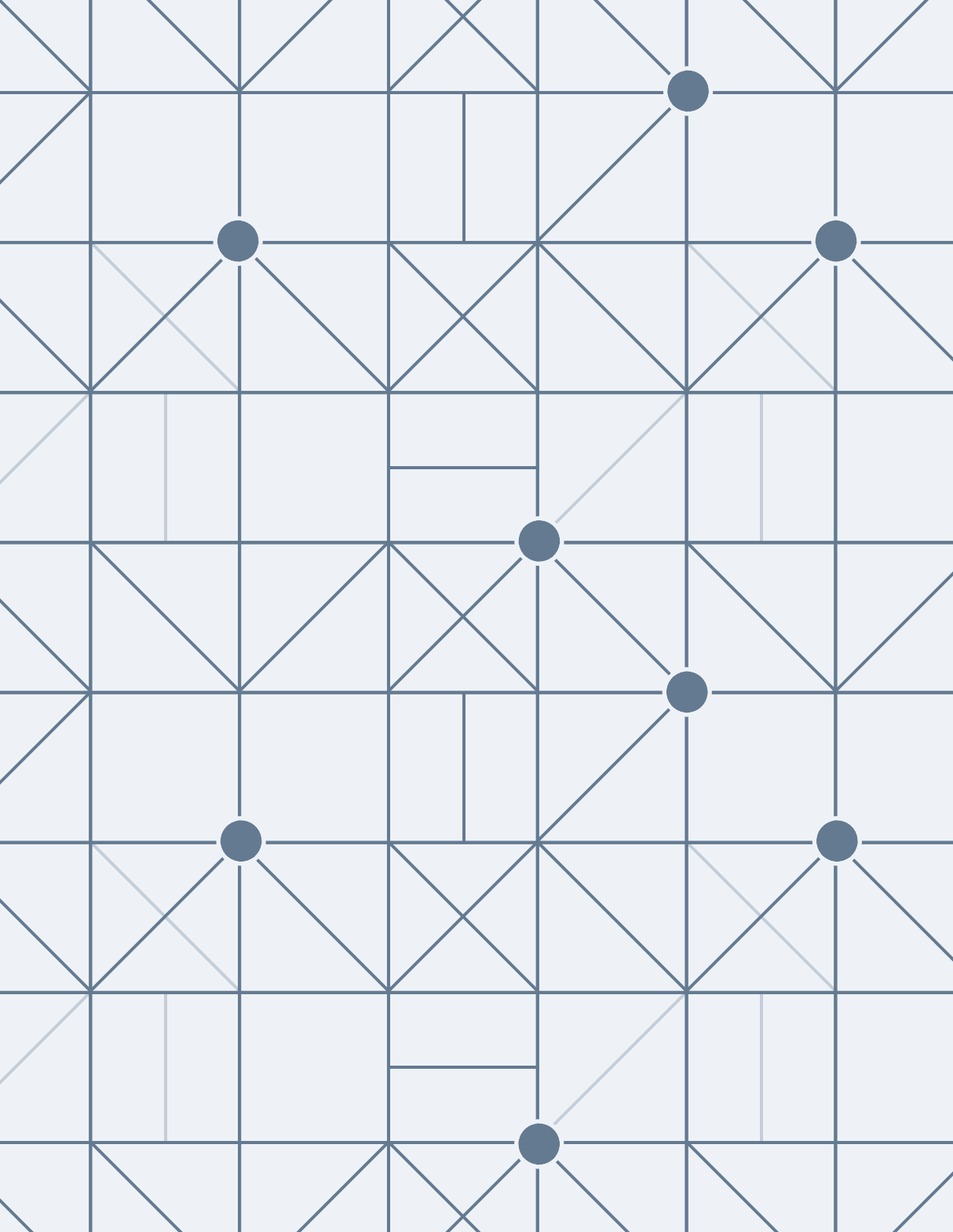

ASSIGNMENT - I

(a)

(i) The model is made to find $\hat{y}$ such that they are closest to true values, as squared errors make a much bigger difference in $|\hat{y} - y|$.

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

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

$$J(\beta) = \frac{1}{2}(y^Ty - y^TX\beta - \beta^TX^Ty + \beta^TX^TX\beta)$$

$$J(\beta) = \frac{1}{2}(y^Ty - 2\beta^TX^Ty + \beta^TX^TX\beta)$$

$$\nabla_\beta J(\beta) = \nabla_\beta \left[\frac{1}{2}y^Ty - \beta^T(X^Ty) + \frac{1}{2}\beta^T(X^TX)\beta\right]$$

$$\nabla_\beta J(\beta) = 0 - X^Ty + \frac{1}{2} \cdot 2(X^TX)\beta$$

$$\nabla_\beta J(\beta) = -X^Ty + X^TX\beta = 0$$

$$\therefore X^Ty = X^TX\beta$$

$$\hat{\beta} = (X^TX)^{-1}X^Ty$$

normal equation.

(c) Computational costs are very high.
If features are linearly dependent, $X^T X \to$ singular and a sol^n would not exist.

why iterative methods are preferred? →
- faster than inversion.

Q.2 Core idea :- to compute the gradient of loss function ($\mathcal{L}$) w.r.t. every weight & bias in a neural network. These help in optimisation to reduce the loss.

Chain Rule:- $\frac{\partial \mathcal{L}}{\partial w_{ij}} = \frac{\partial \mathcal{L}}{\partial \text{output}} \times \frac{\partial \text{output}}{\partial \text{hidden layer}} \times \dots \times \frac{\partial \text{local sum}}{\partial w_{ij}}$

re use of partial derivatives scales linearly with the no of layers & connections.

(b) $z_1 = w_1 x + b_1$ $\quad z_2 = w_2 a_1 + b_2$

$a_1 = \sigma(z_1)$ $\quad a_2 = \sigma(z_2) = \hat{y}$

$$\mathcal{L} = -(y \log a_2 + (1-y) \log(1-a_2))$$

$$\frac{\partial \mathcal{L}}{\partial z_2} = a_2 - y$$

$$\sigma(z) = \frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1 - \sigma(z))$$

$$\frac{\partial a_2}{\partial z_2} = a_2(1 - a_2)$$

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

$$z_2 = w_2 a_1 + b_2 \Rightarrow \frac{\partial z_2}{\partial w_2} = a_1$$

$$\frac{\partial L}{\partial w_2} = (a_2 - y) a_1 \quad \frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial z_2} \cdot \frac{\partial z_2}{\partial b_2}$$

$$z_2 = w_2 a_1 + b_2$$

$$\Rightarrow \frac{\partial z_2}{\partial b_2} = 1 \qquad \frac{\partial L}{\partial b_2} = a_2 - y$$

2. Propagated Error $:= \frac{\partial L}{\partial a_1}$

$$z_2 = w_2 a_1 + b_2 \quad \frac{\partial z_2}{\partial a_1} = w_2 \Rightarrow \frac{\partial L}{\partial a_1} = (a_2 - y) w_2$$

$$\frac{\partial a_1}{\partial z_1} = a_1(1 - a_1) = \sigma(z_1) \Rightarrow \frac{\partial L}{\partial z_1} = (a_2 - y) w_2 a_1 (1 - a_1)$$

$$\frac{\partial z_1}{\partial w_1} = x \quad \frac{\partial L}{\partial w_1} = (a_2 - y) w_2 a_1 (1 - a_1) x$$

$$\frac{\partial z_1}{\partial b_1} = 1 \quad \frac{\partial L}{\partial b_1} = (a_2 - y) w_2 a_1 (1 - a_1)$$

Q.3 (a) ANN input processing :-

→ x_{i+1} is independent of x_i

→ information flows in one dirⁿ.

RNN sequence processing :-

→ information flows in a loop, there is updation in hidden state for any input.

(b) During BPTT, the gradients are repeatedly multiplied by weight matrices of the recurrent layer. The gradient vanishes for weights connected to early time steps. These early weights receive minimal updates, thus the network is unable to learn or retain info from many steps back.

(c) Gates regulate flow of info into & out of the cell state

(d). LSTMs address it through cell state (CS).

(e) ANN : image classification.

RNN : speech recognition.

LSTM : Machine translation.

Q.4 (a) Example – subject-verb agreement.

→ vanishing gradient problem

(b) The memory cell & gating mechanism allow the LSTM to retain important info over long sequences by providing a path for flow.