Question - 4

Poisson regression

Q1) : ML estimation:

$$L = \prod_{i=}^{n} P(y_i / \theta_i) \longrightarrow \text{Likelihood.}$$

$\theta_i \rightarrow$ parameters for ith data point

Here, we try to fit data to a poisson distribution, with different parameters.

$$P(y_i/\theta_i) = P(y_i/\lambda_i) = \frac{e^{-\lambda_i}(\lambda_i)^{y_i}}{(y_i)!}$$

Here 'λ' $\rightarrow$ parameter for poisson

But $\lambda_i \rightarrow$ scalar for ith data point.

$$\underset{d\times 1}{X_i} \xrightarrow{\beta \rightarrow \text{LT}} \underset{1\times 1}{\lambda_i}$$

} d $\rightarrow$ dimensions / features of a data point.

$\Rightarrow$ We perform a linear transformation, & estimate λ_i for a given x_i

$$\lambda_i = \exp(X_i \beta)$$

$\beta \rightarrow D\times 1$ column vector

$X_i \rightarrow 1\times D$ row vector

Inorder to fit data, we need to find parameter λ_i of distribution

MLE :-

→ We use MLE to find λ_i (indirectly we need to find β)

$\beta \rightarrow$?

$$L = \prod_{i=1}^{N} \frac{e^{-\lambda_i} (\lambda_i)^{y_i}}{(y_i)!}, \quad \text{where } \lambda_i = \exp(X_i\beta)$$

→ Goal is to maximise 'L'

→ log-likelihood (it does not change the loss)

$$\log_e(L(\lambda)) = \sum_{i=1}^{N} -\lambda_i + y_i \log_e(\lambda_i) - \log(y_i)!$$

Need to maximise $\log_e(L(\lambda_i))$

$$\Rightarrow \log(L(\beta)) = \sum_{i=1}^{N} -\lambda_i + y_i \log(e^{X_i\beta}) - \log(y_i)!$$

$$= \sum -e^{(X_i\beta)} + y_i X_i \beta - \log(y_i)!$$

* Let $J(\beta)$ be loss function,

Aim is to minimise J, to find 'β'

$$\boxed{J(\beta) = -\log(L(\beta))}$$

$\rightarrow$ loss function

$\Rightarrow$ $$\boxed{J(\beta) = \sum_{i=1}^{N} e^{(x_i\beta)} - y_i(\bar{x}_i\beta) - \log(y_i)!}$$

$\rightarrow$ Find 'β' st $J(\beta)$ is min.

$$\Rightarrow \frac{\partial J(\beta)}{\partial \beta_k} = \sum_{i=1}^{N} e^{(x_i\beta)} \cdot x_i^k - y_i(x_i^k)) \quad : k \to k\text{th element in } \bar{\beta}$$

$$= \sum_{i=1}^{N} x_i^k (e^{(x_i\beta)} - y_i)$$

$$\frac{\partial J(\beta)}{\partial \beta} = X^T \cdot (e^{(x_i\beta)} - Y_i)$$

$$X = \begin{bmatrix} \bar{x}_1 \\ \vdots \\ \bar{x}_N \end{bmatrix}_{N\times D} \Rightarrow \bar{x}_1 = [x_1^1 \ldots x_1^D]_{D\times 1}$$

$$\beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_D \end{bmatrix}_{D\times 1}, \quad Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}_{N\times 1}$$

For minimising $J(\beta)$.

$$\Rightarrow \frac{\partial J(\beta)}{\partial \beta} = 0 \Rightarrow X^T (e^{(X\beta)} - Y) = 0$$

$$\Downarrow$$

Need to Solve a non-linear equ

→ Hence, we use gradient descent to solve it. Since no close form expression exists.

→ <u>Gradient Descent</u>:

$$\beta_{n+1} = \beta_n - \eta \left(\frac{\partial J}{\partial \beta} \right)$$

$$\beta_{n+1} = \beta_n - \eta \left(X^T \left(e^{X\beta_n} - y \right) \right)$$

$\eta \rightarrow$ learning rate.

→ when it converges

$$\lambda_i = \exp(x_i \beta)$$

$$\hat{y}_i = \hat{\lambda}_i$$

→ <u>Prediction</u>:

$$\boxed{\hat{y}_i = \exp(x_i \beta)}$$

4.4 Regularisation:

⇒ As ~~expla~~ derived in 4.1:

cost func / loss fn (No regularisation).

$$J(\beta) = \sum_{i=1}^{N} e^{(x_i \beta)} - y_i(\bar{x}_i \bar{\beta}) - \log(y_i!)$$

$$\Rightarrow \beta_{n+1} = \beta_n - \eta\left(\frac{\partial J}{\partial \beta}\right)$$

For L_1 regularisation

* hyperparameter : λ

$M \rightarrow$ no. of features

$$J(\beta) = \sum_{i=1}^{N} e^{x_i \beta} - y_i(\bar{x}_i \bar{\beta}) - \log(y_i)! + \lambda \sum_{j=1}^{M} |\beta_j|$$

$$\therefore \underbrace{\left.\frac{\partial J}{\partial \beta}\right|}_{L_1\text{-regularised}} = \underbrace{\left.\frac{\partial J}{\partial \beta}\right|}_{\text{non regularised}} + \lambda \begin{bmatrix} 1 \\ \vdots \\ 1 \\ \vdots \end{bmatrix}_{M\times 1}$$

⇒ Hence: for L_1-regularisation

$$\beta_{n+1} = \beta_n - \eta \underbrace{\left.\left(\frac{\partial J}{\partial \beta}\right)\right|}_{\text{non-reg}} - \lambda' \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}_{M\times 1}$$

Use of L1-regularisation

↳ This tries to shrink the ~~features~~ weights of unimportant features.

⇒ Helping us to know "useful" features.

For L_2 regularisation

hyperparameter : λ

$$\Rightarrow J(\beta) = \sum_{i=1}^{N} e^{x_i\beta} - y_i(\bar{x}_i\bar{\beta}) - \log(y_i)! + \frac{\lambda}{2}\sum_{j=1}^{M} |\beta_j|^2$$

$$\Rightarrow \left.\frac{\partial J}{\partial \beta}\right|_{L_2\text{-reg.}} = \left.\frac{\partial J}{\partial \beta}\right|_{\substack{\text{non-}\\ \text{reg}}} + \lambda(\beta)$$

$$\Rightarrow \beta_{n+1} = \beta_n - \eta\left(\left.\frac{\partial J}{\partial \beta}\right|_{\substack{\text{non-}\\ \text{reg}}}\right) - \lambda'(\beta_n)$$

$$\boxed{\beta_{n+1} = \beta_n(1-\lambda) - \eta\left(\frac{\partial J}{\partial \beta}\right)}$$

⇒ Used for avoiding over-fitting.