ML Assignment - 1

Q.1 $y = ax + b$ derive formulas for a & b that minimizes the least square error.

Ans Given that the input is one dimensional assume that we have n input points such as

$$x = \{x_1, x_2, x_3, x_4 \cdots x_n\}$$

Now to predict points $Y = \{y_1, y_2, y_3 \cdots y_n\}$ we will use the formula $y = ax + b$.
Lets assume that we have the actual class values as $Z = \{z_1, z_2, z_3 \cdots z_n\}$
The mean square error will be calculated as follows

$$MSE = \frac{\sum_{i=1}^{n} (z_i - y_i)^2}{n}$$

substituting $y = ax + b$ we get

$$MSE = \frac{\sum_{i=1}^{n} (z_i - (ax_i + b))^2}{n} \quad -(1)$$

To minimize this error we need to mimize the equation (1)

(P.T.O)

Since n is a constant we have to minimize the following

$$\sum_{i=1}^{n} (z_i - (ax_i + b))^2$$

On Expanding the equation we get

$$MSE = \frac{1}{n}\left[(z_1 - (ax_1 + b))^2 + (z_2 - ((ax_2 + b))^2) + (z_3 - (ax_3 + b))^2 + (z_4 - (ax_4 + b))^2 \ldots .\right]$$

$$MSE = 1/n\left[(z_1^2 - 2z_1(ax_1 + b) + (ax_1 + b)^2) + (z_2^2 - 2z_2(ax_2 + b) + (ax_2 + b)^2 + (z_3^2 - 2z_3(ax_3 + b) + (ax_3 + b)^2) + (z_4^2 - 2z_4(ax_4 + b) + (ax_4 + b) + \ldots + \ldots + \ldots\right]$$

On grouping the similar terms together we get

$$MSE = 1/n\left[(z_1^2 + z_2^2 + \ldots z_n^2) - 2a(z_1x_1 + \ldots + z_nx_n) - 2b(z_1 + z_2 + z_3 \ldots + z_n) + a^2(x_1^2 + x_2^2 + \ldots x_n^2) + 2ab(x_1 + x_2 + x_3 \ldots x_n) + nb^2\right]$$

Lets say

$$\hat{z^2} = z_1^2 + z_2^2 + z_3^2 + \dots z_n^2 / n$$

$$\hat{z} = z_1 + z_2 + \dots z_n / n$$

$$\hat{zx} = z_1 x_1 + z_2 x_2 + z_3 x_2 \dots z_n x_n / n$$

$$\hat{x^2} = x_1^2 + x_2^2 \dots x_n^2 / n$$

$$\hat{x} = x_1 + x_2 + x_3 \dots x_n / n.$$

Now we get the MSE equation as follows

(2) $\longrightarrow MSE = \hat{z^2} - 2a\hat{zx} - 2b\hat{z} + a^2\hat{x^2} + 2ab\hat{x} + b^2$

To minimize this equation we will use partial derivative with respect to a and with respect to b. We will set the partial derivative equal to 0.

$$\therefore \frac{\partial(MSE)}{\partial(a)} = 0 \qquad \frac{\partial(MSE)}{\partial(b)} = 0.$$

$$\frac{\partial (MSE)}{\partial (a)} = -2\hat{z}x + 2a\hat{x}^2 + 2b\hat{x} \quad - (3)$$

Divid with 2 we get $-\hat{z}x + a\hat{x}^2 + b\hat{x} = 0 \quad - (4)$

$$\frac{\partial (MSE)}{\partial (b)} = -2\hat{z} + 2a\hat{x} + 2b \quad - (5)$$

Divide with 2 we get

$$-\hat{z} + a\hat{x} + b = 0 \quad - (6)$$

For equat H we shift the z term to right & we get

$$a\hat{x}^2 + b\hat{x} = \hat{z}x$$

dividing by $\hat{x}$

$$\frac{a\hat{x}^2}{\hat{x}} + b = \frac{\hat{z}x}{\hat{x}} \quad - (7)$$

Similary for equation (6) we get

$$a\hat{x} + b = \hat{z} \quad - (8)$$

Subtracting (7) from (8)

$$a\left(\hat{x} - \frac{\hat{x}^2}{\hat{x}}\right) = \hat{z} - \frac{\hat{z}x}{\hat{x}}$$

$$\therefore a = \frac{\left(\hat{z} - \frac{\hat{zx}}{\hat{x}}\right)}{\left(\hat{x} - \frac{\hat{x}^2}{\hat{x}}\right)}$$

Using this equation/value of a to find value of b we get

$$b = \hat{z} - \frac{\left(\hat{z} - \frac{\hat{zx}}{\hat{x}}\right)}{\left(\hat{x} - \hat{x}^2/\hat{x}\right)}$$

$$\therefore a = \frac{\hat{z}\hat{x} - \hat{zx}}{(\hat{x})^2 - \hat{x}^2} \quad \text{—— (9)}$$

$$b = \hat{z} - \hat{x}\left(\frac{\hat{z}\hat{x} - \hat{zx}}{(\hat{x})^2 - \hat{x}^2}\right) \quad \text{—— (10)}$$

These values of a and b will minimize the mean square error.

Q.2 Regularized Linear Regression

(a) What will happen if we overestimate the value of λ for lassa? what would happen to the number of non-zero elements of w.

Ans. The sparsity in weights will increase if λ is overestimated in LASSO. The formula in LASSO is as follows

$$\lambda||\beta|| - (1)$$

Now in equation since λ is multiplied by $||\beta||$ and since the algorithm tries to decrease or minimize the error, it will lead to many β values zero and this will lead to omission of many non-zero elements Although this helps in decreasing the model complexity, it will lead to underfitting

(b) What if underestimate the value of λ for RIDGE. What happens to value of w?

Answer) The formula in RIDGE is given as

$$\lambda||\beta||_2^2$$

which is

$$\lambda\left(\beta_1^2 + \beta_2^2 + \beta_3^2 + \cdots \beta_n^2\right)$$

(2b) (Continued)

Now if we underestimate the value of λ then the term λ will be negligible and it will give the same equation as linear regression using OLS. Since OLS reduces the bias, it will be beneficial but it will lead to overfitting.

(c) Calculate the partial derivative of penalty terms for LASSO and RIDGE.

Ans) LASSO:-

$$J(w) = \frac{1}{N} \sum_{i=1}^{N} (y_i - (w_0 + w^T x_i))^2 + \lambda ||w||_1$$

$$\text{Penalty term} = \lambda ||w||_1 = \lambda \sum_{i=1}^{d} |w_i|$$

$$\frac{\partial (\text{Penalty})}{\partial w} = \begin{cases} \lambda & , \text{if } w_i > 0 \\ -\lambda & , \text{if } w_i < 0. \end{cases}$$

Ridge:-

$$\text{Penalty} = \lambda ||w||_2^2$$

$$\frac{\partial (\text{Penalty})}{\partial w} = 2\lambda w_i$$

(d) (i) For what ~~for what~~ values of w_i will their behaviours differ.

Ans The lasso coefficients will be reduced by constant factor. Their values will be lower than ridge coefficients and will become zero faster.

(ii) The formula for penalty in ridge is

$$\lambda ||w||_2^2$$

which means

$$\lambda \cdot |w_i^2|$$

The formula for penalty term in Lasso is

$$\lambda ||w||_1$$

which means $\lambda \cdot w_i$

(iii) From the above formulas it can be seen that with negative w_i ($-w_i$) the ridge ~~factor~~ will square and the effect of negative sign won't be there but in lasso the negative term will affect the penalty terms value and in term affect the values of β.

(iv) In terms of partial derivative the w_i value won't effect LASSO as the partial derivative ~~does not have~~ is 1 or -1

(P.T.O)

(iv) For negative or positive values of w_i the partial derivative term in lasso will give λ or $-\lambda$ but in ridge it will reduce or increase the penalty term by 2. ~~For value~~

(v) For $w_i = 0$ the penalty term will be 0 for both, the ridge coefficient will be reduced factor of linear regression.

Q.3a Write down the equation for weight update step.

$$\omega = \omega + \alpha \frac{\partial}{\partial \omega} (l(\omega))$$

'+' indicates maximizing Likelihood.

$$\frac{\partial (l(\omega))}{\partial \omega_j} = \left(y \frac{1}{sigm(\omega^T x)} - (1-y) \frac{1}{1-sigm(\omega^T x)} \right) \frac{\partial\, sigm(\omega^T x)}{\partial \omega_j}$$

$$= \left(y \cdot \frac{1}{sigm(\omega^T x)} - (1-y) \frac{1}{1-sigm(\omega^T x)} \right) g(\omega^T x)$$

$$\times (1 - sigm(\omega^T x)) \frac{\partial}{\partial \omega} (\omega^T x)$$

$$= (y (1 - h_\omega(x)) - (1-y) h_\omega(x)) x_j$$

$$= (y - h_\omega(x)) x_j$$

$\therefore$ Batch gradient step

$$\theta_j = \theta_j + \alpha \sum_{i=1}^{M} (y_i - sigm(\omega^T x)) x_j^i$$

(3b) Does it always converge? What is a suitable stopping condition?

Ans) (i) A good stopping condition can be to check if the difference between the updated weights and previous weights is negligible

$$\text{i.e } |w' - w| < \varepsilon \qquad 0 < \varepsilon < 1$$

(ii) Another good stopping condition would be a fixed number of iteration. Study shows that ~~iteration~~ between 5000 to 10000 iteration is a good stopping value.

(iii) If the learning rate α is not set properly, it will never converge.

(iv) Line search algorithm is good solution for finding a good α rate which will give a better solution.

Q.5a. Prove that for any arbitary tree with unequal branching nodes, there exists a binary tree that implements same classification functionality.

Ans) In this proof we will only be interested in trees with nodes who have children or subnodes greater than 2.

(i) We start ~~by~~ from the root node and traverse the tree to find node with children greater than 2

(ii) If we dont find such a node the tree is a binary tree.

(iii) If we find a tree with children greater than 2, say n children we will ~~keep~~ ~~the~~ build the tree as follows

(a) Keep the old node as it is

(b) Keep the leftmost child of the old node as ~~the~~ it is.

(c) Create a new right node and assign ~~its~~ n-1 children of the old node as children of the new right node.

(d) Do this recursively till all nodes ~~are~~ have children less than or equal to two.

(iv) Thus we have converted to functionaly equivalent binary tree.

5(b) Consider a tree with just 2 levels, a root node connected to B leaf nodes (B>=2) what are upper & lower limits on functionally equivalent binary tree as function of B

Ans The maximum number of leaf nodes in a binary tree of depth d is equivalent to

$$2^d.$$

∴ Since the given tree which has B leaf nodes should be

$$B \leq 2^d$$

Taking log we get

$$\log_2 B \leq d$$

To this we add 1 for the root node

$$\therefore d > \log_2 B + 1 - (\text{Minimum Value}$$

The upper limit for binary tree ~~width~~ of depth d will be

$$d \leq B.$$

(5c) For binary tree with B terminal nodes, the number of nodes will be 2B-1.

Since $B \geq 2$ the minimum number of nodes will be 3.

Now for a tree with B+1 ~~nodes~~ terminal nodes ~~the~~ with depth d. To convert this to binary tree we need to have 2B-1 nodes. To conver we take n-1 right children, add a new node and make the n-1 ~~children~~ children of the new node so for example if we have 2 level tree with 1 ~~st~~ extra node we get binary tree as

~~2(B)~~ 2B-1+2 = 0

∴ For B+1 the maximum number of nodes should be

$2(B+1)-1$.

∴ It will be $2B+1$.