2021

DATE:

Assignment - 1

Q1)

(a) Linear Regression is trying to fit a line between the features and output (variable we are predicting). To find the line equation which fits best to the data, we assume that there is a linear relationship between our features and our output. We are going to find the line equation ensuring that our error is minimized. We choose a line as our best fit line which gives the minimum error. Minimizing square error is a meaningful choice - because while calculating the error - it can be positive or even negative, to avoid the negatives which may reduce our error a lot and can give wrong predictions, we square the error. By minimizing the squared error, we can easily find the regression coefficients and also to avoid outliers.

2021

DATE:

(b)

$$\hat{y} = X\hat{\beta}$$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}_{n \times 1} \qquad X = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1d} \\ 1 & x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & & & & \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}_{n \times d+1}$$

$$\hat{\beta} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \hat{\beta}_2 \\ \vdots \\ \hat{\beta}_d \end{bmatrix}_{d+1 \times 1} \qquad J(\hat{\beta}) = \frac{1}{2} \| y - X\hat{\beta} \|^2$$

$$\| y - X\hat{\beta} \|^2 = [y - X\hat{\beta}]^T [y - X\hat{\beta}]$$

$$[y - X\hat{\beta}]^T [y - X\hat{\beta}] = y^T y - y^T X\hat{\beta} - \hat{\beta}^T X^T y + \hat{\beta}^T X^T X \hat{\beta}$$

Scalar Scalar Scalar Scalar

$(\text{Scalar})^T = \text{Scalar}$

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

$$\hat{\beta}^T X^T y = y^T X \hat{\beta}$$

$$= y^T y - 2 y^T X\hat{\beta} + \hat{\beta}^T X^T X \hat{\beta}$$

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

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

$$\downarrow \qquad \downarrow$$

$$0 - 2X^T y + 2X^T X\hat{\beta}$$

2021

DATE:

$$\frac{\partial J(\hat{\beta})}{\partial \hat{\beta}} = \frac{1}{\cancel{2}}\left(0 - \cancel{2}(X^T y) + \cancel{2} X^T X \hat{\beta}\right)$$

$$\frac{\partial (J(\hat{\beta}))}{\partial \hat{\beta}} = 0$$

$$-X^T y + X^T X \hat{\beta} = 0$$

$$X^T X \hat{\beta} = X^T y$$

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

$$\boxed{\hat{\beta} = (X^T X)^{-1} X^T y} \rightarrow \text{Normal Equation}$$

(c) In high-dimensional or large-scale datasets calculating the inverse of $X^T X$ requires high computational power and often leads to a slower model where as iterative methods (such as gradient descent) allows us to calculate it more effiencently and also sometimes taking the derivatives of larg functions ~~Can be difficult~~ and equating them to zero to find solution is diffi

2021

DATE:

Q2)

(a) Core idea behind backpropagation is to minimize the error calculated by the Loss function. The Loss function calculates the difference between the ~~add~~ observed and the predicted values. Back propagation helps us to find the parameters which will provide the best fit curve. It finds these parameters by using the method of chain rule and optimizing those by gradient descent. The Chain rule enables us to efficiently calculate the derivates with the parameters to be found in the function. Chain rule breaks down the derivaties of the function layer by layer backward making it computationaly efficient to calculate the derivaties.

(b)

2021

DATE:

(b)

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial a_2} * \frac{\partial a_2}{\partial z_2} * \frac{\partial z_2}{\partial w_2}$$

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

$$= (a_2 - y) * a_1$$

$$\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial a_2} * \frac{\partial a_2}{\partial z_2} * \frac{\partial z_2}{\partial b_2}$$

$$= \frac{\partial L}{\partial z_2} * \frac{\partial z_2}{\partial b_2} = (a_2 - y) * (1)$$

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial a_2} * \frac{\partial a_2}{\partial z_2} * \frac{\partial z_2}{a_1} * \frac{\partial a_1}{\partial z_1} * \frac{\partial z_1}{\partial w_1}$$

$$= \frac{\partial L}{\partial z_2} * \frac{\partial z_2}{\partial a_1} * \frac{\partial a_1}{\partial z_1} * \frac{\partial z_1}{\partial w_1}$$

$$= (a_2 - y) * w_2 * a_1(1 - a_1) x$$

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial a_2} * \frac{\partial a_2}{\partial z_2} * \frac{\partial z_2}{\partial a_1} * \frac{\partial a_1}{\partial z_1} * \frac{\partial z_1}{\partial b_1}$$

$$= (a_2 - y) * w_2 * a_1(1 - a_1) * 1$$

2021

DATE:

(c) By Using gradient descent:

$$w_1 = w_1 - \alpha \frac{\partial L}{\partial w_1} \qquad b_2 = b_2 - \alpha \frac{\partial L}{\partial b_2}$$

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

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

α - Learning rate

Learning rate determines the stepsize which is taken to update our parameter to its minima. If it is small, the step size is going to be smaller and would reach the minima very slowly. If it is too large, then it may diverge from the minima

2021

DATE:

Q3) (a) ANN takes one input and process it independently. It has no memory, so it is not suitable for sequential data where as RNN's are suitable for sequential data. They store the previous inputs in hidden state which are used with the current input to predict the outcome.

(b) RNN's struggle with long-term dependencies because RNN's backpropagate through time and also store the previous inputs. While using the chain rule, the derivative for the long term memory becomes large which might vanish or explode (depending on the weight's value vanish <1 explode >1)

This causes the neural network to loose the long ~~Memory~~ term memory.

2021

DATE:

(c) The gates control the information flow. They actually decide whether the memory is relevant or not, control what memory to add, store in the process of updating the memory.

Forget gate : Tells us how much of the long term memory is relevant to pass

Input gate : Controls how much of the new input is to be stored.

Output gate : Controls how much of the memory to be given as output.

(d) The LSTM solves the vanishing gradient problem by setting the forget gate to close to 1 to preserve the long term memory and would set the input gate to close to 0 to Prevent the long term memory to vanish.

2021

DATE:

(e) ANN → Image Classification
RNN → Stock-price prediction / Forecasting
LSTM → Text to speech → long range ~~dependices~~ dependencies

4) (a) "I asked my sister, after returning ~~from~~ from the gym, if she wanted ~~the~~ to cook ~~for~~ with me."

To understand the word 'she', the model must remember 'my sister'.
Yes, a standard RNN would struggle as it gets vanished and will forget the word 'my sister'.

(b)

2021

DATE:

(6)

The memory cell c_t and $\tilde{c}_t$ stores the memory where as the input, forget and output gates filter out the memory and check whether if the previous information is relevant or irrelevant to remember.

In the previous example that I have given -

" I asked my sister, after returning from the gym, if she wanted to cook with me "

Here f_t for the phrase " after returning from the gym " would be set to zero because it is irrelevant for the model to remember.