Notes on Algorithm of Machine learning (Linear regression).

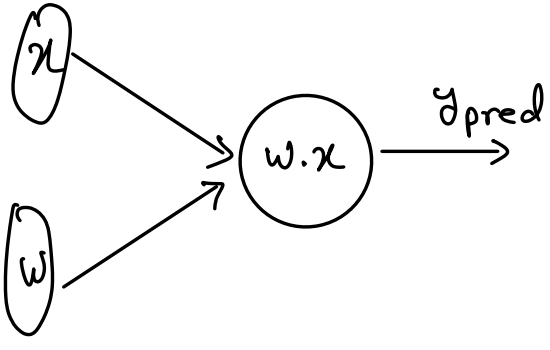
Algorithm;

Forward pass; Compute prediction

Ex; input data; x

Assign 'w' (weight) for the model that encodes the learning part.

Prediction; $y_{pred} = w.x$



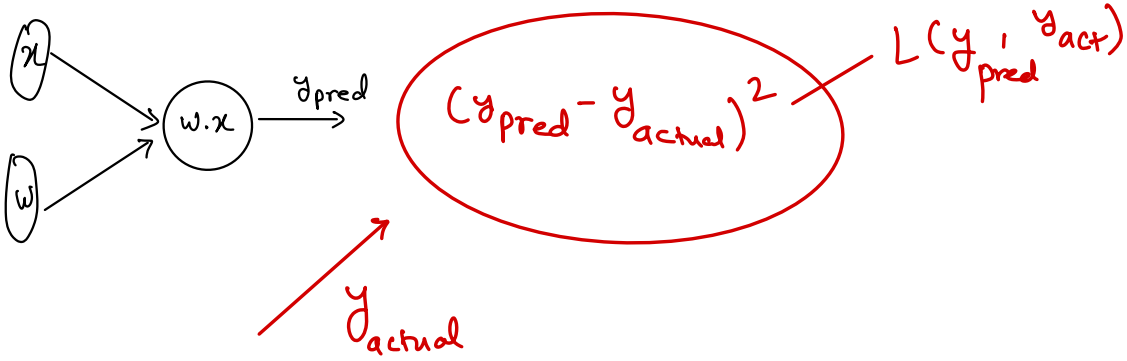
Python script: Define a function that does this job;

```
def forward (w):
    return w.x
```

Loss Calculation;

Take y_{pred} from forward pass as the input and check how far is this value compared to actual result;

Lets actual result be given by; y_actual

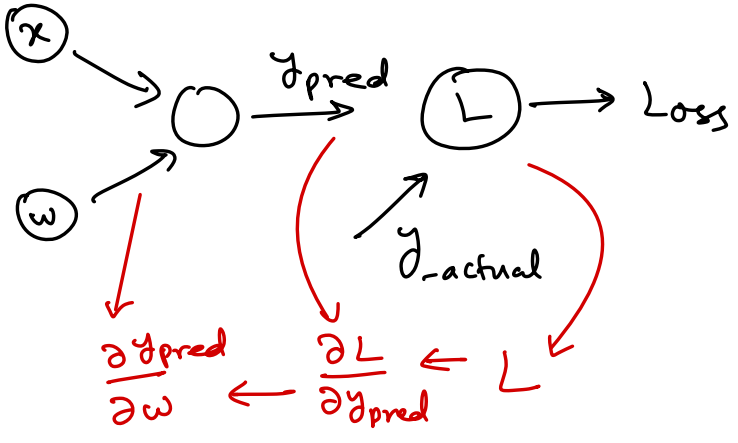



```
def L(ypred, yactual):
    return ((ypred - yactual)**2).mean
```

$$Loss = \frac{1}{n} \sum_{i=1}^{n} (y_{pred} - y_{actual})^2$$ #

Back propagation ;

This is the process to update the model's weights based on computed loss. It is done by calculating the gradients & using these gradient to adjust the parameters in the dir that reduces the loss



$$\frac{dL}{dw} = \frac{\partial L}{\partial y_{pred}} \cdot \frac{\partial y_{pred}}{\partial w}$$

In our case, $L = (y_{actual} - y_{pred})^2 = (wx - y_{pred})^2$

$$\frac{\partial L}{\partial y_{pred}} = 2(wx - y_{pred}) \cdot 1, \quad \frac{\partial y_{pred}}{\partial w} = \frac{\partial (wx)}{\partial w} = x$$

$$\frac{\partial L}{\partial w} = 2x\,(wx - y\_pred)$$

$$\boxed{\frac{\partial L}{\partial w} = 2x \cdot (y\_actual - y\_pred)}$$

Optimization (weight update using gradient);

This is done to update the parameter such that the loss function is reducing to minimum;

It is done by gradient descent; where each parameter is updated by substracting small portion of gradient controlled by learning rate 'η'.

$$w = w - \eta \cdot \frac{dL}{dw}$$

In script;

$$w\_ = learning\_rate \times gradient.$$