Deep Neural Networks

Jawwad Ahmed Shamsi

1

Forward Propagation

2

Activation Functions (studied so far)

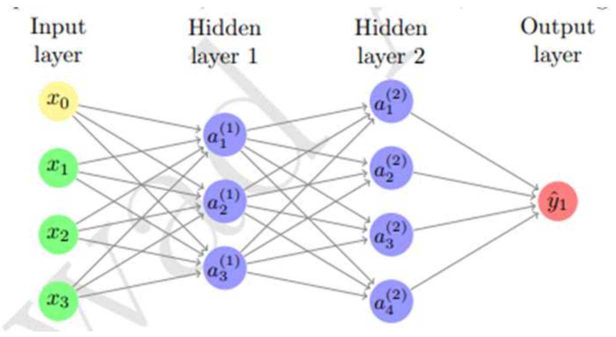
Sigmoid

- Binary Classification
- Outputs Probabilities 0 ->1

Softmax

- Multi-classification
- Sum of all probabilities is 1

3

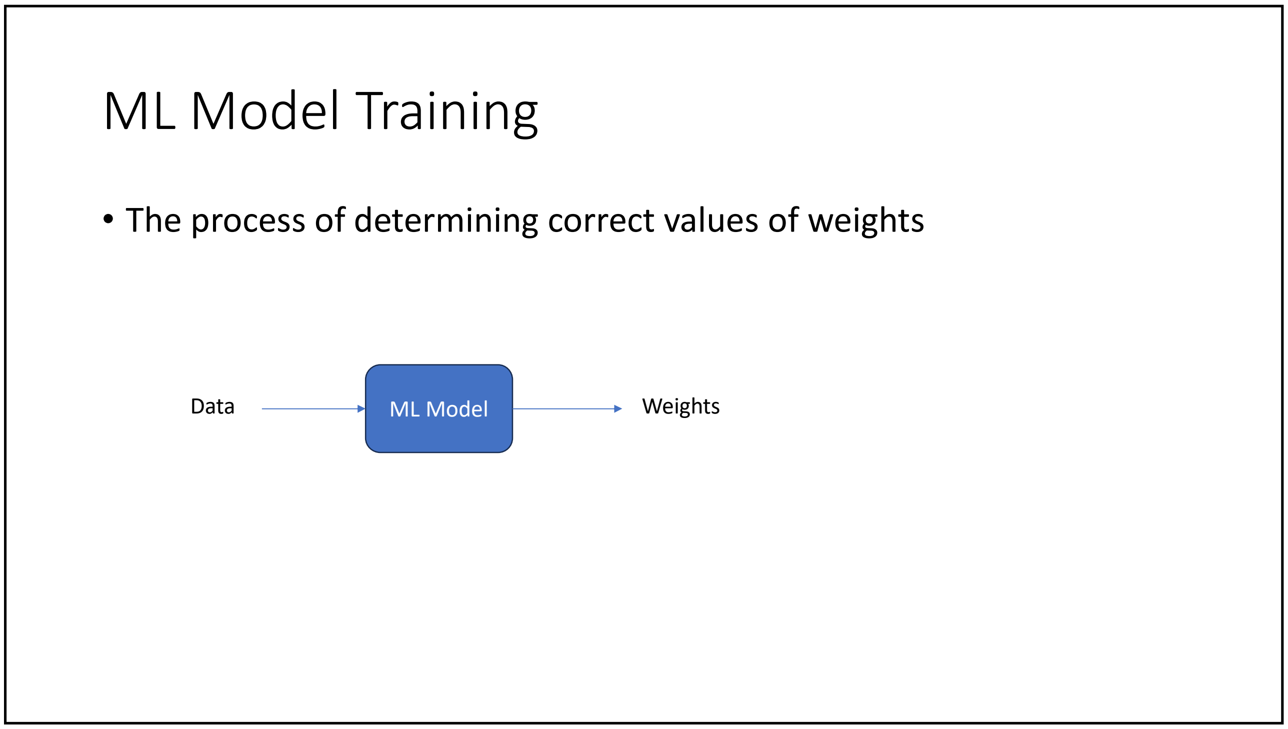


$$a_1^1 = \sigma(w_{11}^1.x_1 + w_{12}^1.x_2 + w_{13}^1.x_3)$$
$$a_2^1 = \sigma(w_{21}^1.x_1 + w_{22}^1.x_2 + w_{23}^1.x_3)$$
$$a_3^1 = \sigma(w_{31}^1.x_1 + w_{32}^1.x_2 + w_{33}^1.x_3)$$

where w_{jk}^i denotes weight b/w the jth node at the ith layer and kth node at the i-ith layer

$$\hat{y} = \sigma(w_{11}^3 a_1^2 + w_{12}^3 a_2^2 + w_{13}^3 a_3^2 + w_{14}^3 a_4^2)$$

4

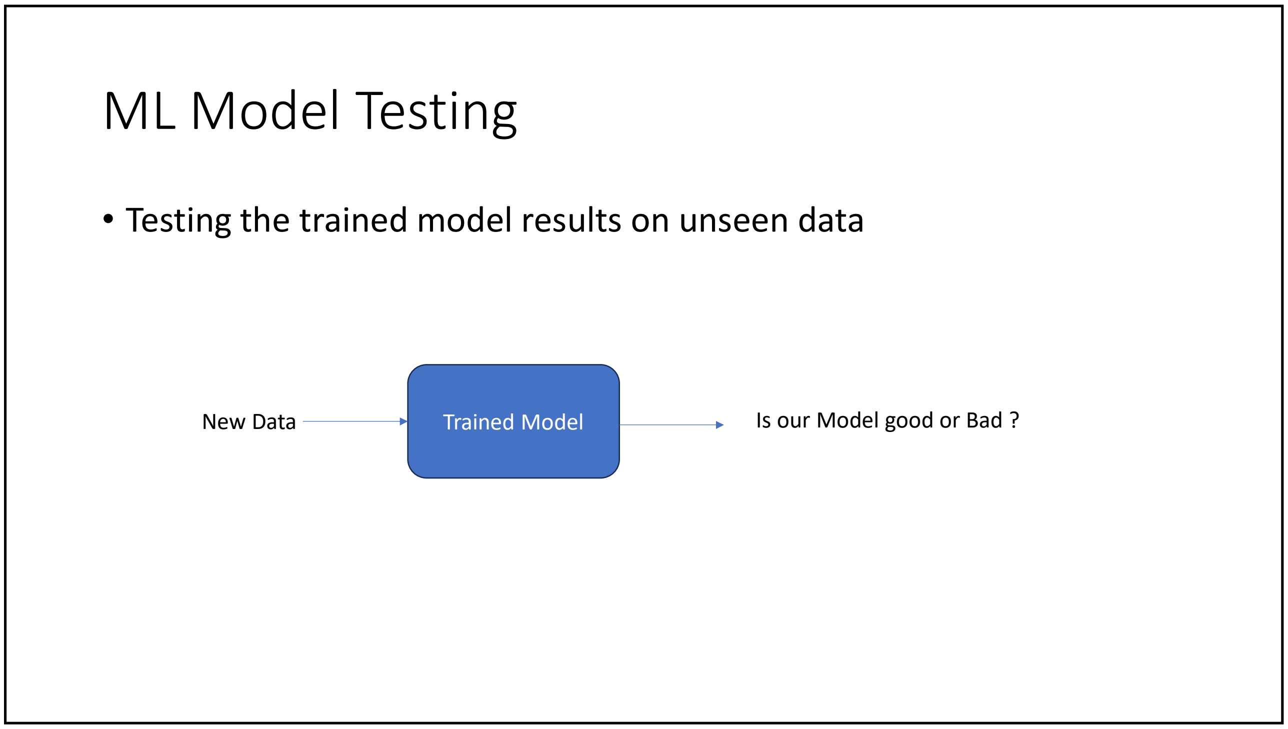


ML Model Training

- The process of determining correct values of weights

$$\begin{tikzcd} \text{Data} \arrow[r] & \boxed{\text{ML Model}} \arrow[r] & \text{Weights} \end{tikzcd}$$

5



ML Model Testing

- Testing the trained model results on unseen data

$$\begin{tikzcd} \text{New Data} \arrow[r] & \boxed{\text{Trained Model}} \arrow[r] & \text{Is our Model good or Bad ?} \end{tikzcd}$$

6

How do we check if the Model is good?

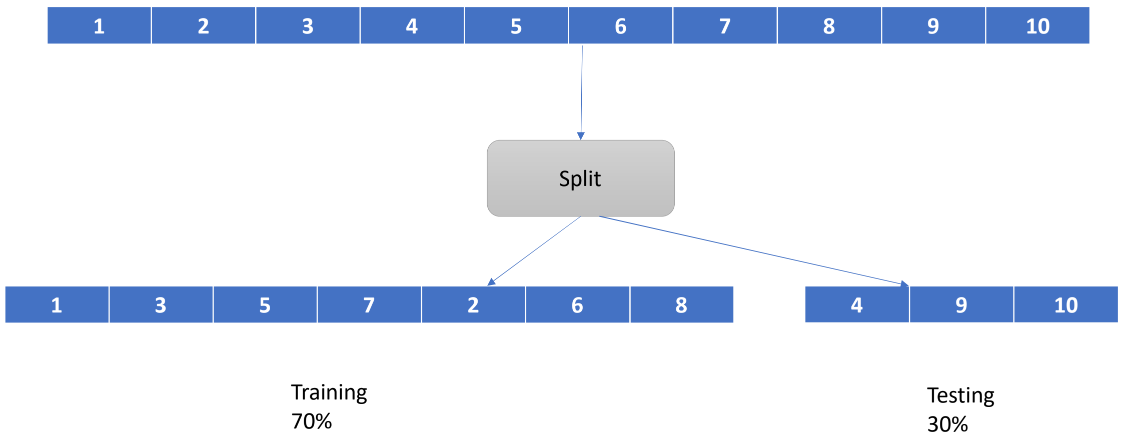
7

Model Evaluation

- Accuracy
- Precision
- Recall

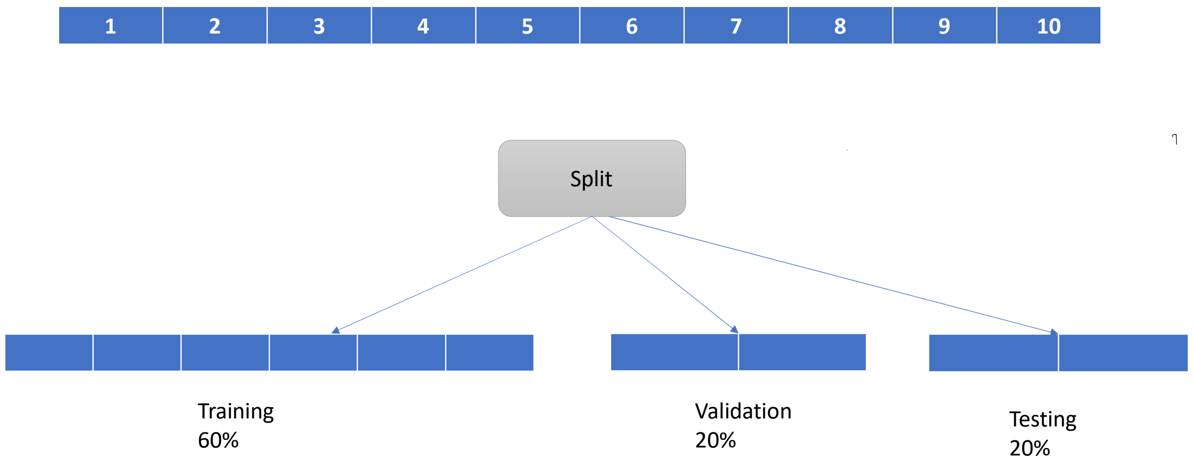
8

Data Split



9

Data Split (2)



10

Cost Function

11

Cost Function

Difference b/w Estimated and Actual $J(w) = |\hat{y} - y|$

J(w)

Function of Weights

Goal is to minimize

12

Types of Error Functions

- Continuous Values (Regression)

Mean Square Error $E(W,b) = \frac{1}{n}\Sigma_{i=1}^{n}(\hat{y} - y_i)^2$
MSE penalizes outliers
Or Absolute Mean Error $E(W,b) = \frac{1}{n}\sum_{t=1}^{n}|\hat{y} - y_i|$

- Categorical Values (Cross Entropy)

$E(W,b) = -\Sigma_{i=1}^{m}\hat{y}_i log(p_i)$
cross-entropy over all training examples can be computed as follows:
$E(W,b) = -\Sigma_{i=1}^{n}\Sigma_{i=1}^{m}\hat{y}_i log(p_i)$
Here is the target probability (p) is the predicted probability and (m) is the number of classes.

13

Example

- For instance
  - Elephant,
  - Lion,
  - and Giraffe.
  - Suppose the training probability distribution is as follows: Class A: Elephant 0.0 Class B: Lion 1.0 Class C: Giraffe 0.0
  - This implies that the particular training instance has 100% probability of Class B (Lion) and 0 % probability of other two classes.

14

Example contd. (Testing)

- Now suppose during testing our algorithm predicts the following distribution:
- Class A: Elephant 0.3
- Class B: Lion 0.2
- Class C: Giraffe 0.5
- E= -(0.0 * log (.3) + 1.0 * log(0.2)+ 0.0 *log (0.5))= -1.609
- Let's compute error for better predictions
- Class A: Elephant 0.2Class B: Lion 0.6Class C: Giraffe 0.2

15

Back Propagation

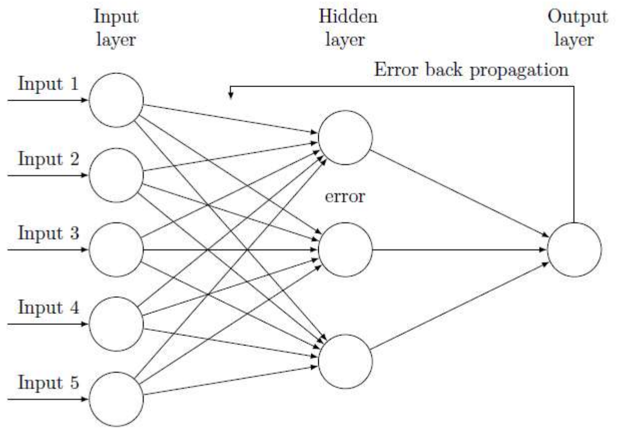
16

Back Propagation

- Moving backwards
  - from output to input
- Adjusting weights
- Minimize error

17

Back Propagation

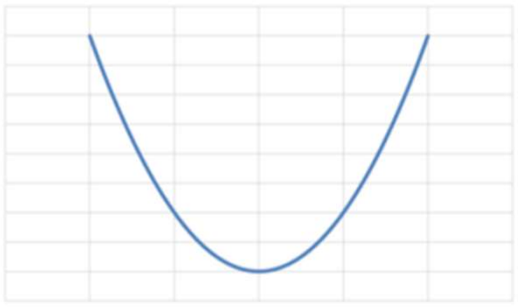


18

Some Basics

19

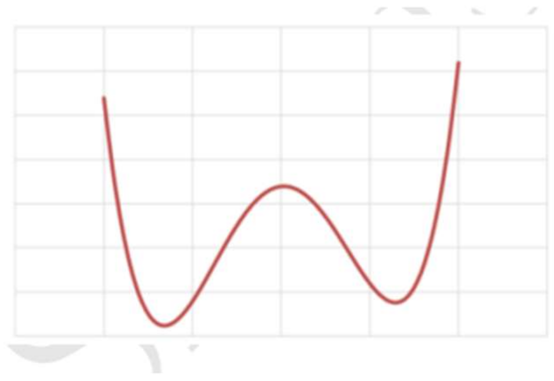
Convex Function



- Global Minima

20

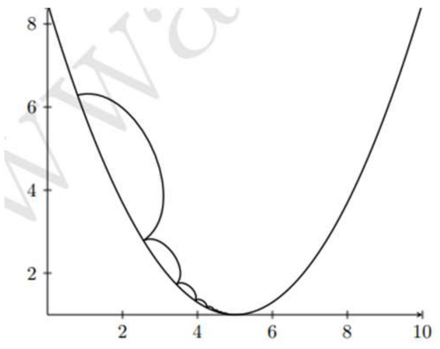
Non-Convex Function



- More than one local minima

21

Finding Minima



- Need to find minima
- Do we have a global minima?

Or

Do we have a few local minima?

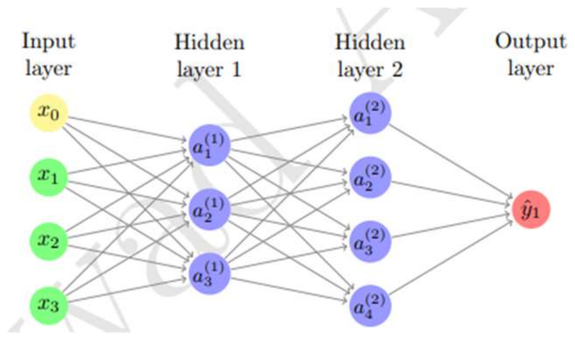
22

Hyper Parameter vs Parameters

23

Brute Force Approach

- Try all possible values of weights



24

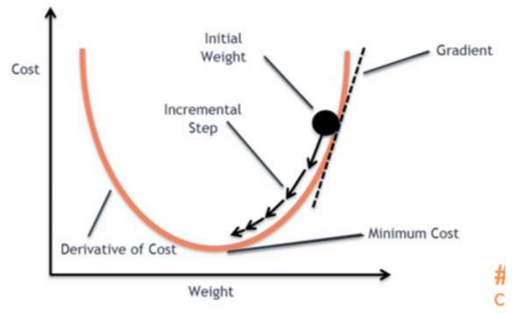
Gradient Descent

25

Gradient Descent

- Select initial weights
- Compute slope of the error function
- choose the steepest slope.
- Choose the descent direction (adjust weights) opposite to the direction of the slope
- the next value of weight is computed using the product of slope and the learning rate, where the learning rate is the magnitude of adjustment.
- iterate until convergence

26



27

```
for $k = 0, 1, 2, \ldots$ do
    $g_k \leftarrow \nabla f(x_k)$
    $x_{k+1} \leftarrow x_k - t_k g_k$
end for
```

$$\Delta W_i = -\alpha \frac{\partial E}{\partial W_i}$$

In the above equation, α gives us the amount of step size; whereas,

$$\frac{\partial E}{\partial W_i}$$

28

Analyzing Gradient descent

- Compute gradient descent over all the points in a dataset. That is, update weights after completing the iteration over the whole batch. For instance, if the training dataset contains 1000 data examples, then weights are adjusted after completing the whole iteration over 1000 data points.

29

Gradient Descent

- Pros
  - Stable descent
  - Less Oscillations
- Cons
  - May get stuck in local minima
  - Extremely slow

30

Stochastic Gradient Descent Algorithm

Select a random point and compute gradient w.r.t that point.

Pros:

- Faster than Batch Gradient Descent
- Can converge faster
- Likely to find error which is near to the global minima

- Cons
  - Higher Oscillations
  - Faster than gradient descent but may not reach global minima

31

Mini Batch Gradient

- Divide training into mini batches and select random training examples from these mini batches.

32