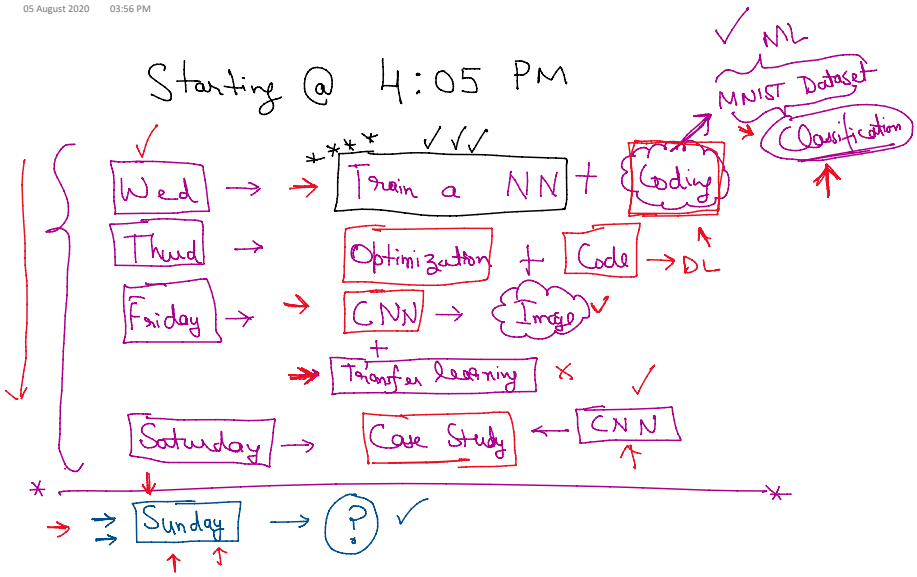
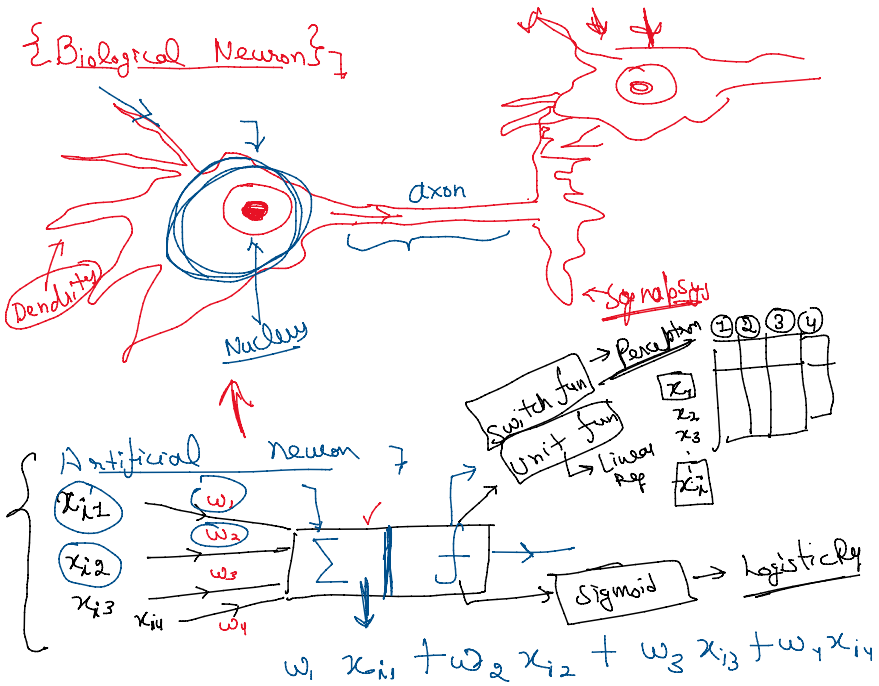
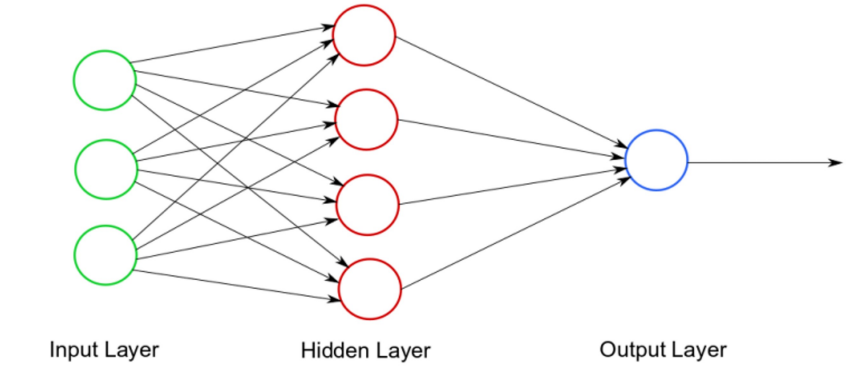
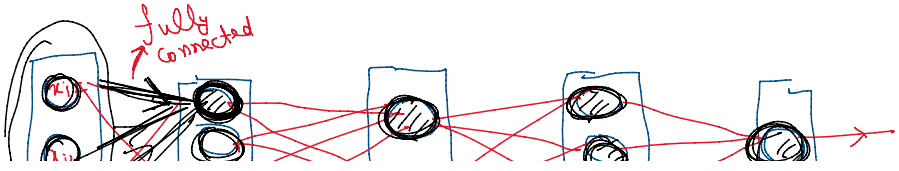
Training a Neural Network

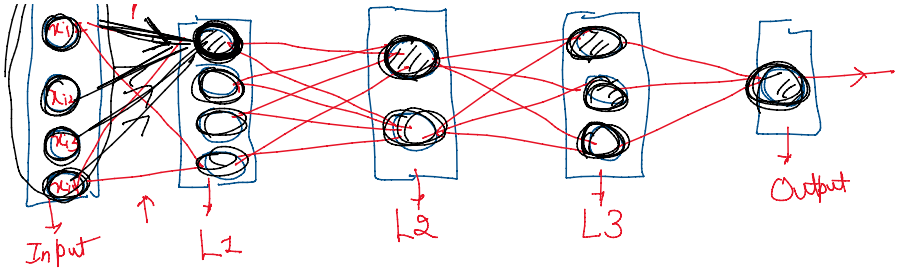
05 August 2020 03:56 PM







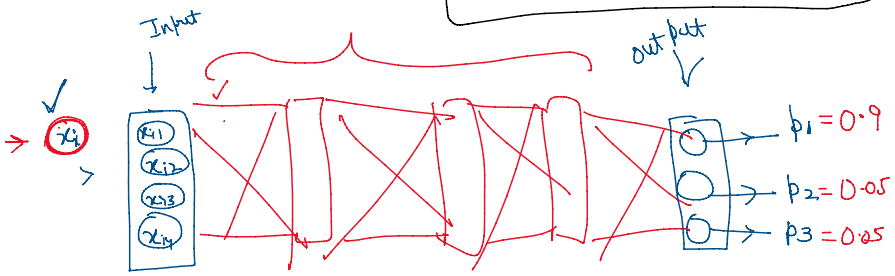




SPECIFICATIONS ↓

→ Graph → Edges → Inter connections
nodes → Artificial Neurons

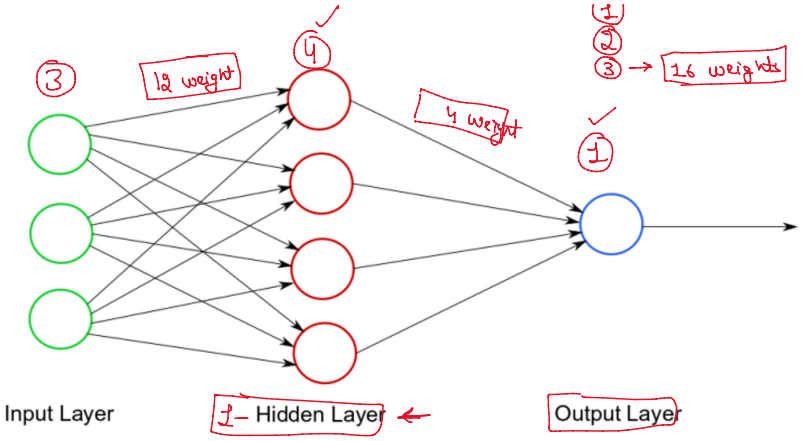
→ Input layer → No. of nodes = No. of cols
{if no feature eng is done}
&
Output layer → If Regression Task ↓
No. of nodes = 1
If Classification Tasks ↓
No. of nodes = No of classes in target variable

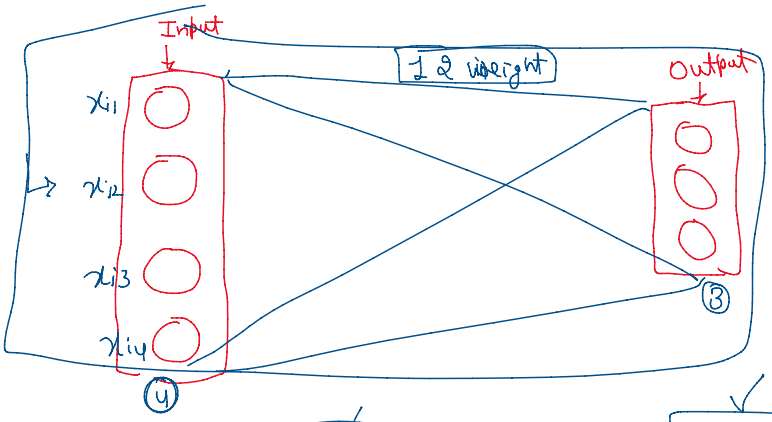


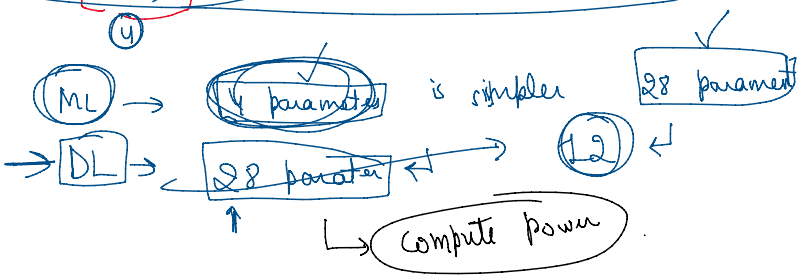
→ Each interconnection (edge) is going to have a weight associated with itself

→ Each Neuron (node) is going to have an Activation function

→ Topology → Structure

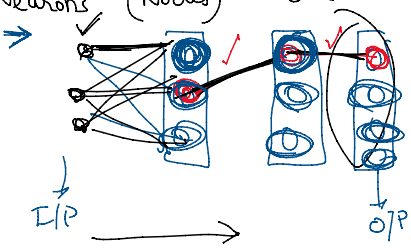






Assumptions to simply the architecture of NN

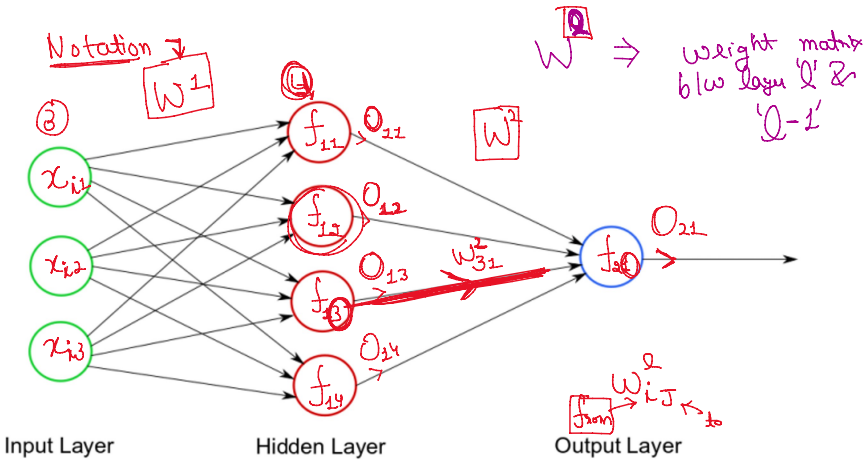
1. Neurons (Nodes) are arranged in layer



2. Multiple layers of neurons will exist & these layers are arranged sequentially
3. Neurons that belong to the same layer will never interact (no interconnections)
4. Neurons are densely connected with the adjacent layer neurons only.
5. Every edge (interconnection) will have weights associated with it.
6. All neurons in Hidden layers will have same

⑥ { All neurons in Hidden layers will have same activation function }

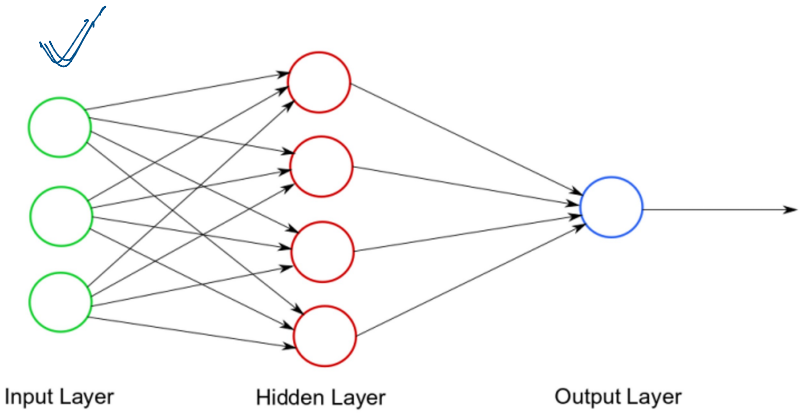
05:15 BREAK



$$W^1 = \begin{bmatrix} w^1_{11} & w^1_{12} & w^1_{13} & w^1_{14} \\ w^1_{21} & w^1_{22} & w^1_{23} & w^1_{24} \\ w^1_{31} & w^1_{32} & w^1_{33} & w^1_{34} \end{bmatrix}_{3*4}$$

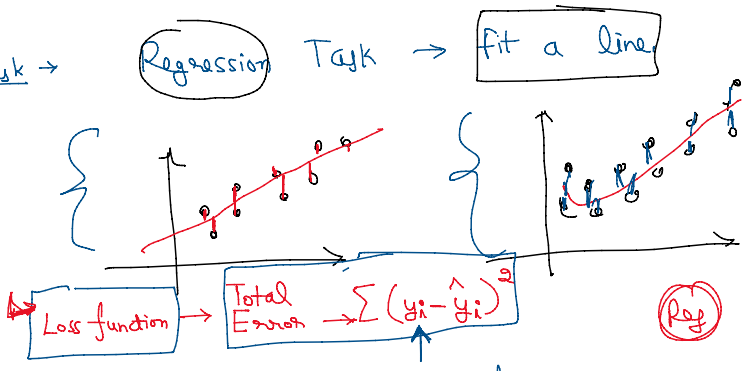
$$W^2 = \begin{bmatrix} w^2_{11} \\ w^2_{21} \\ w^2_{31} \\ w^2_{41} \end{bmatrix}_{4*1}$$

16 weights



Task → Regression Task → fit a line

Task → Regression Task → fit a line.



Loss function → Total Error → $\sum (y_i - \hat{y}_i)^2$ (Reg)

1. Define the Loss function ✓
   - Mean squared errors (y_i, $\hat{y}_i$)
2. Pose an optimization Problem.
   - → $w^* = \arg\min_w \frac{1}{N} \sum (y_i - w^T x_i)^2$
3. Solve this optimization Problem → SGD
   - $W^1 = [\ ]_{3*4} \Rightarrow 12$; $W^2 = [\ ]_{4*1}$ ✓

SGD

4. Random initialization → W^1 (↓ 12), W^2 (↓ 4)
5. Pick a point x_i from dataset ✓

Forward Propagation of a data point

a. Feed forward i^{th} data point through the network ✓

b. Compute the loss of i^{th} data point → $(y_i - \hat{y}_i)^2$

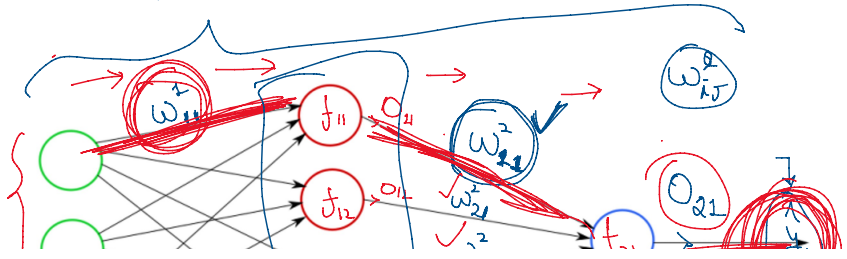
Backward propagation

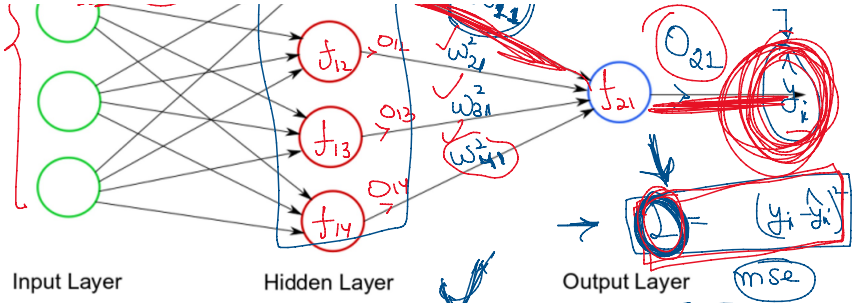
c. Compute all the derivatives using chain rule & differentiation and memoization

d. Update the weights

$$(w_{ij}^l)_{new} = (w_{ij}^l)_{old} - \alpha \frac{\partial L}{\partial (w_{ij}^l)_{old}}$$

6. Repeat Steps 5 until convergence. ✓





W^* $(\omega^2_{12})_{new} = (\omega^2_{11})_{old} - \eta \frac{\partial L}{\partial (\omega^2_{11})_{old}}$

$(\omega^1_{12})_{new} = (\omega^1_{12})_{old} - \eta \frac{\partial L}{\partial \omega^1_{11}} \cdot \frac{\partial O_{21}}{\partial} \cdot \frac{\partial}{\partial \omega'_{11}}$

$\{(\omega'_{11})_{new} = (\omega'_{11})_{old} - \eta$ (very small)

Starting @ 4:10 PM

sloped tangent

Topic :

- Training deep neural networks
- Regularization
- {Coding ::} → {Keras → TensorFlow Development, Development → Research}

$0.01 * 0.01 * 0.01$

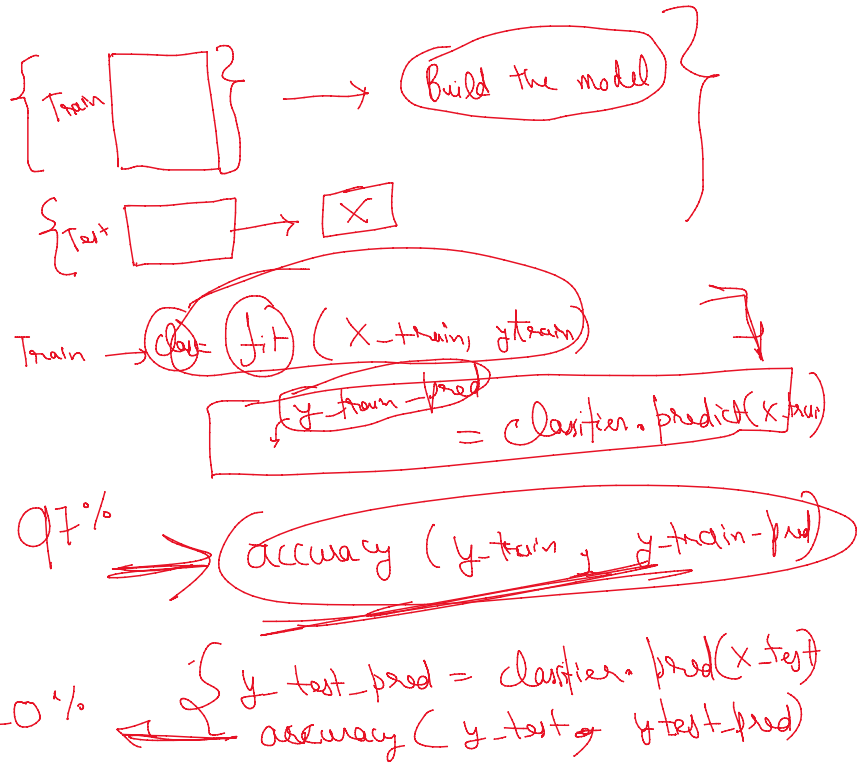
Vanishing Grad → 2000s → 2012

Backprop

Dropout → Nitish

1. Dropout ← {Vanishing grad, Overfitting}
2. Batch Normalization

Overfitting



How to train Deep Neural Networks ?

1. Preprocess the Data → Data Normalization
   → Train_test split ✓

2. Input layer → # of nodes = No. of columns in the dataset (X_train)

3. Hidden layer →
   - Choose no. of Hidden layers.
   - Choose no. of neurons in each hidden layer.
   - Choose an activation function → ReLU ✓
   - Randomly initialize the weights
     If activation function is :
     → Sigmoid / tanh — use glorot normal
     → relu — use he normal
   - Regularization
     → Batch Normalization
     → Dropout

4. Output layer → No. of neurons :

For classification task = No. of classes

For regression task = 1

5. **For Regression** ↴

→ activation function in output layer = Linear

→ loss = mean squared error

→ Metric = mean absolute error

For Classification ↴

→ activation function in output layer = Softmax

→ loss = Categorical cross entropy

→ Metric = Accuracy

6. Choose an Optimizer

↳ Adam

5:19 — 5:25

Evaluates → Test

Reg { RMSE, MSE, MAE

Classif { Accuracy