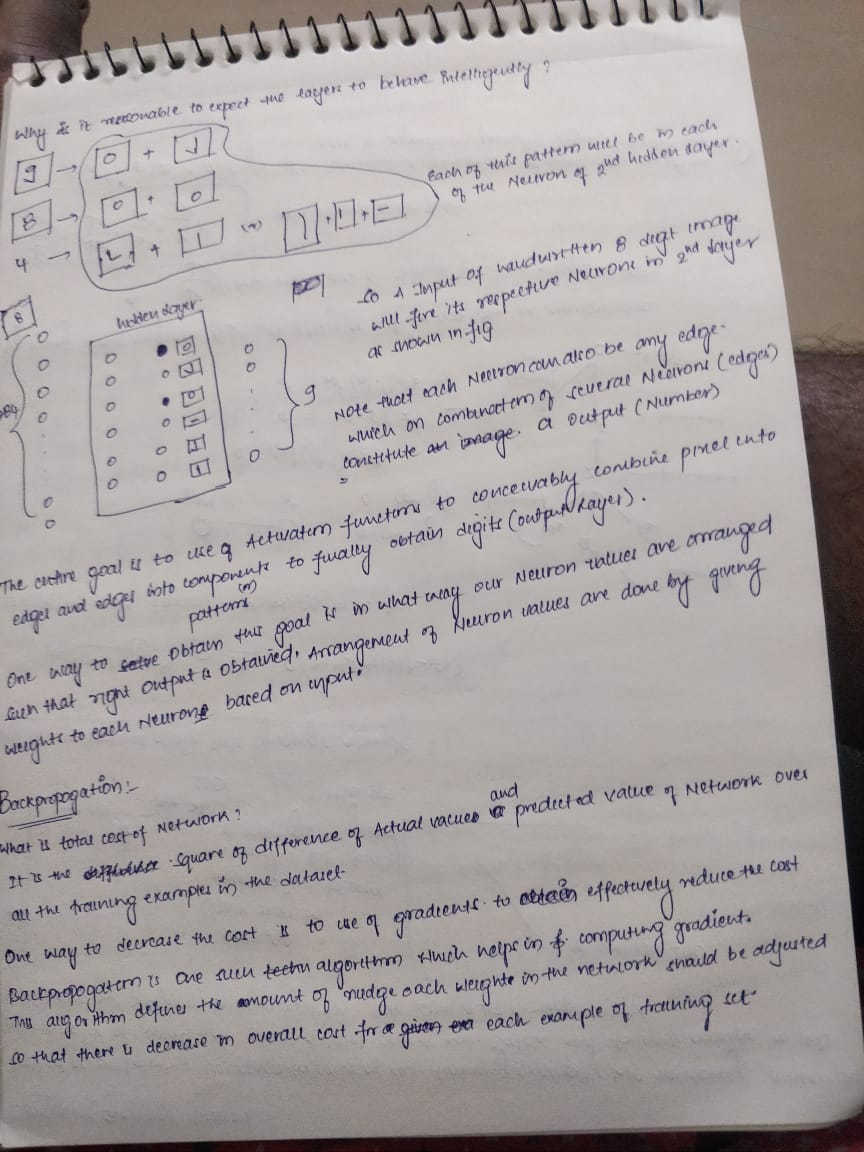
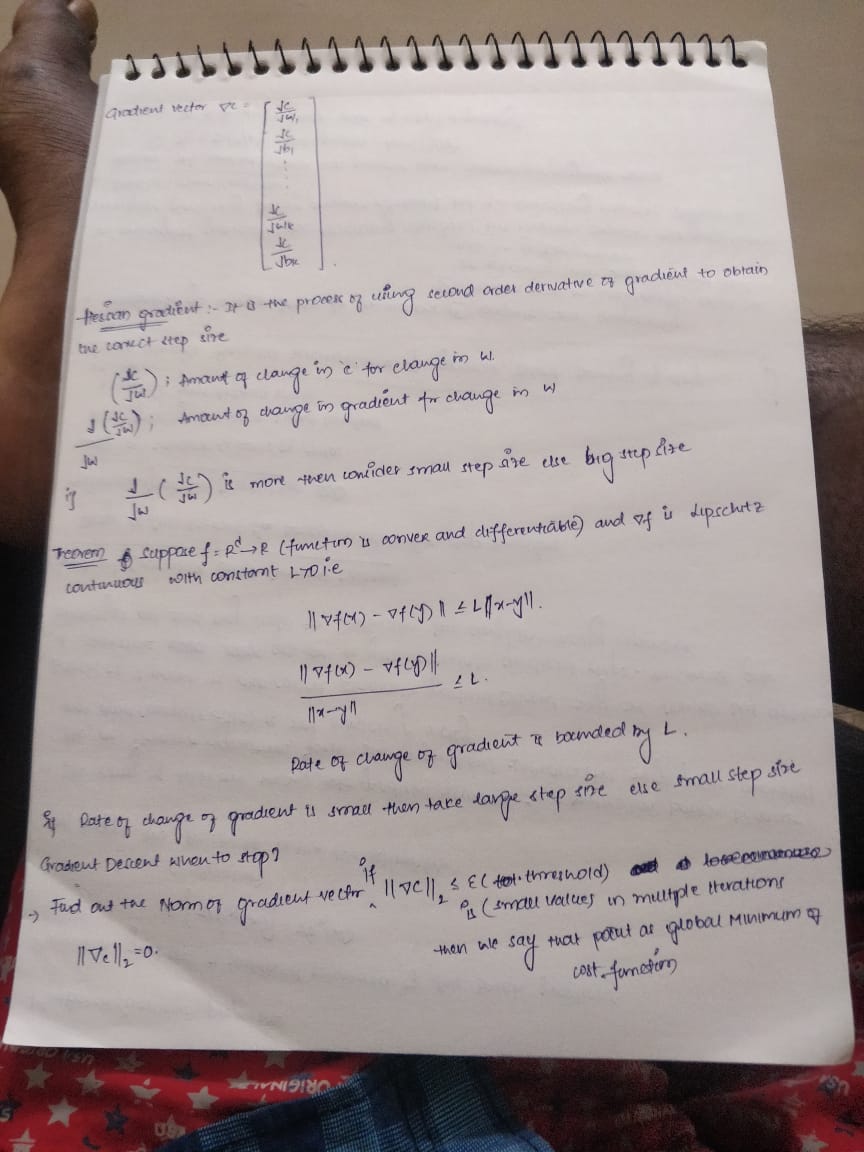
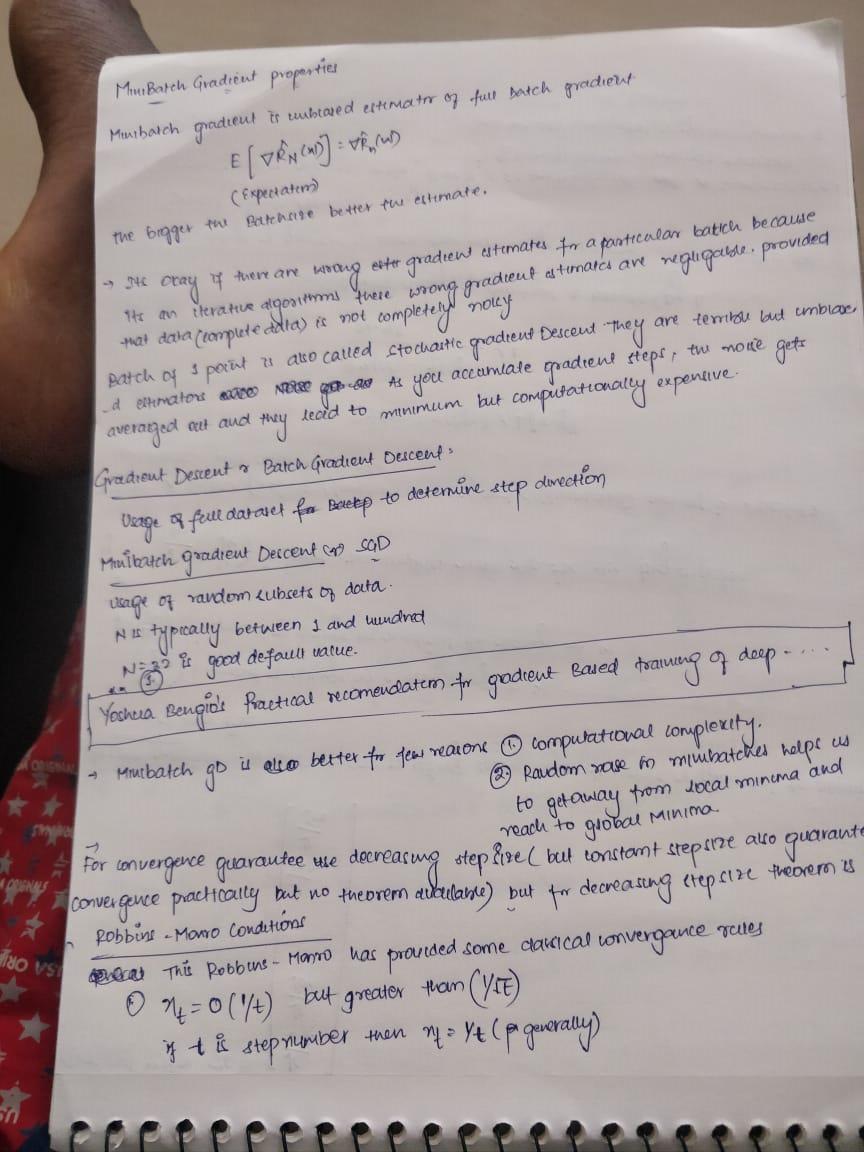
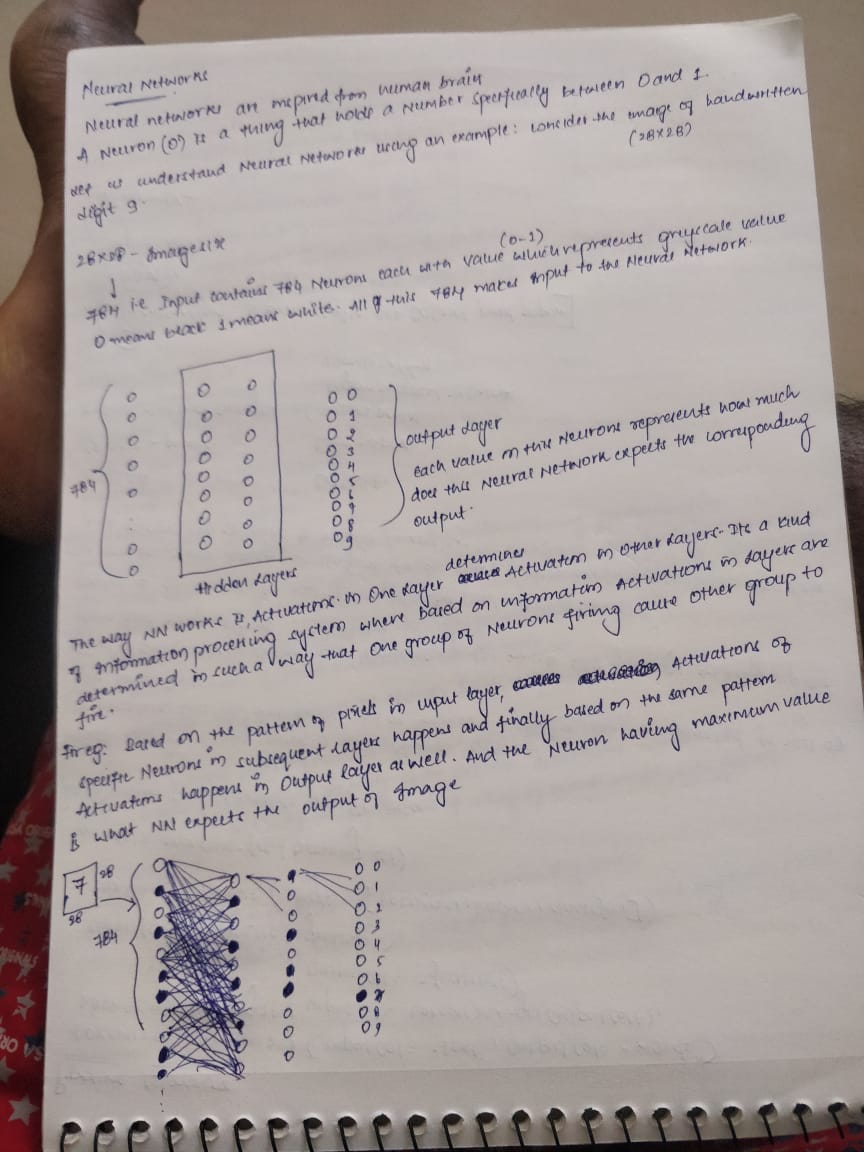
**Neural Networks:**



**Multi Layered Perceptron:**

**What is the use of having MLP’s?**

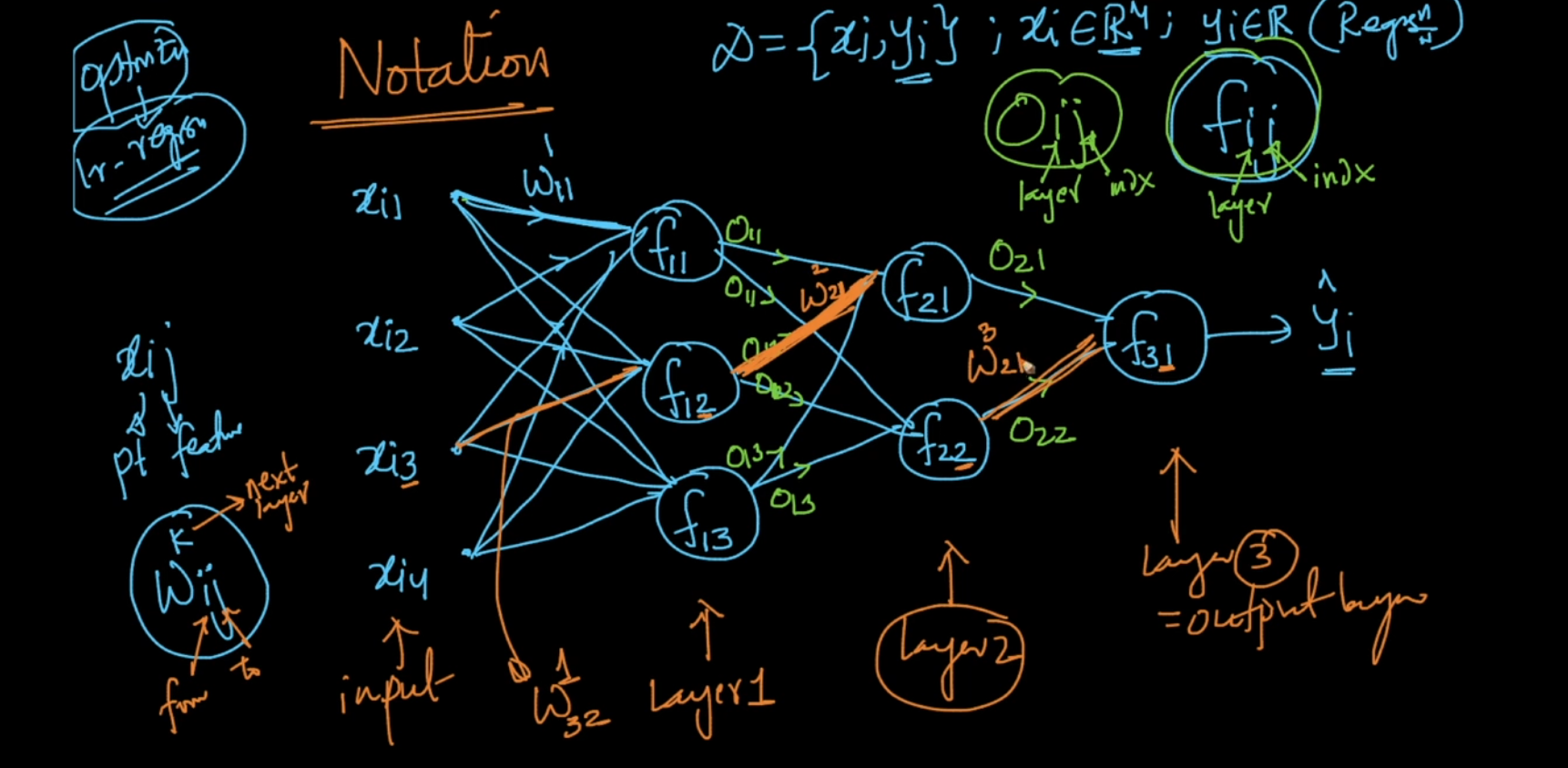
Answer:

1. Biological Inspiration
2. Mathematical Argument:

By using a multi layered structure, we can arrive at complex mathematical functions to solve the problem.

They are very powerful models, but they over fit easily.

**Notation of MLP:**



**What is model Training in neural networks?**

Training means finding the best parameter (weights and biases which have minimum loss values) by using the training data.

The process of model training involves finding out the best parameters W and b at which the cost function is minimum.

This process can be called as optimization and there are optimization algorithms like SGD or GD which leverages the concept of gradients to find the best w and b.

**SGD-Stochastic Gradient Descent:**

Step-1: The algorithm starts by randomly initializing the W’s and b’s

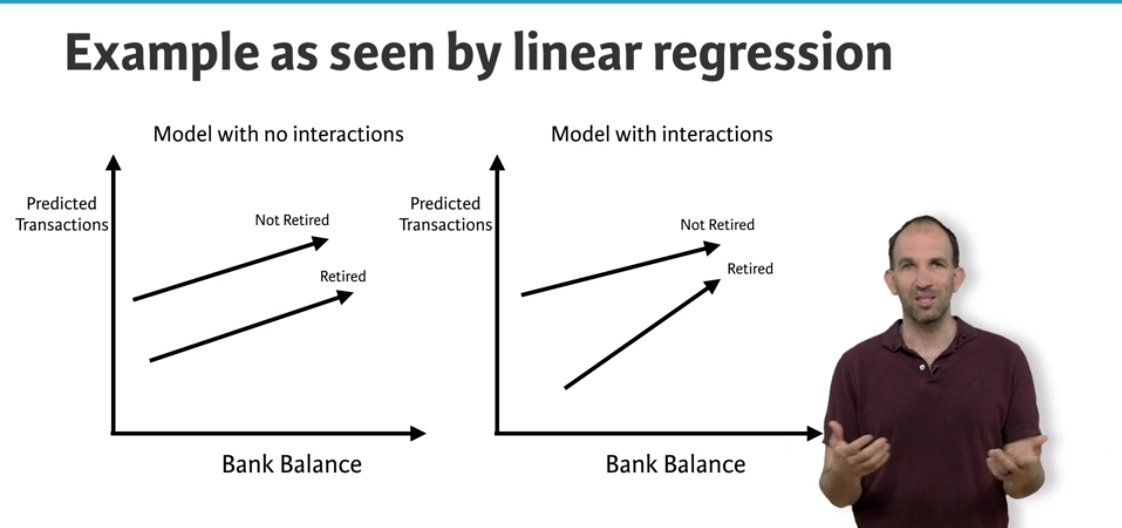
Step-2: Using the randomly initialized values all the datapoints(random subset of training data) are feed forwarded through the network and cost is calculated at the end of the network.

Step-3: Gradients are calculated for each weight( The effect of each W value on cost function is cacluated) and by using below formulae the weights in the network are updated

Wnew=Wold –eta\*Gradient(dc/dw)

Step-4: Using these new weights the random subset of training data is again feed forwarded over the network and cost calculated in last layer and again the gradient is calculated and weights are updated.

Step-5: This process is repeated iteratively whenever the loss function become very minimal and it no longer changes and settles at one particular value.

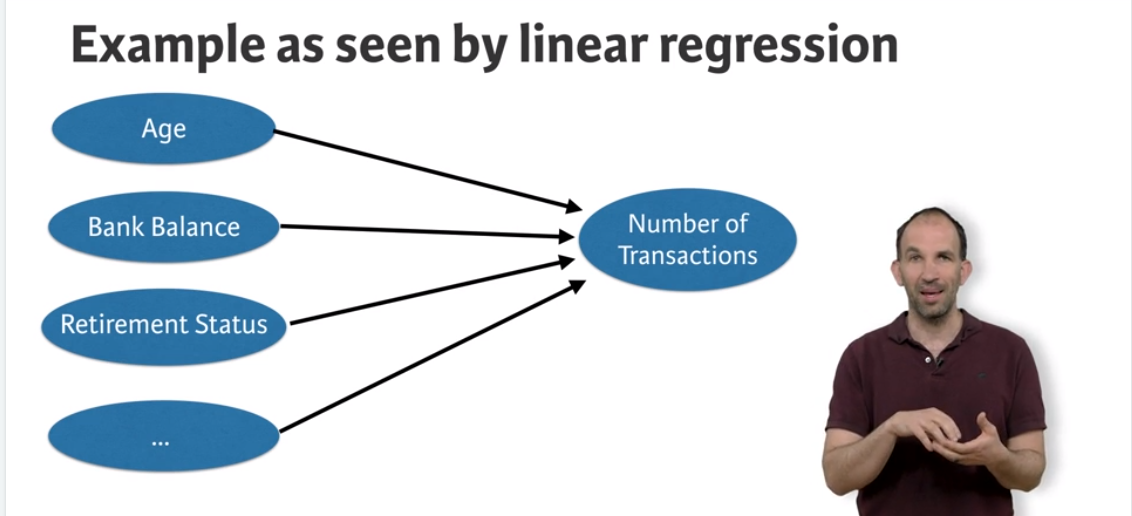


On the left is Linear model without any interactions both the lines are parallel, But in case of neural networks which allows interactions the lines are not parallel.

Neural networks accounts for interactions very well.

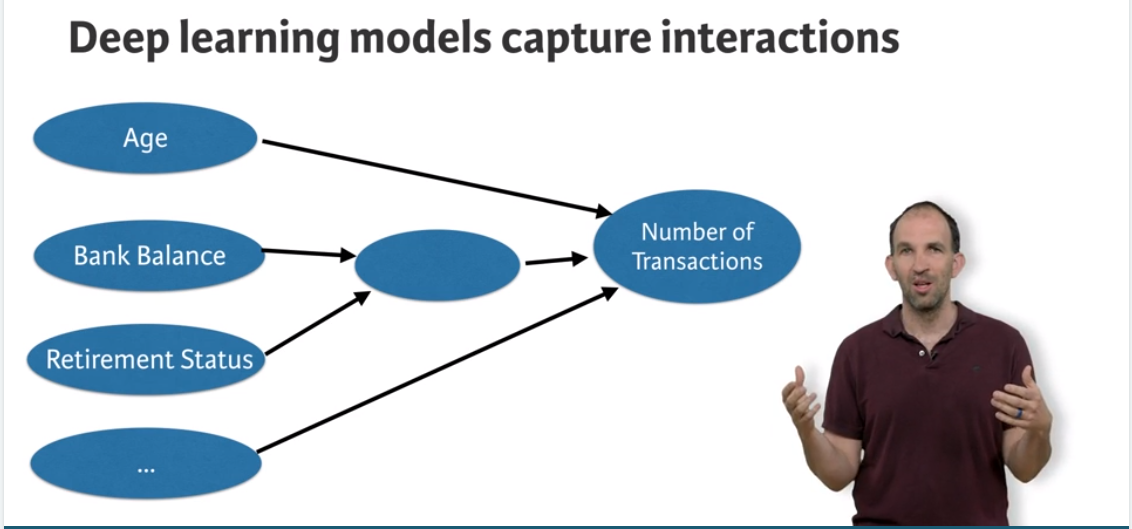
Deep learning uses really powerful neural networks.

**Difference of using Neural Networks:**

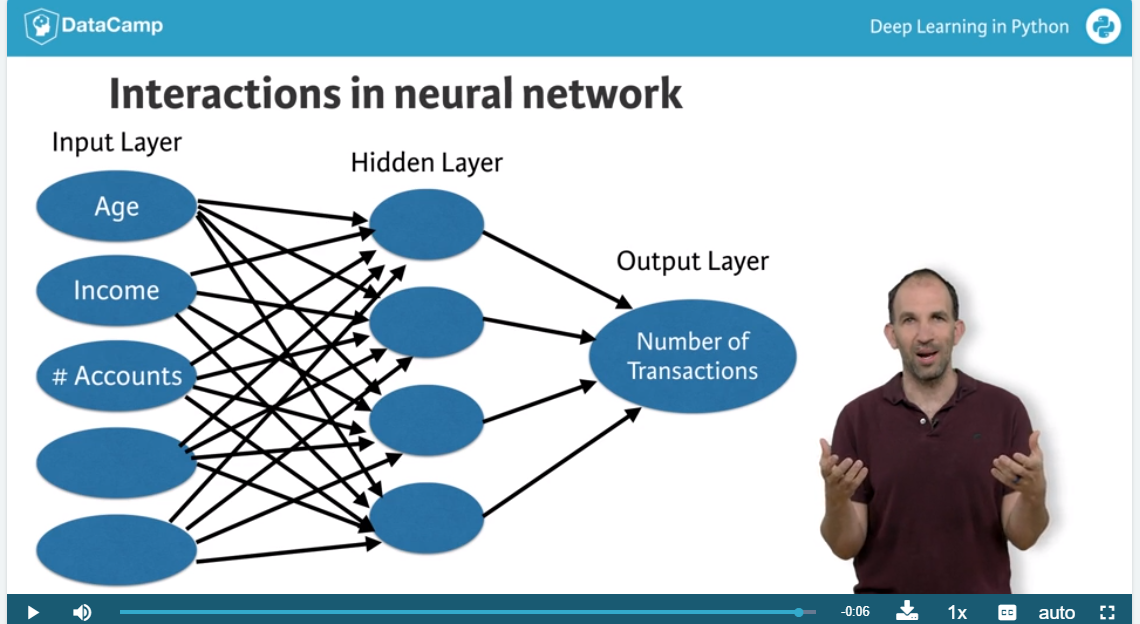


Linear regression predicts number of transactions as sum of individual features and will not consider any interaction between them.

**Whereas Deep learning captures interactions as below:**

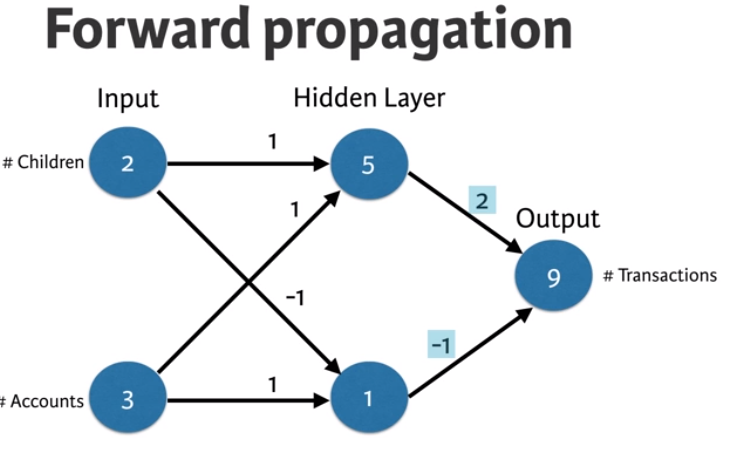


Ideally the below diagram summarises how interactions in Neural networks works.

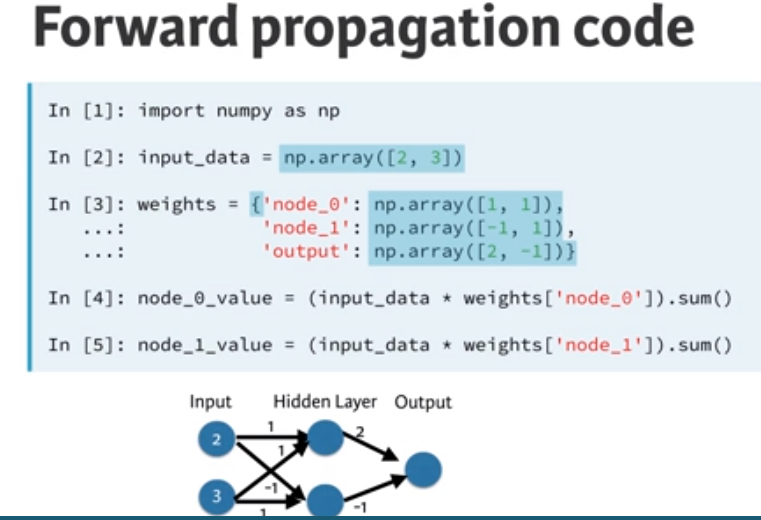


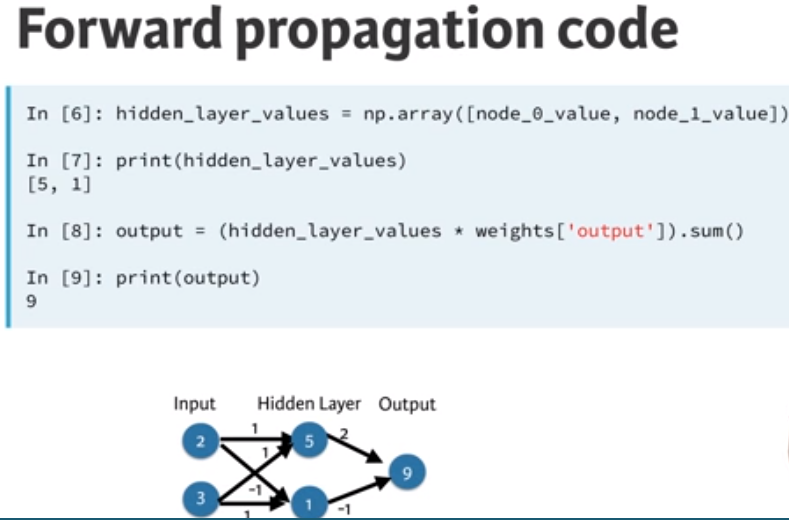
Ideally more the nodes in Hidden layer, more the interactions.

**Forward Propagation:**



Code for above Forward Propagation:





**Activation Functions:**

Activation functions are applied to each node in the Hidden network.

Usage of activation function makes neural networks more powerful by introducing non linearity to the solution.

**Deep Networks:**

Deep networks internally build representations of patterns in the data.

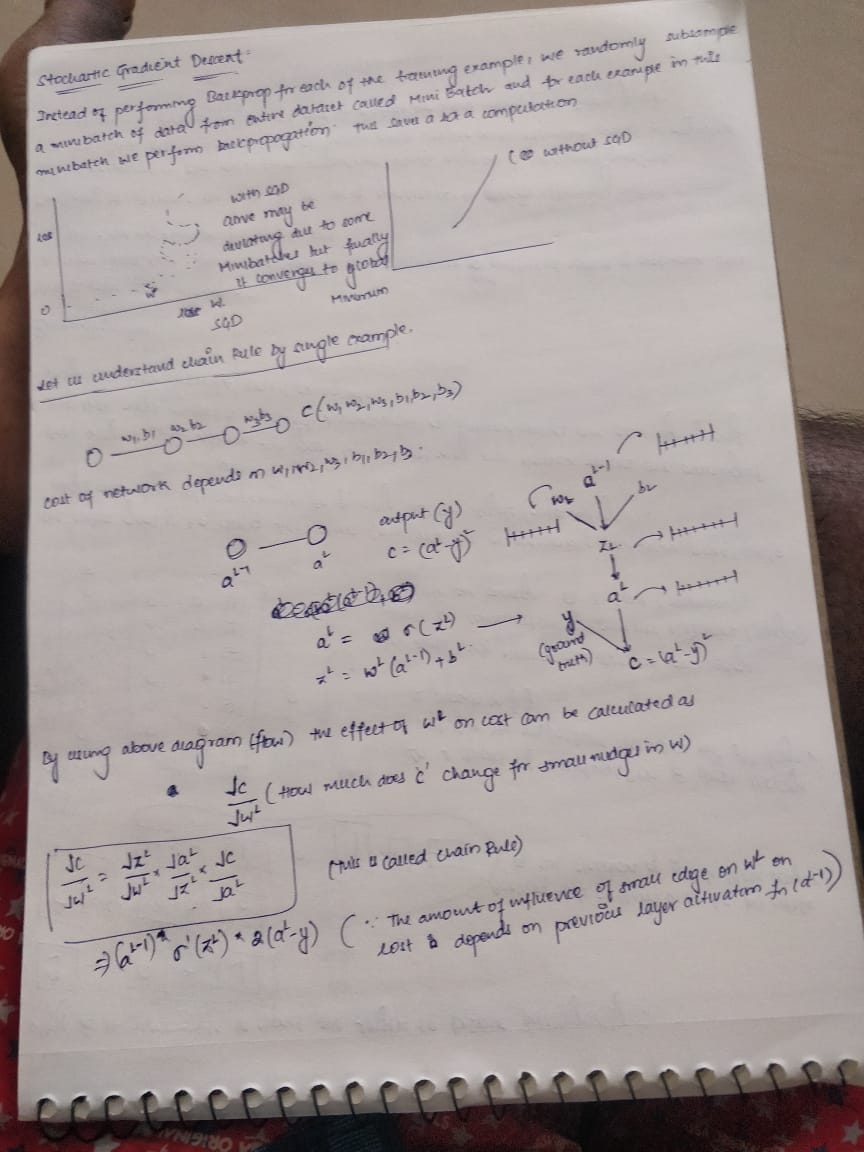
By using Deep Networks , Neural networks partially replace the need of feature engineering.

Subsequent layers in deep networks sophistically build robust representations of raw data.

Good thing about deep learning is the modeller doesn’t need to tell the model of interactions.

Based on the given data the model will automatically find the weights that finds the relevant patters useful for predictions.

**Back Propagation:**

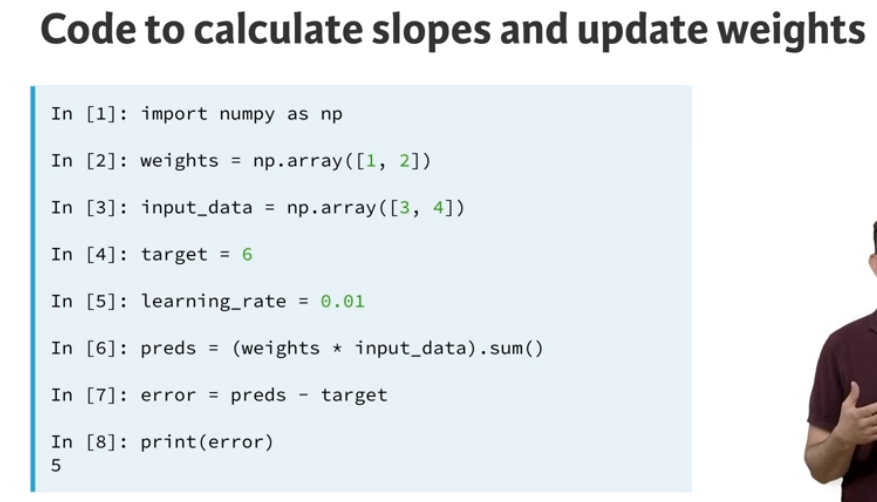
****

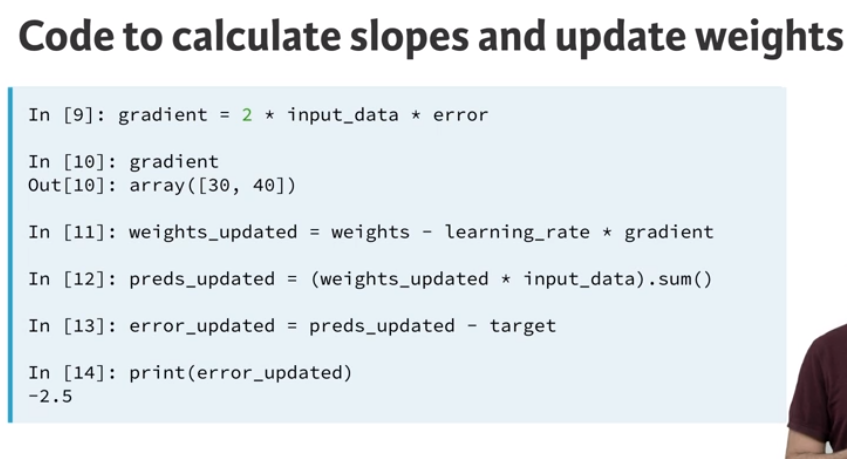
Value of gradient calculated during backprop is obtained using three terms.

1. Derivative of activation function of node for which this weight feeds into (aL node in the above diagram)
2. Activations of the node feeding into the weight (aL-1 node in the above diagram)
3. Derivative of loss function w.r.t the node it feeds into (aL node in our diagram)

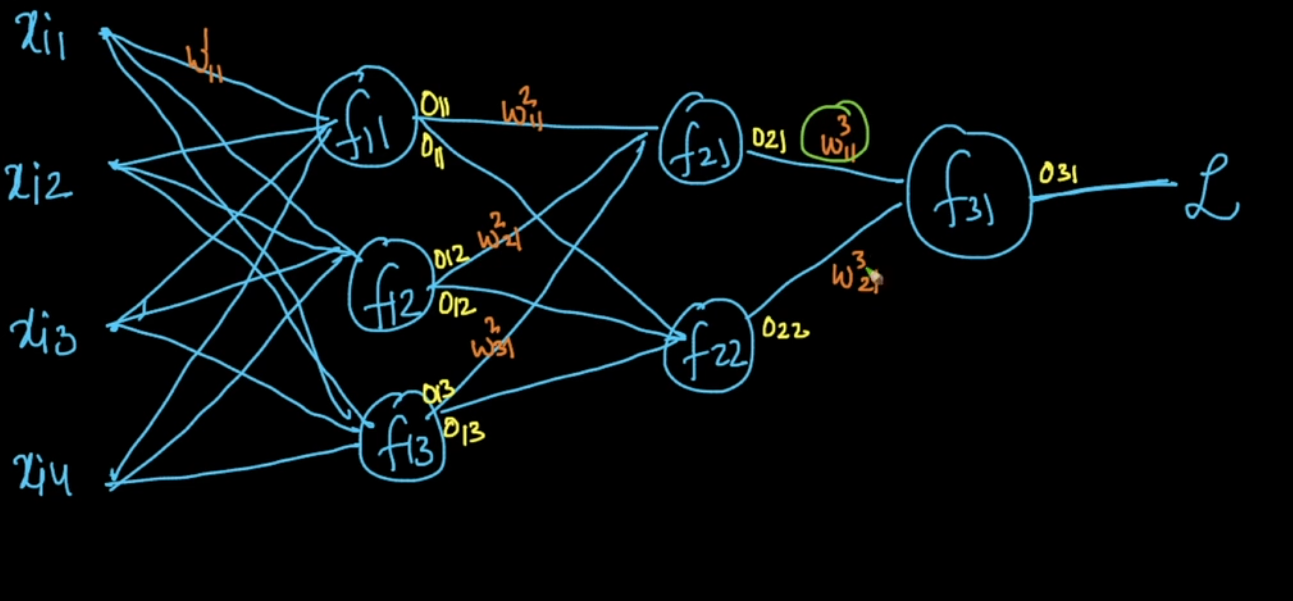
Hence to obtain gradient for a weight during back prop these above 3 values are mandatory.

**Code for Gradient calculation (for example in the above image):**





**Back Propagation in MLP’s (Applied AI ):**



**Using Chain Rule during gradient Calculation:**

**Derivative of L w.r.t to w311 :**

