Kmit

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

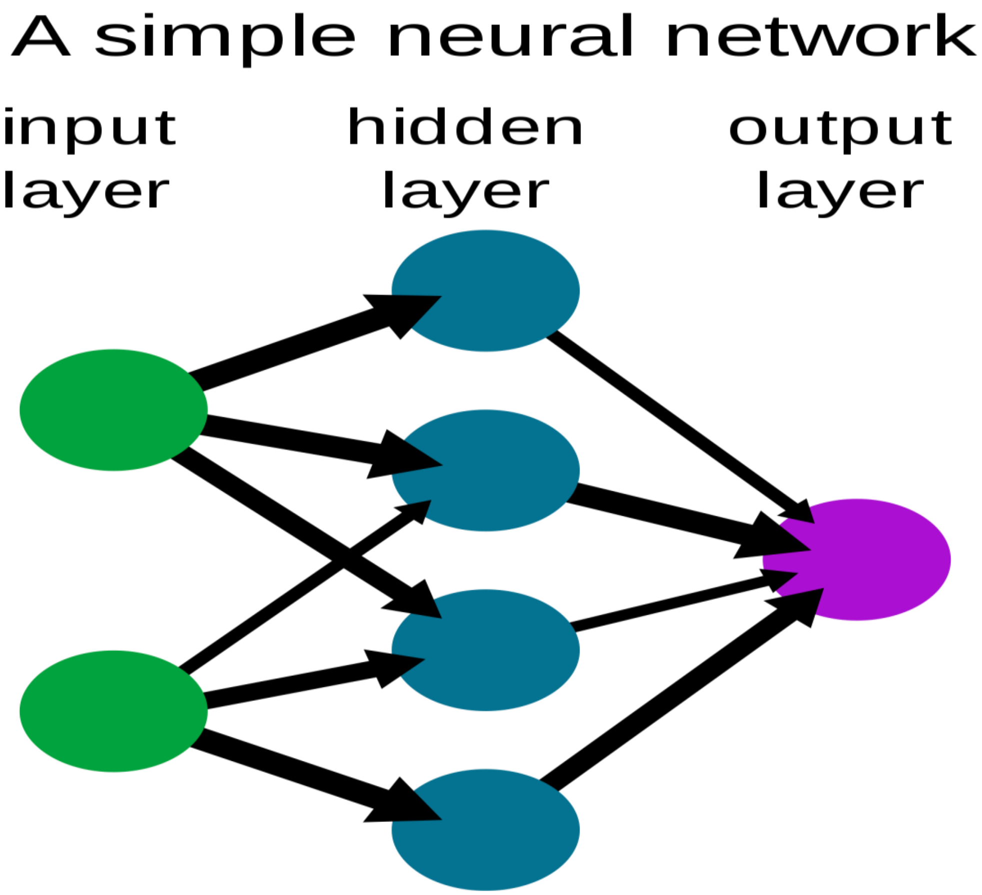
AN AUTONOMOUS INSTITUTION - ACCREDITED BY NAAC WITH 'A' GRADE

Narayanaguda, Hyderabad.

Deep Learning

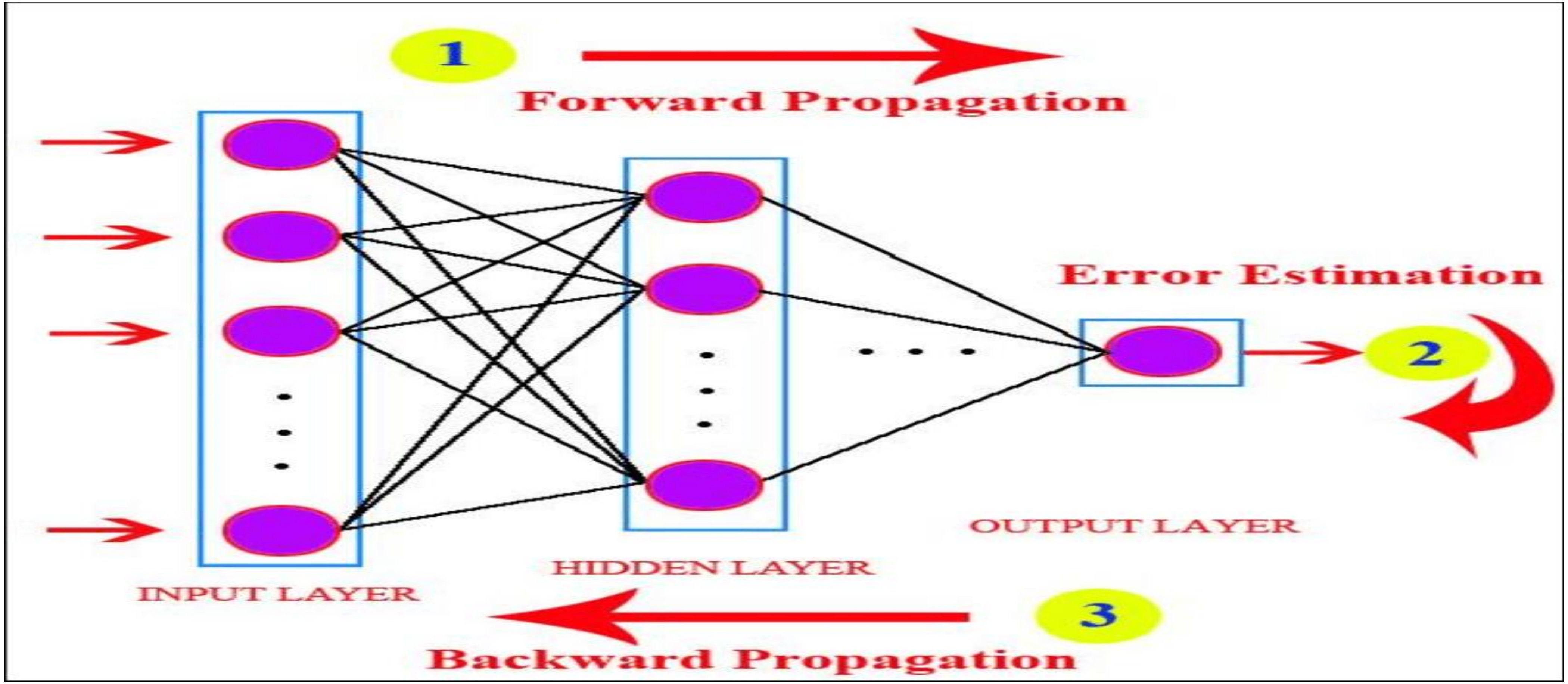
ANN-Binary classification

BY
ASHA



A simple neural network consists of three components :

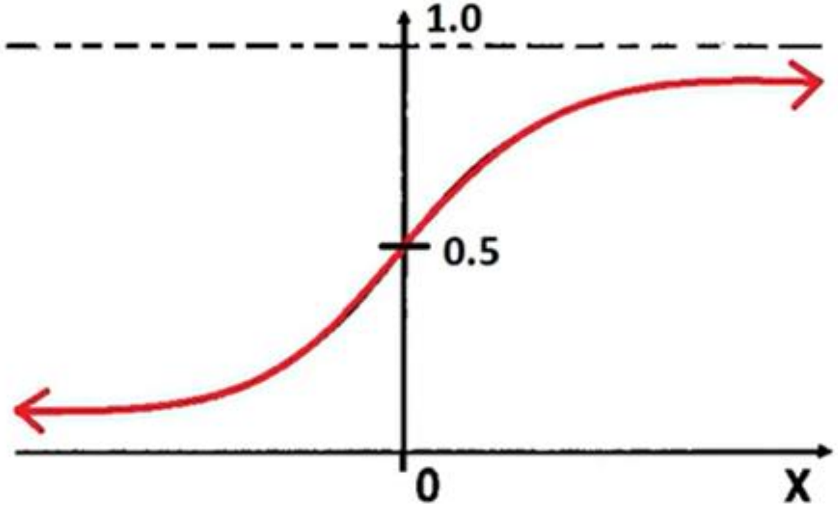
- Input layer
- Hidden layer
- Output layer


1
Forward Propagation
Error Estimation
2
OUTPUT LAYER
HIDDEN LAYER
INPUT LAYER
3
Backward Propagation

Common Activation Functions

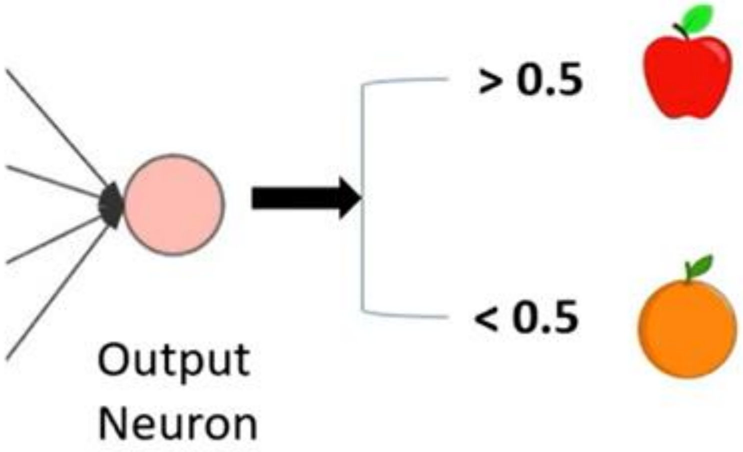
- **Sigmoid**: Maps inputs to the range (0, 1). It’s used in binary classification tasks but can suffer from vanishing gradient problems in deep networks.
- **Tanh (Hyperbolic Tangent)**: Maps inputs to the range (-1, 1), centering the output, which can sometimes lead to better convergence than sigmoid. However, it also suffers from vanishing gradient problems.
- **ReLU (Rectified Linear Unit)**: Introduces non-linearity by outputting the input directly if it's positive and zero otherwise. It’s widely used in deep learning due to its simplicity and effectiveness. It also helps with the vanishing gradient problem.
- **Softmax**: Typically used in the output layer for multi-class classification problems. It converts logits (raw output values) into probabilities that sum to 1 across the classes.

Sigmoid Function



Sigmoid

$$f(x) = \frac{1}{1+e^{-x}}$$



Loss Function

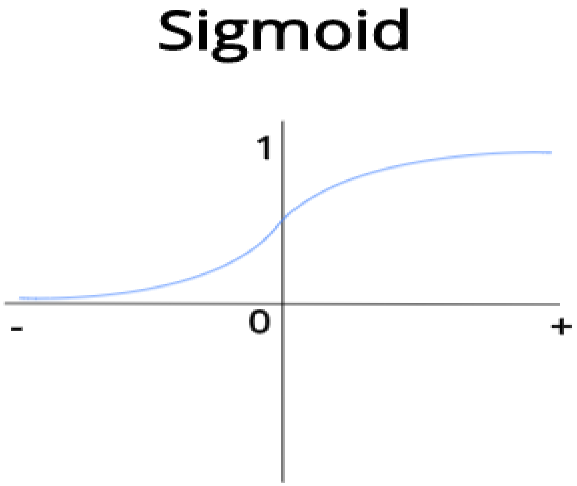
- A measure of how well the neural network's predictions match the actual results. Common loss functions include:
- **Mean Squared Error (MSE):** For regression tasks.
- **Cross-Entropy Loss:** For classification tasks.

Classification Loss:

- ***Binary Classification***:
- Binary Cross Entropy Loss
- ***Multi-Class Classification***:
- Categorical Cross Entropy Loss

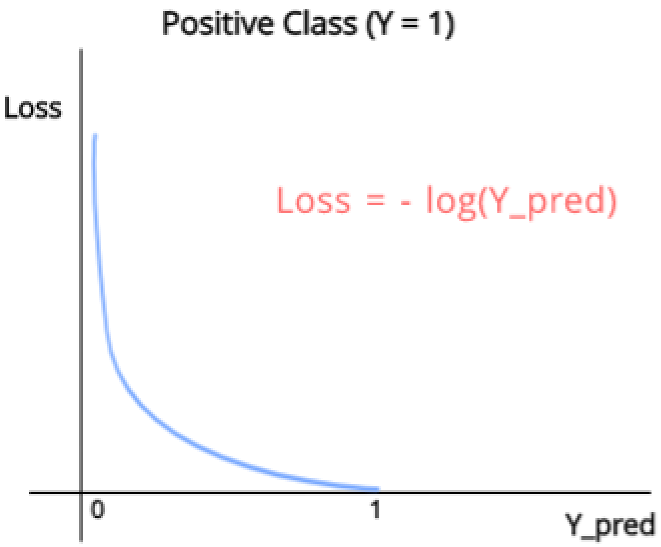
Binary Classification

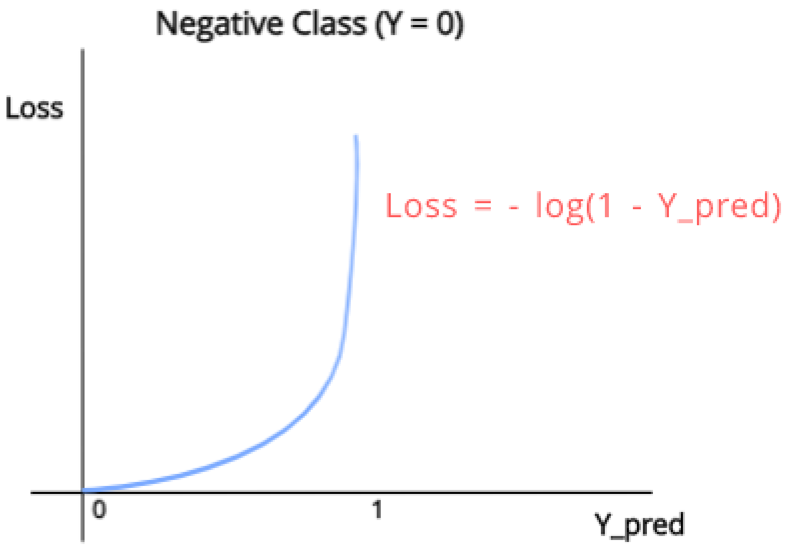
- In binary classification, there will be only one node in the output layer even though we will be predicting between two classes. In order to get the output in a probability format, we need to apply an activation function. Since probability requires a value in between 0 and 1 we will use the **sigmoid function** which can *squish* any real value to a value between 0 and 1.



If the output is above 0.5 (50% Probability), we will consider it to be falling under the **positive class** and if it is below 0.5 we will consider it to be falling under the **negative class.**

The loss function we use for binary classification is called **binary cross entropy (BCE)**.





Y	Y_pred		
1	1	→	Loss is 0
1	0	→	Loss is very high
0	1	→	Loss is very high
0	0	→	Loss is 0

As you can see, there are two separate functions, one for each value of Y.
When we need to predict the **positive class** (Y = 1), we will use
Loss = -log(Y_pred)
And when we need to predict the **negative class** (Y = 0), we will use
Loss = -log(1-Y_pred)

We can mathematically represent the entire loss function into one equation as follows:

$$Loss = (Y)(-log(Y_{pred})) + (1 - Y)(-log(1 - Y_{pred})$$

Remains when Y = 1

Removed when Y = 0

Remains when Y = 0

Removed when Y = 1

BUILDING ANN FROM SCRATCH

Step 1: Load the Dataset

Load the breast cancer dataset from scikit-learn. The Churn Modeling dataset is commonly used in machine learning to predict customer churn in the banking sector. Churn refers to the loss of customers, which can significantly impact a company's profitability. The dataset typically contains various features about customers that can help predict whether they will leave the bank.

Step 2: Split the Dataset

Split the data into training and testing sets. This helps in evaluating the performance of the ANN on unseen data.

Step 3: Normalize the Features

Normalize the input features to ensure all features contribute equally to the training process. This can be done using methods like standard scaling.

Step 4: Initialize the ANN Parameters

Decide on the architecture of the ANN, including the number of layers and the number of neurons in each layer. Initialize the weights and biases for each layer randomly.

Step 5: Define the Activation Functions
Choose activation functions for the neurons in each layer. Common choices are the sigmoid function for the output layer (binary classification).

Step 6: Forward Propagation
Implement the forward propagation process, where the input data passes through the network layer by layer, applying the weights, biases, and activation functions to produce the final output.

Step 7: Compute the Loss
Calculate the loss using an appropriate loss function. For binary classification, the binary cross-entropy loss is typically used.

Step 8: Backward Propagation
Implement backward propagation to compute the gradients of the loss function with respect to the weights and biases. This involves calculating the gradient of the loss function from the output layer back to the input layer.

Step 9: Update the Weights and Biases
Use an optimization algorithm, such as gradient descent, to update the weights and biases using the gradients computed during backward propagation. This step aims to minimize the loss function.

Step 10: Train the ANN
Iterate through the training dataset for a specified number of epochs. In each epoch, perform forward propagation, compute the loss, perform backward propagation, and update the weights and biases.

Step 11: Evaluate the ANN
After training, evaluate the performance of the ANN on the testing set. This involves using the trained model to make predictions on the test data and comparing these predictions to the true labels to calculate metrics like accuracy.

Step 12: Make Predictions
Use the trained ANN to make predictions on new or unseen data by performing forward propagation with the learned weights and biases.

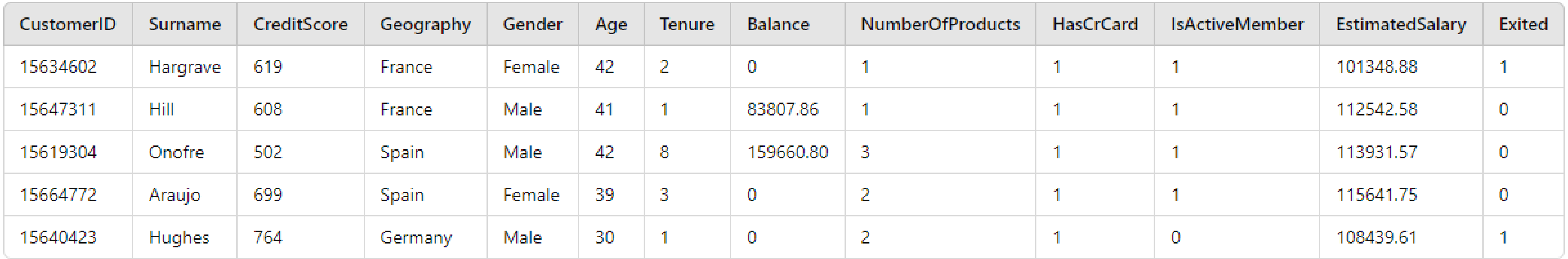
Overview of the Churn Modeling Dataset

A typical **Churn Modeling** dataset for a bank might include the following columns (features):

1. **CustomerID**: Unique identifier for each customer.
2. **Surname**: The surname of the customer (often excluded for modeling).
3. **CreditScore**: The credit score of the customer.
4. **Geography**: Country of residence (e.g., France, Spain, Germany).
5. **Gender**: Gender of the customer (e.g., Male, Female).
6. **Age**: Age of the customer.
7. **Tenure**: Number of years the customer has been with the bank.
8. **Balance**: Account balance.
9. **NumberOfProducts**: Number of products the customer has with the bank.
10. **HasCrCard**: Whether the customer has a credit card (1 = Yes, 0 = No).
11. **IsActiveMember**: Whether the customer is an active member (1 = Yes, 0 = No).
12. **EstimatedSalary**: The estimated salary of the customer.
13. **Exited**: The target variable (1 if the customer has left the bank, 0 otherwise).

Sample Dataset

Here is a small sample of what the Churn Modeling dataset might look like:



CustomerID	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumberOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
15634602	Hargrave	619	France	Female	42	2	0	1	1	1	101348.88	1
15647311	Hill	608	France	Male	41	1	83807.86	1	1	1	112542.58	0
15619304	Onofre	502	Spain	Male	42	8	159660.80	3	1	1	113931.57	0
15664772	Araujo	699	Spain	Female	39	3	0	2	1	1	115641.75	0
15640423	Hughes	764	Germany	Male	30	1	0	2	1	0	108439.61	1

ANY
QUESTIONS

THANK YOU