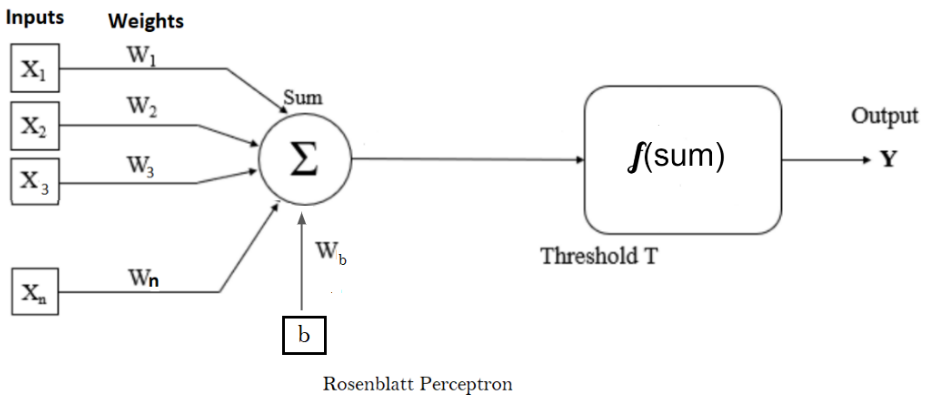
Summarizing Multi-Layer Perceptron

Deep learning helps in solving complex problems, and how a perceptron mimics the human neuron to solve simpler problems. Instead of having a single perceptron, put a bunch of perceptrons together and see how they perform a task.

Let’s have a quick look at the workings of a Perceptron.

Consider a single perceptron that takes some inputs, performs the weighted summation of the inputs, and applies the sigmoid function.

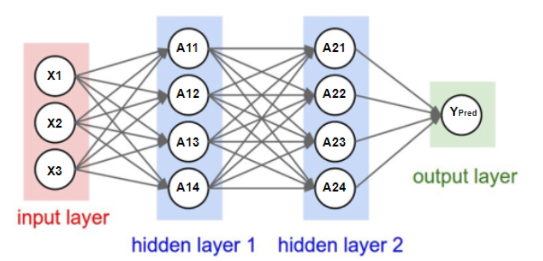
The **sigmoid function** forms an S-shaped graph, where x approaches infinity, the probability becomes 1, and as x approaches negative infinity, the probability becomes 0.



The Logistic regression performs the same as a single perceptron. It finds a model that takes the input, finds the coefficients (as perceptron weights), and predicts the output. Building a logistic regression model is similar to training a perceptron.

When training data X is passed to a perceptron, it returns predicted values called yhat, which should be as close to the actual target values as possible. To achieve this, the neural network will start with some random weights and adjust the weights iteratively by finding the derivatives with respect to each of the weights to get the predictions as close to the actual values as possible.

Instead of a single perceptron, multiple perceptrons are connected to form a deep neural network structure. The first layer is known as the **Input layer,** the final layer is known as the **Output layer,** and the layers lying between the input and output layers are known as the **Hidden layer**. Each layer could have a different number of neurons.



The Bias term in the neural network is the same as the intercept in the line equation. This bias term is the same theta or threshold value in McCulloch Pitts neuron. Instead of having this value in the function, it is added as part of inputs having weight associated with it.

Every node (neuron) contains a bias term, which may or may not be explicitly showcased in the neural network architecture. A neural network has a bunch of inputs, and every node has some weight. It computes the weighted summation of the input, applies the activation function, passes the output to the next layer, and so on.

Happy Learning!!