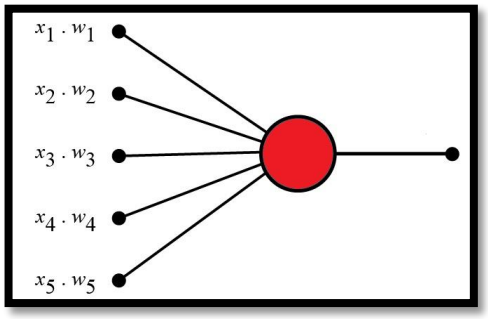
Perceptron: https://www.w3schools.com/ai/ai_perceptrons.asp



1. Used for supervised machine learning
2. Also called binary classifier (0,1)
3. This always divides your data into two regions
   - In case of 2 inputs … creates a line b/w two regions (x, y)
   - In case of 3 inputs … creates a plane b/w regions (x, y, z)
   - In case of 4 or >4 … creates hyper-plane b/w regions
4. **Line Equation**…2D(Ax+By+C=0)…3D(Ax+By+Cz+D=0)
5. Perceptron has two stages (i. **Training** ii. **Prediction**)
6. In training perceptron calculates **weights** and **bias** values (dataset)
7. (**Input Values →Summation →Threshold Value →Activation Function →Output**)
8. **Limitation**: Perceptron is used to classify **linear** or sort of linear (less accuracy) data. (no non-linear)

Perceptron Trick: How to train a perceptron and find weights and bias

1. Pick a random data and run a loop until **convergence** or generally **1000 times (epoch)** or more **10k**
2. How to label first line data point **regions** i.e. Positive or Negative (https://www.desmos.com/calculator)
3. Ax+ By+ C =0
   - Changing in C moves line parallel
   - Moving line towards negative direction (+)
   - Moving line towards positive direction (-)

Example

Line coefficients values (e.g. 2x+ 3y+ 1 = 0)
Data point coordinates (e.g. 4, 6)
Step-01: Put 1 at the end of data point coordinates (i.e. 4, 6, 1)
Step-02: Towards positive direction 2 3 1

- 4 6 1

-2x -3y 0 (Put in demos calculator)

4. We cannot make large line shifts for this we use **learning rate** (0.01 to 0.1) to move slightly;

New coefficients = old coefficients – learning rate * data point coordinates
= 2 3 1 – 0.01 * (4 6 1)
= 2 3 1 – 0.04 0.06 0.01

Limitations:

- First problem is the line (i.e. w1, w2 and b values) we get from this trick we cannot say that these values or line is best for classifying that data.
- You cannot quantify or tell how good or bad your result is.

Perceptron Loss Function:

1. **Loss function** is a way of telling that how good or bad your ML model is performing. (Quantifies)
   In case of perceptron it is **f (w1 w2 b)**