- Performance of traditional learning algorithms (like **SVM/ logistic regression**) plateaus after certain amout of training data.
- Sigmoid activation has nearly zero gradient for large positive and negative values making learning harder.
- On the other hand, ReLU has 1 gradient for all the positive values.
- **Linear Regression** - compute a line using the train dataset (to classify data or to get predictions) $$ y = w^TX+b$$
- **Logistic regression** - algorithm used when the output labels are either 0/1 (binary classification)
  $$y' = sigmoid(w^TX+b)$$ Convex Cost function of logistic regression model is $$ J(w,b)=-1/n \sum_{i=0}^n y~log(y') + (1-y)log(1-y')$$ $$w := w - learningrate*dJ(w,b)/dw$$ $$b := b - learningrate*dJ(w,b)/db$$
- Avoid using for loops for deep learning computations and rather use the builtin python and numpy functions. The builtin functions can parallelize the operations.
- **Avoid using rank 1 aray in python**:

```
    a = np.random.rand(5)
    a.shape() will be (5,)
```

Instead use:

```
    a = np.random.randn(5,1)
    a.shape() will be (5,1)
```

- **Activation functions** -
  - tanh function works better than the sigmoid function (almost always) as the range of the former function is between -1 to 1 which leads to almost 0 mean data.
  - 0 mean data is easier to learn for the next layer.
  - Whereas in sigmoid activation function the mean of the data is around 0.5
  - Sigmoid and tanh have 0 slope for large and small values.
  - 0 slope slows down gradient descent or the learning process.
  - ReLU activation function can be used instead.
  - If the putput of your neural network is between 0 to 1 (binary classification) then you can use sgimoid activation function in o/p layer
- **Circuit theory and deep networks** - There are functions that can be computed by a small 'L' layer deep network that shallower networks require exponentially more hidden units to compute.
  - Example - Compute XOR of n inputs
  - Deep networks reuire O(n) hidden neurons, shallow nework of one hidden layer requires $O(2^n)$ neurons
- Hyperparameters which give best performance may change with time due to change in computer infrstructure/ CPU/ GPU. So it is good to try different hyperparameters once in a while for best performance.