# Introduction to Neural Nets videos #project #done #t-cycle-1

- Why do we need machine learning ?
  - Solutions to certain class of problems can't be coded by hand into programs. E.g., probability of credit card transaction being a fraud, detecting a custom object in a novel orientation and new lighting in a cluttered scene.
  - ML Approach is to collect examples with correct answers (training set or training examples) and use an algorithm to learn tha charateristics.
- Datasets
  - MNIST database on handwitten digits is the ML equivalent of fruit-flies.
  - ImageNet Object recognition challenge task database
- What are (real) Neural Networks ? How the (nourons in the) brain works ?
  - Neurons receive inputs from dendridic tree that are connected to other (upstream) neurons. They send the output through the axon which are connected to downstream neurons through a synapse.
  - When the inputs are more than a certain level it causes the "axon hillock" to trigger a depolarization wave which makes the downstream synapse to release receptor molecules (vescicles). These in-turn bond to cell membrane of downstream neuron chaging their shape so they will let ions or other molecules in and therefore changing the behaviour of the downstream neuron (increasing or decreasing the charge - which results in the neuron firing or not).
  - Axons branch, new synapses get created and synapses learn to change with learning to increase of reduce the signal they pass-on to the downstream neuron. Mystery is what rules do they use ?
  - Neurons start out generic and become specialized with learning. So early on in learning it's possible for the brain to relocate functionality from one part to another.
  - Typical human brain has ≈10^{11} neurons with each having ≈10^{4} synapses. So in effect ≈10^{14}⟶10^{15} synapses in all. A VERY large number.
- Simple Models of Neurons
  - Idealized models. capture essential details but are "wrong" - details can be added back to make it more faithful to reality. But remember model is not reality.
  - **Linear Neurons**: $y = b + \sum_i x_i w_i$
  - **Binary Threshold neurons**
    $$y = \begin{cases} 1 & \text{if } (b + \sum_i x_i w_i) > 0 \\ 0 & \text{otherwise} \end{cases}$$
  - **Rectified linear neuron**
    $$y = \begin{cases} z & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases} \quad \text{where } z = (b + \sum_i x_i w_i)$$

    - **Sigmoid neurons** (Have smooth derivatives, make learning using backprop easier)
      $y = \frac{1}{1+e^{-z}}$ where $z = (b + \sum_i x_i w_i)$
    - **Stochastic binary neuron** use sigmoid function result as the probability they will fire - but output is 1 or 0 the probability of 1 is given by sigmoid function applied to the inputs.
- A Simple example of learning
    - 2 layers - 1 input layer (1 neuron per pixel) and 1 output layer (1 neuron per class). In this case we are using 400 pixel images to recognize 10 digits (0-9).
    - rep weights as a map per class. So each class has a map that has value in (x, y) equal to weight of pixel at input location (x.y) for the class. Weights can be random at start and to be adjusted during learning.
    - Given an image, in this method, each class will output a likelihood value. Top valued class is the guess. Goal is to make guess accurate.
    - For each training example, increment weights from active pixels to the correct class. Then decrement weights from active pixels to whatever class the network guesses.
    - This can decently learn "templates" for digit shapes but is insufficient - e.g., where the allowable variations are large.
- Three types of learning
    - Supervised Learning
        - Each traning case contains input vector x and target output y. Correct answer is known.
        - Regression -> The target output is a real number/vector.
        - Classification -> The target output is a class label.
        - Start by choosing a model class and learn (adjust parameters of model) to minimize error.
    - Re-inforcement learning
        - output is an action or sequence of actions and only supervisory signal is an occasional scalar reward.
        - Goal is to maximize future rewards. Discount factor gives immediate reward more weight than far-in-future rewards.
    - Unsupervised Learning
        - Was largely ignored. Mainly clustering. Can be used to pre-process data before SL or RL. Principal Component Analysis