**Lecture 1.1**

**Why we need Machine Learning?**

Sometimes it is difficult to write programs to satisfy every specific need. Applications like speech recognition, object detection and fraud detection keep changing overtime. Hence new programs cannot be written over and over again.

On the other hand what we can do is, have a bunch of inputs with results to be fed to a system to learn patterns and features in it. In this way the system comes up with its own program during the training mechanism which can be used on an entirely new dataset (test data).

Areas of interest are:

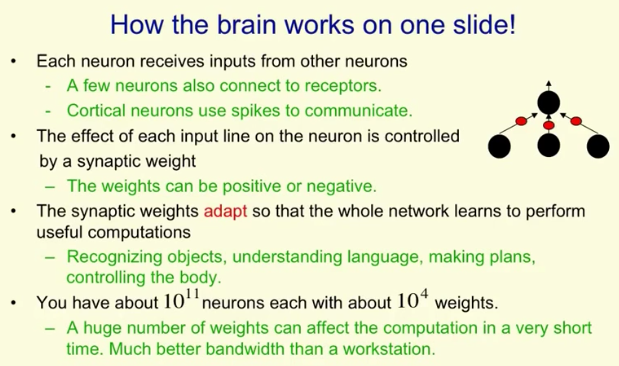
* Object Detection
* Pattern Recognition
* Speech Recognition

**Lecture 1.2**

**What are Neural Networks?**

Reasons to study neural computation:

* To understand how the brain actually works
* To understand a style of parallel computation inspired by neurons and their adaptive connections
* To solve practical problems using novel learning algorithms developed by the brain.



Any part of the brain can adapt to learn anything (vision/auditory)

**Lecture 1.3**

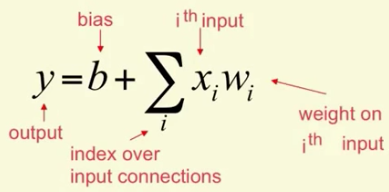
**Simple models of neurons**

To understand something complicated it must be idealized:

* Atoms -> solar systems
* Idealization removes complicated details not essential for understanding main principles
* Once basic principles are understood, it is easy to add complexity to make a more faithful model
* A point to note: it is worth understanding models that are known to be wrong.

Eg: neurons communicate real values rather than spikes of activity (in reality this is false).

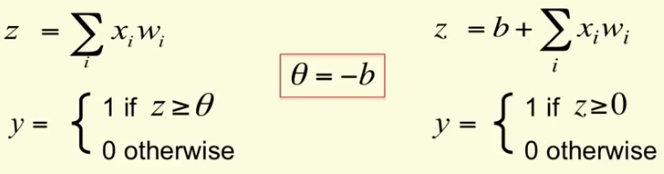
**Linear neurons**



**Binary Threshold Neurons**

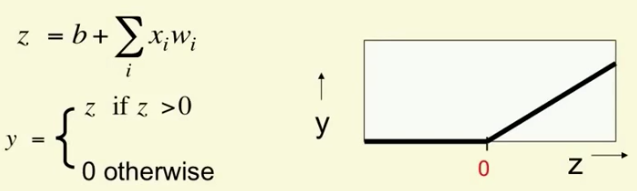
First weighted sum of inputs is computed and then those above threshold are passed.

Equation equivalence:



**Rectified Linear Neurons**

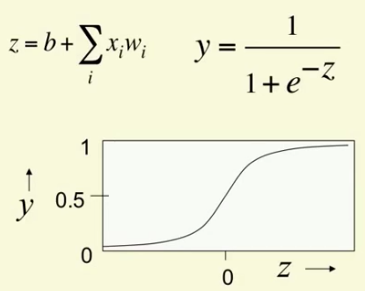
* First computes weighted sum of inputs
* Then output is a non-linear function of total input



* So it is linear above 0
* And makes hard decisions at 0

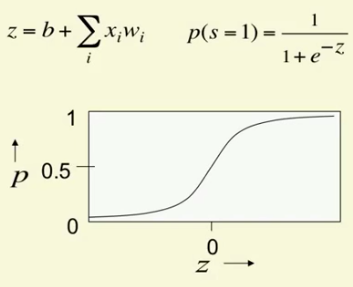
**Sigmoid Neurons**

Output is smooth. Logistic function is used.



**Stochastic Binary Neurons**

Similar to logistic units but the output is treated as the probability of producing a spike.



When similar thing is done for ReLu, the output is treated as Poisson rate for spikes.

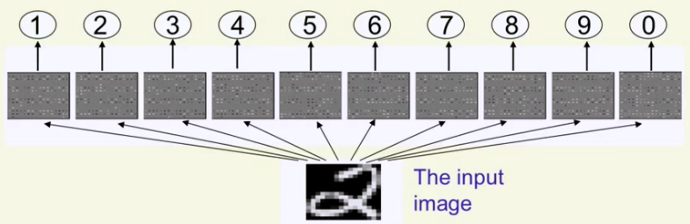
**Lecture 1.4**

**Simple Example of Learning**

Considering a 2-layer neural network for digit recognition:

* First layer is the input neurons containing pixel intensities
* Second layer is the output neurons representing the digits (shapes)

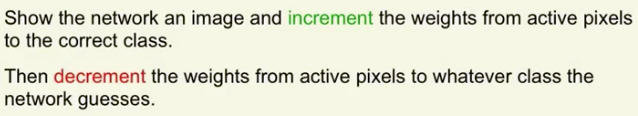
So given a digit, the output neuron for that digit must be fired.



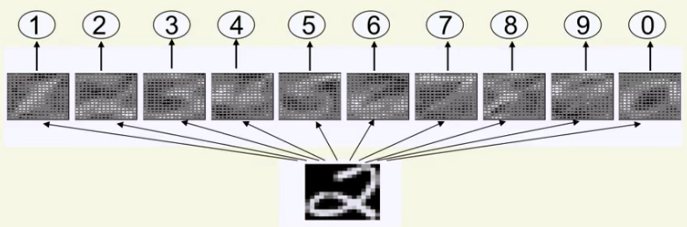
In order to visualize the weights for a digit while training, each output unit has its own ‘*map*’ of the input image to show the weights of pixel in particular location.

Initial weights are randomly assigned.

In order to avoid giving high weights to all classes:



After learning:



A simple two layer network is insufficient. This is because it is equivalent to template matching and there are no features to capture.

**Lecture 1.5**

**Three Types of Learning**

* Supervised learning
* Reinforced learning
* Unsupervised learning

**Supervised learning:**

* Classification
* Regression

*How supervised learning works?*

* We choose a model class (a function) having some parameters ‘W’, that maps each input vectors ‘x’ to predicted outputs ‘y’. 
* While learning these parameters are adjusted in every iteration so that the difference between the predicted output and actual target output are as minimal as possible.

**Reinforcement Learning:**

**Unsupervised Learning:**