Computer Vision and Machine Learning

(Neural Network-1)

Bhabatosh Chanda

bchanda57@gmail.com

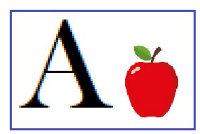
Beginning

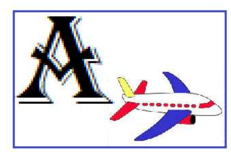
- The concept of 'Artificial Intelligence' cropped up in the first half of 20th century.
- Advent of digital computer produced a machine which is found to be superior to human deings in terms of number crunching.
- People started wondering if this machine be made to perform the other tasks usually done by human beings, such as
  - Identifying objects or events
  - Understanding and translating different kinds of signal: textual, audio, video
  - Making decision
- In short, if machine can mimic the intelligent behavior of humans.

Story: Gender identification

- Two-class classification problem
- Moral of the story –
  - ***Data distribution:*** The distribution of test data should be same as that based on which prior is developed.
  - ***Data imbalance:*** Size of data from different classes should be comparable.

Learning

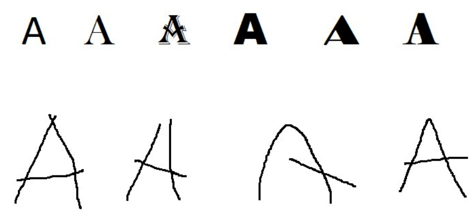




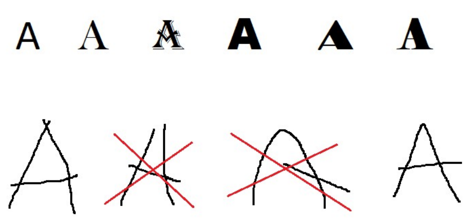
....

Learning

Learning



Learning



Learning

- Understanding the ideal or the model
  - Data or sample are instances of the ideals.
  - Each data is composed of a set of representative features / attributes / properties of objects or events.
  - Similarity of a sample to model identifies its class.
  - Emphasizing different attributes (features) leads to different types of classification.

Features or attributes

- An entity (object or event) is described by a set of features.
  - Features should be *representative* as well as *distinctive*.
  - Features should be *uncorrelated among themselves*.
- Example: Suppose each person of an academic institute is associated with certain features, e.g., age, height, weight, function, address, salary/stipend, education, experience
- Each feature may be used for different purposes.

Regression

- Regression is a technique to establish relation between in independent variables (features) and dependent variables.
- Features or independent variables are primary observations, measurements.
- Depending on type of of relation between dependent variable (*decision or inference*) and independent variable(s), we may classify the regression as
  - *Linear or non-linear regression*
  - *Logistic regression*
- There are other types of regressions such as *ridge regression, lasso regression*, etc.

Regression

- Linear or non-linear regression
  - **Dependent variable** is a random variable having **continuous value**.
  - Used in *predicting unknown values* given the input features (independent variables).
  - Regression model may be linear or non-linear such as higher-order polynomial, trigonometric, etc.
- Logistic regression
  - **Dependent variable** is a random variable having **discrete values**.
  - Used in *predicting class* of the object whose features are given as input.
  - Regression model may be represented as decision surface or decision boundary.

Learning

- Generating model (along with its parameters) through regression analysis is called *learning*.
- Determining the partition boundary between populations (classes)
  - Maximizing separation between instances of different populations (classes).
  - Minimizing dissimilarity (or maximizing similarity) of the instances of a class.
  - ***This is basic objective of classification.***

Learning

- Machine learning algorithms may be categorized as
  - *supervised learning*, and
  - *unsupervised learning*.
- Machine learning is essentially a form of applied statistics with
  - increased emphasis on estimating complicated predicting functions, and
  - decreased emphasis on proving confidential interval around these functions.

Idea of machine learning

- A system (here, machine or computer) is said to have learned
  - to do some task *T*
  - from a set of examples *E*
  - in terms of a performance measure *P*,

if its performance improves
  - as measured by the same *P*
  - to carry out the same task *T*
  - by dealing with the example set *E*.

- An example $\boldsymbol{x} \in R^n$ is a collection of features, each x_i is a feature measured objectively from application domain.

Performance *P*

- To evaluate the ability of machine learning algorithms quantitively.
- Performance measure is task specific.
  - Example: PSNR or SSIM for denoising task.
  - Example: Accuracy for classification tasks.
- Example set is divided into two parts: *Training set* and *test set*.
  - **Machine learns** from training set (used as experience).
  - ***Performance is evaluated*** on test set (unseen during training).

Machine learning tasks *T*

- **Classification with missing inputs:** Sometimes all elements of feature vector may not be available or known. Plausible solution could be

  (i) developing multiple functions for different sets of available features, and

  (ii) imputation of missing data → prediction of missing values.

Machine learning tasks *T*

- **Classification:** To decide which of the *k* classes the given input belongs to. Learning system tries to develop a mapping (function)

$$f:\ R^n \rightarrow \{1, 2, \cdots, k\}$$

- **Prediction (regression):** To predict a numerical value for the given input. So the task is similar to classification except the representation of output. Thus the mapping (function) is

$$f:\ R^n \rightarrow R$$

The machine learning framework

$$\mathbf{y} = f(\mathbf{x})$$

output prediction function feature

- **Training:** given a *set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, estimate the prediction function f by minimizing the prediction error on the set
- **Testing:** apply f to a never before seen *test example* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

Slide credit: L. Lazebnik

Structured Learning

Machine learning is to find a function *f*

$$f : X \rightarrow Y$$

Regression: output is a scalar

Classification: output is a "class label"

(one-hot vector)

1	0	0

Class 1

0	1	0

Class 2

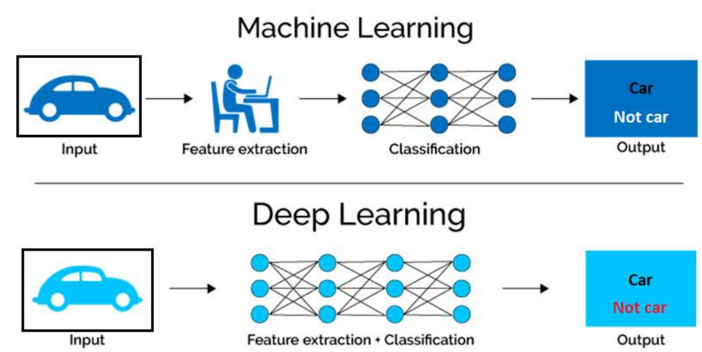
0	0	1

Class 3

Structured Learning/Prediction: outputs sequence, matrix, graph, tree ……

Output is composed of <u>components with dependency</u>

Machine learning vs. Deep learning



Machine learning network

Machine learning models for classification have followings are common:

- ***Input layer:*** quantitative representation of object features
- ***Hidden layer(s):*** apply transformations with nonlinearity
- ***Output layer:*** Result for classification, regression etc.

- The models are trained through ***supervised learning***.
  - Training data are explicitly labelled (known output).
  - Weights are updated to minimize error between prediction and the ground truth.

Linear regression

- Task is to build a system to predict a scalar value $y \in R$ as output from the given input $\boldsymbol{x} \in R^n$.
- Suppose $\hat{y}$ is the value predicted by the system, i.e.,

$$\hat{y} = \boldsymbol{w}^T \boldsymbol{x}$$

where $\boldsymbol{w} \in R^n$ is parameter vector that controls behaviour of system.

Linear regression (contd.)

- Learning process determines the value of $\boldsymbol{w}$ by minimizing the error

$$MSE_{train} = \frac{1}{m} ||\hat{y}^{(train)} - y^{(train)}||_2^2$$

- MSE_{train} depends on $\boldsymbol{w}$, so $\boldsymbol{w}$ can be obtained by

$$\nabla_w MSE_{train} = 0$$

known as ***normal equation***.

Linear regression (contd.)

- Given $(\boldsymbol{x} \in R^n, y \in R)$ pair related by $y = \boldsymbol{w}^T \boldsymbol{x}$, the solution of $\nabla_w MSE_{train} = 0$ is given by

$$\boldsymbol{w} = \left(\boldsymbol{X}^{(train)^T} \boldsymbol{X}^{(train)}\right)^{-1} \boldsymbol{X}^{(train)}\ \boldsymbol{Y}^{(train)}$$

where $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \cdots, \boldsymbol{x}_m]$ and $\boldsymbol{Y} = [y_1, y_2, y_3, \cdots, y_m]$

- **Example:** $(x \in R, y \in R)$ pair related by $y = wx$, the solution of w is given by

$$w = \frac{\sum_{i=1}^{m} x_i y_i}{\sum_{i=1}^{m} x_i^2}$$

Linear regression (contd.)

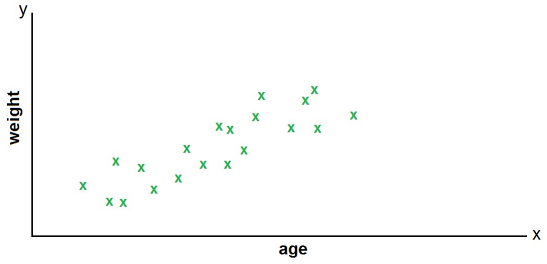
- A more general relation between $\boldsymbol{x} \in R^n$ and $y \in R$ may be expressed as

$$\hat{y} = \boldsymbol{w}^T \boldsymbol{x} + b$$

- If we append a '1' to $\boldsymbol{x}$ and including 'b' as a weight
  - Relation between y and $\boldsymbol{x}$ becomes affine, but
  - Relation between y and $\boldsymbol{w}$ remains linear.

Linear regression (contd.)

Example:



Linear regression (contd.)

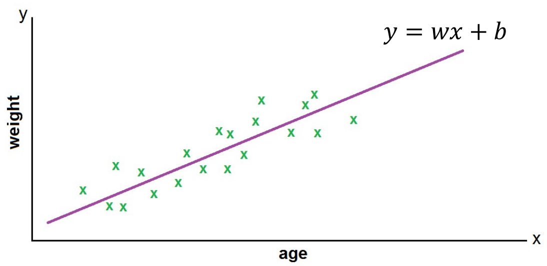
- x → independent variable (e.g., age of a deer, time in quarter, etc.)
- y → dependent variable (resp., weight of a deer, pairs of shoes sold)
- Let us consider relation between x and y may be modeled as a straight line:

$$y = wx + b$$

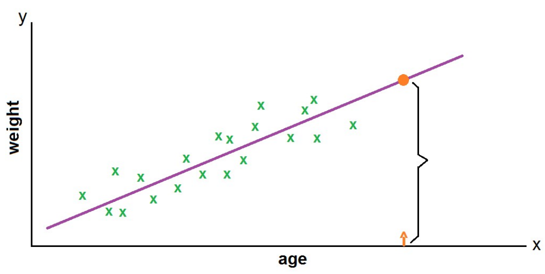
- Exploiting linear regression technique, we estimate

$$w = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sum(x-\bar{x})^2} \qquad b = \bar{y} - w\bar{x}$$

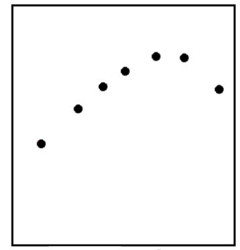
Linear regression (contd.)



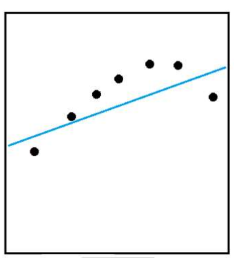
Prediction

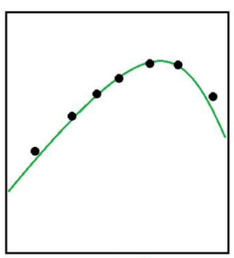


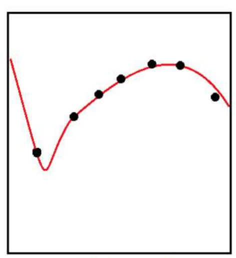
Regression


Data

Regression


Linear


Quadratic


Higher order

Prediction: multiple input

- So far we have discussed the cases where input is a single variable.

$$y = f(x)$$

- No. of input variables (independent variables) may be more than 1.

$$y = f(x_1, x_2, x_3, \dots , x_n)$$

- A contrived example may have following input variables:

Variable	1	2	3	4	5	6	7	8	9	10
x_1	37	42	38	34	41	42	36	40	39	43
x_2	95	93	97	96	98	98	94	97	99	95

Prediction: binary output

- Output is one of two distinct values.

Variable	1	2	3	4	5	6	7	8	9	10
x_1	37	42	38	34	41	42	36	40	39	43
x_2	95	93	97	96	98	98	94	97	99	95
y	0	0	0	0	1	1	0	1	1	1

- Represents a decision making with two options OR a binary classification problem.
- Imagine: x_1 is temperature, x_2 is humidity and y denotes the decision whether carry an umbrella (y=1) or not (y=0).

Prediction: Logistic regression

- We first compute corresponding output

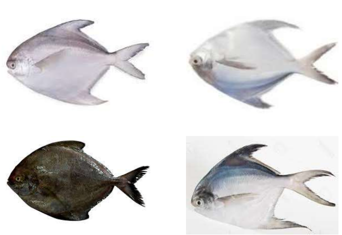
$$z = f(x_1,\ x_2) = b + a_1x_1 + a_2x_2$$

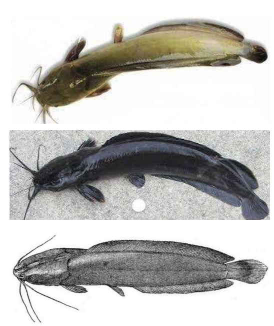
- Output is transformed to probability by means of a logistic function

$$Prob. = l(z)$$

- $Prob.$ indicates the prediction of default option (carrying umbrella) or, in other words, predicts raining.
- Thus if $Prob. > thres$ we choose default option; otherwise negation.
- Parameters a_i, b are estimated by maximum likelihood method to satisfy observed (training) data.

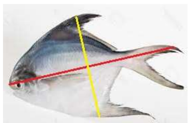
Binary classification: Pomfret and Magur

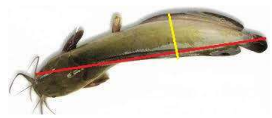




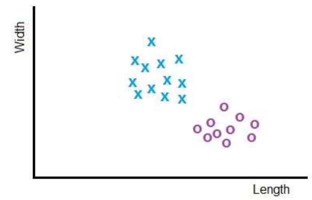
Two class problem: Pomfret and Magur

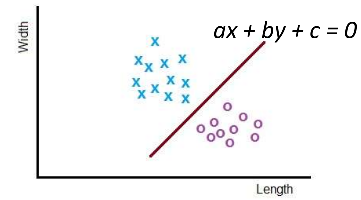
Features: Pomfret and Magur





Two-class problem: Feature space





Boundary function

- Find coefficients a, b and c of equation of a straight line

$$ax + by + c = 0$$

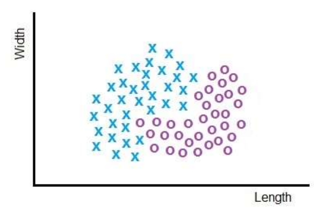
such that for all observation a feature pair (x, y):

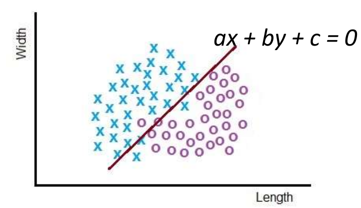
$ax + by + c > 0$ if (x,y) belongs to C_1

$ax + by + c < 0$ if (x,y) belongs to C_2

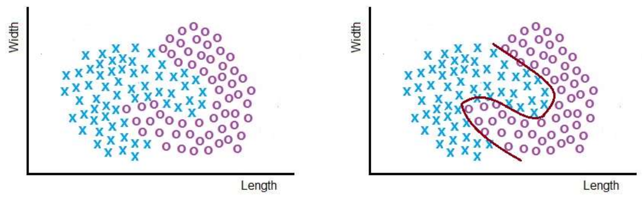
- If the desired condition is not satisfied for any feature pair we call a classification error has occurred.
- In general, decision boundary must be estimated to minimize this error.

Two-class problem





Two-class problem



Overfitting and underfitting

- The ability to perform well on previously unseen data is called ***generalization***.
- The target of machine learning to keep ***generalization error*** or ***test error*** as low as possible.
  - Note that system is built by minimizing the train error.
  - Is there any relation between training error and test error?

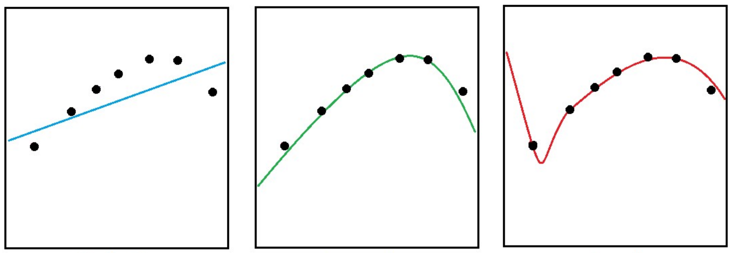
Overfitting and underfitting (contd.)

- Training and test data are accumulated by same data generating process.
  - Each example in training and test datasets are ***independent*** to each other.
  - The training and test datasets are ***identically distributed***.
- The *i.i.d.* assumption allows us to study the relationship between the training error and the test error.
  - Expected training error and the expected test error of a model are equal.

Overfitting and underfitting (contd.)

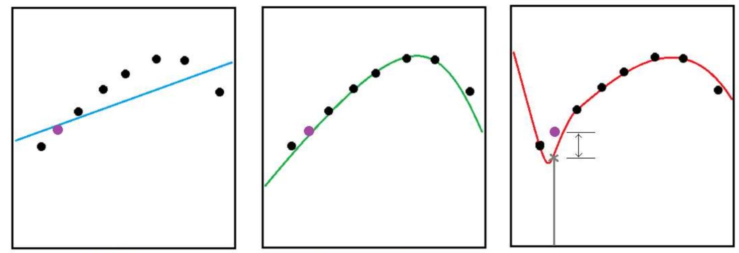
- ***Two criteria*** that determines how well a machine learning algorithm performs are its ability to
  1. make the training error small, and
  2. Make the gap between the training error and the test error small.
- These correspond to two problems: ***overfitting*** and ***underfitting***.
  - If the training error is not small → underfitting
  - If gap between training and test errors is not small → overfitting.

Overfitting and underfitting (contd.)



Underfit **Appropriate** **Overfit**

Overfitting and underfitting (contd.)



Underfit **Appropriate** **Overfit**

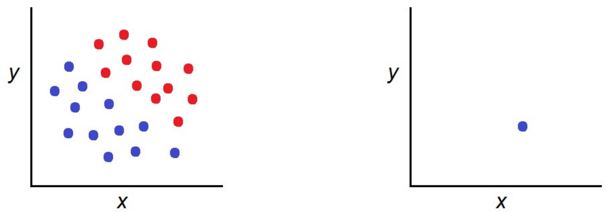
How to set the boundary function

- Based on the training data set.
  - All at a time.
    - Linear discriminant analysis
  - One at a time.
    - Perceptron network, neural network

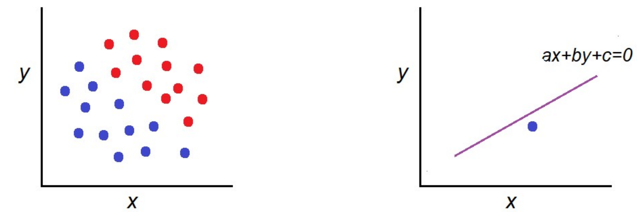
How to set the boundary function

- Based on the training data set.
  - All at a time.
    - Linear discriminant analysis
  - One at a time.
    - Perceptron network, neural network

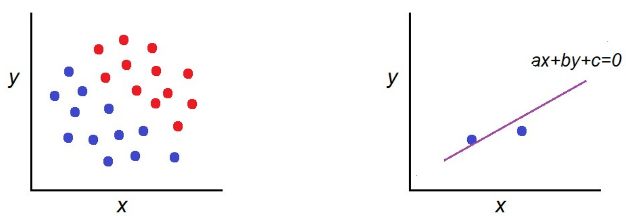
Forming the decision boundary



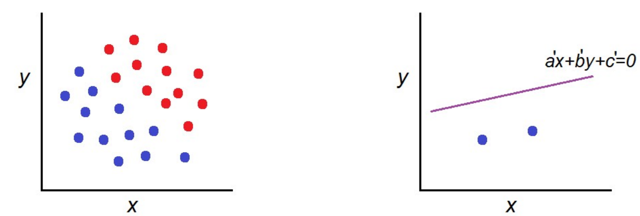
Forming the decision boundary



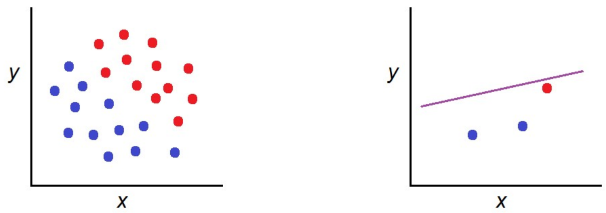
Forming the decision boundary



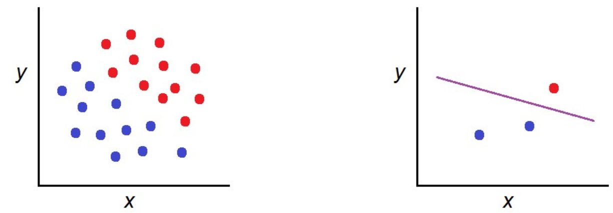
Forming the decision boundary



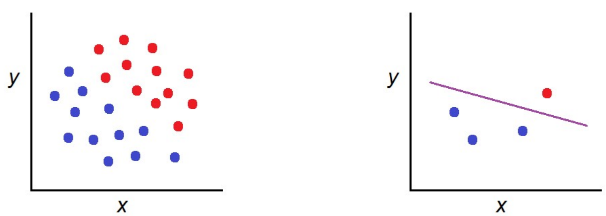
Forming the decision boundary



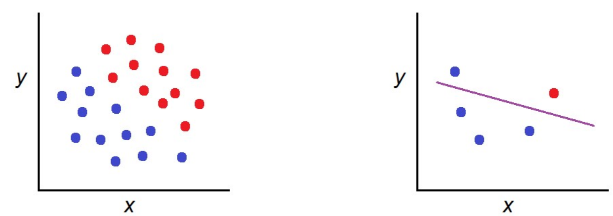
Forming the decision boundary



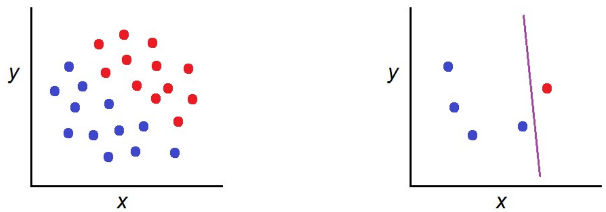
Forming the decision boundary



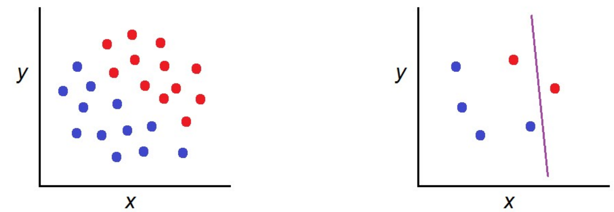
Forming the decision boundary



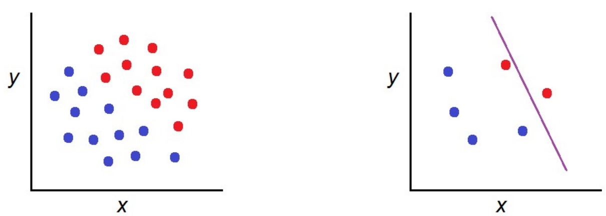
Forming the decision boundary



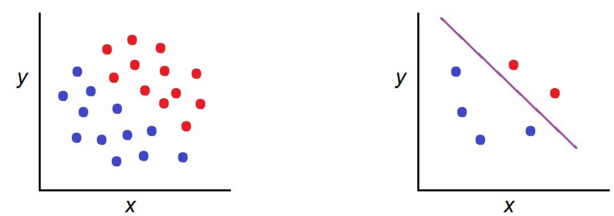
Forming the decision boundary



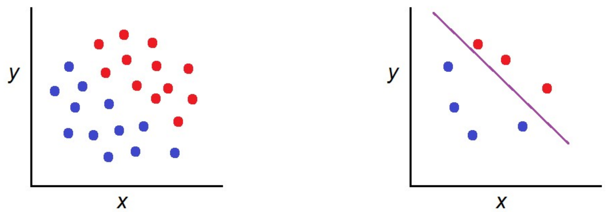
Forming the decision boundary



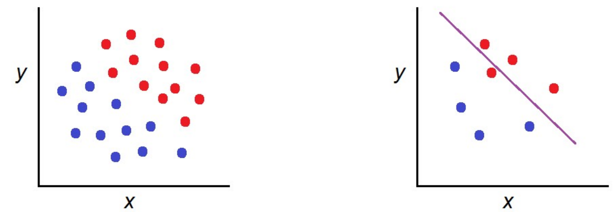
Forming the decision boundary



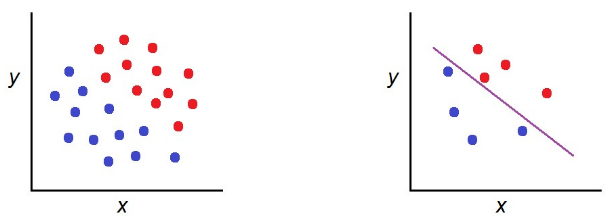
Forming the decision boundary



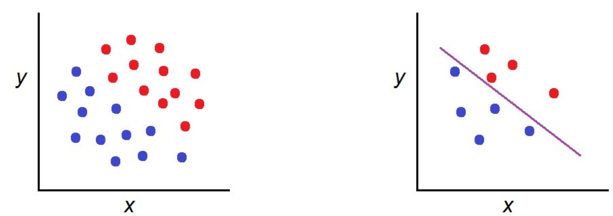
Forming the decision boundary



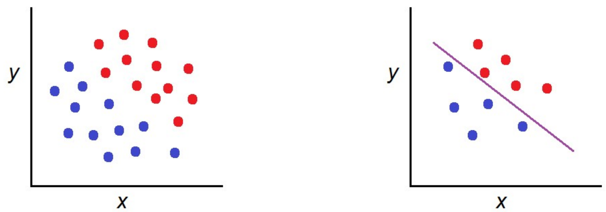
Forming the decision boundary



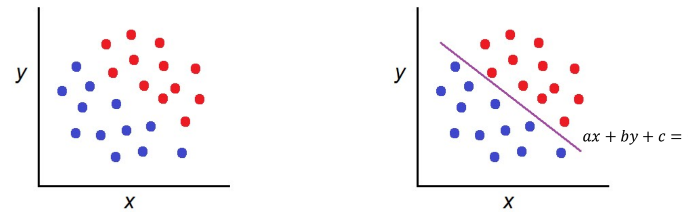
Forming the decision boundary



Forming the decision boundary



Forming the decision boundary



Boundary function

- Find coefficients *a*, *b* and *c* of equation of a straight line

$$ax + by + c = 0$$

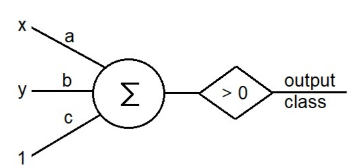
such that for all observation a feature pair (x, y):

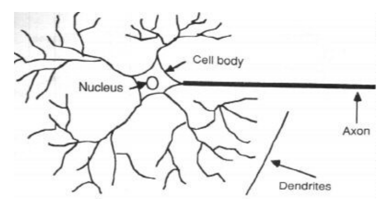
$ax + by + c > 0$ if (x,y) belongs to C_1

$ax + by + c < 0$ if (x,y) belongs to C_2

- If the desired condition is not satisfied for any feature pair we call a classification error has occurred.
- In general, decision boundary must be estimated to minimize this error.

Linear classifier and neuron



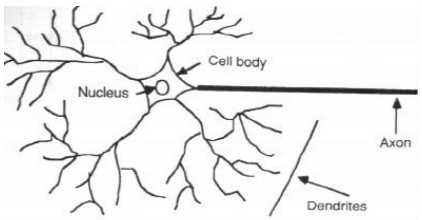


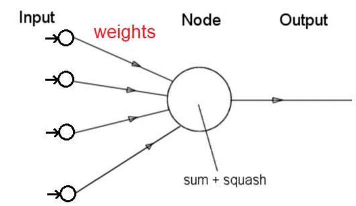
What are Artificial Neural Networks?

- Mimics the function of the brain and nervous system
- Highly parallel
  - Process information much more like the brain than a serial computer
- Learning

- Very simple principles
- Very complex behaviours

Neuron versus Node





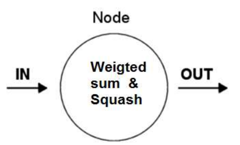
Function of a node

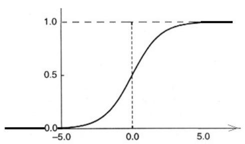
- At node

  Output $O = f(\sum w_i x_i)$

where $f(.)$ is a squashing function.

- Squashing function limits node output.





Neural Networks: History

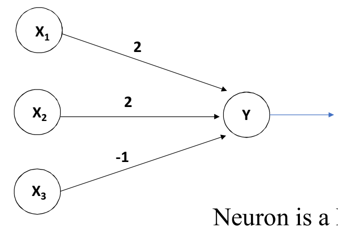
- McCulloch & Pitts (1943) are generally recognised as the designers of the first neural network
- Many of their ideas still used today (e.g. many simple units combine to give increased computational power and the idea of a threshold)

- 1949-First learning rule
- 1969-Minsky & Papert - perceptron limitation - Death of ANN
- 1980's - Re-emergence of ANN - multi-layer networks

Theory of Back Propagation Neural Net (BPNN)

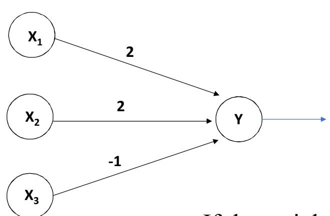
- Use many samples to train the weights (W), so it can be used to classify an unknown input into different classes
- Will explain
  - How to use it after training: forward pass (classify /or the recognition of the input)
  - How to train it: how to train the weights and biases (using forward and backward passes)

The First Neural Networks



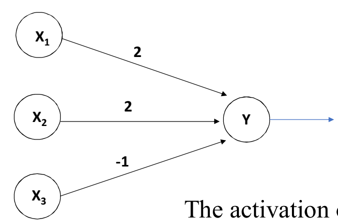
Neuron is a McCulloch-Pitts network are connected by directed, weighted paths.

The First Neural Networks



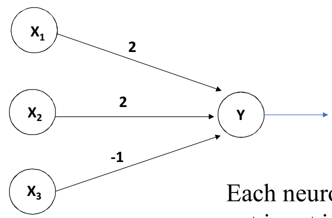
If the weight on a path is positive the path is excitatory, otherwise it is inhibitory.

The First Neural Networks



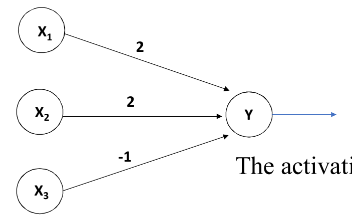
The activation of a neuron is binary. That is, the neuron either fires (activation of one) or does not fire (activation of zero).

The First Neural Networks



Each neuron has a fixed threshold. If the net input into the neuron is greater than the threshold, the neuron fires.

The First Neural Networks



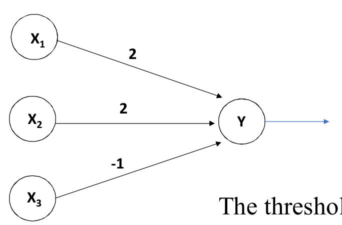
The activation function for unit Y is

$$f(Y) = \begin{cases} 1, & \text{if } Y >= \theta \\ 0, & \text{otherwise} \end{cases}$$

where Y is the total input signal received

θ is the threshold for Y.

The First Neural Networks

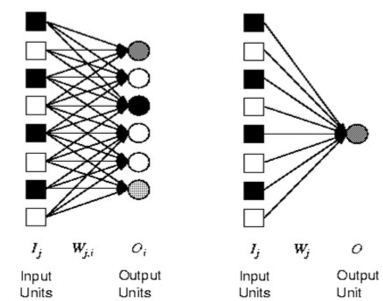


The threshold is set such that any non-zero inhibitory input will prevent the neuron from firing.

Perceptron network

- Synonym for single layer, feed-forward network capable of learning.
- Output $O = f\left(\sum_j W_j I_j + b_j\right)$

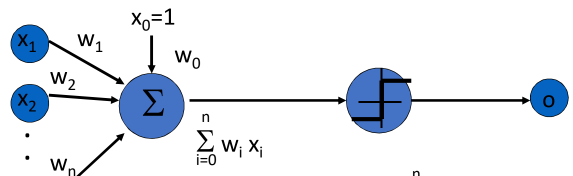
where 'b' is bias, which however, may be included as additional weight.



Perceptron Network **Single Perceptron**

Perceptron

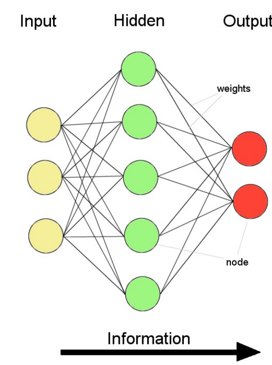
- Linear treshold unit (LTU)



$$o(x_i)=\begin{cases} 1 & \text{if } \sum_{i=0}^{n} w_i I_i > 0 \\ -1 & \text{otherwise} \end{cases}$$

Feed-forward nets

- Information flow is unidirectional
  - Data is presented to *Input layer*
  - Passed on to *Hidden Layer*
  - Passed on to *Output layer*
- Information is distributed
- Information processing is parallel
- True while testing new data



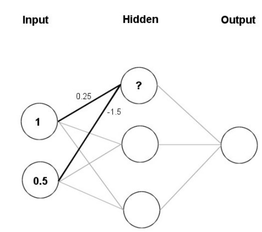
Standard activation functions

- The hard-limiting threshold function
  - Corresponds to the biological paradigm
    - either fires or not ***(Perceptron)***
- Sigmoid functions ('S'-shaped curves)
  - The hyperbolic tangent (symmetrical)
  - Both functions have a simple differential
  - Only the shape is important ***(Neuron)***

$$\phi(x) = \frac{1}{1+e^{-ax}}$$

Example: node function

- Feeding data through the net:



$(1 \times 0.25) + (0.5 \times (-1.5)) = 0.25 + (-0.75) =$ **- 0.5**

Squashing: $\frac{1}{1+e^{0.5}} = 0.3775$

Data

- Input data is presented to the network in the form of activations in the input layer
- Examples
  - Pixel intensity (for pictures)
  - Share prices (for stock market prediction)
- Data usually requires pre-processing
  - Analogous to senses in biology
- How to represent more abstract data, e.g. a label?
  - Choose a pattern, e.g., 0-0-0-0-1-0-0-0-0-0 for "digit 5"

Loss function or Error or Cost function

- Training sample is composed of
  - Input data (feature vector) and
  - Actual class label (also known as groundtruth)
- Given the input, feed forward network predicts class label
  - based on current parameters
  - Loss or error or cost is measured as total deviation from groundtruth

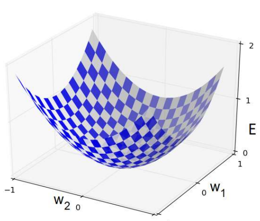
Cost or Loss or Error: $E(\boldsymbol{w}) = \sum(\textit{Predicted label} - \textit{Actual label})^2$

where $\boldsymbol{w}$ is parameter vector.

Training the network

- Means setting correct weights (including bias) or parameters of the network.
- Backpropagation
  - Requires training set (input / output pairs)
  - Starts with small random weights
  - Compute error between predicted label and actual label (groundtruth)
  - Error is used to adjust weights (supervised learning)

  → Gradient descent on error landscape

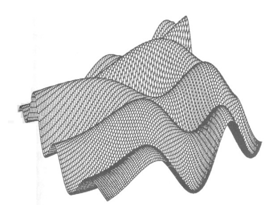


$E(\boldsymbol{w})$

Training the network

- Means setting correct weights (including bias) or parameters of the network.
- Backpropagation
  - Requires training set (input / output pairs)
  - Starts with small random weights
  - Compute error between predicted label and actual label (groundtruth)
  - Error is used to adjust weights (supervised learning)

  → Gradient descent on error landscape



$E(\boldsymbol{w})$

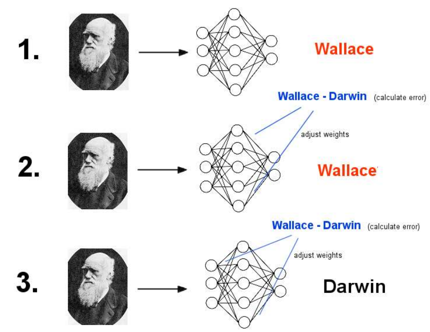
Maths: Weight setting by gradient descent

- Error function: $E(w) = \frac{1}{2n}\sum_x \| y(x,w) - a_x \|^2$
- A small change in error E may be given by

$$\Delta E \approx \frac{\partial E}{\partial w_1}\Delta w_1 + \frac{\partial E}{\partial w_2}\Delta w_2 = \left(\frac{\partial E}{\partial w_1} \quad \frac{\partial E}{\partial w_2}\right)^T \cdot (\Delta w_1 \quad \Delta w_2)^T = \nabla E \cdot \Delta w$$

- Let $\Delta w = -\eta \nabla E$ which implies $\Delta E = -\eta \| \nabla E \|^2 \leq 0$
- This suggests updating weights as $w_k^{(t+1)} = w_k^{(t)} - \eta \frac{\partial E}{\partial w_k}$

Training the network: Example



Different Non-Linearly Separable Problems

Structure	*Types of Decision Regions*	*Exclusive-OR Problem*	*Classes with Meshed regions*	*Most General Region Shapes*
Single-Layer	*Half Plane Bounded By Hyperplane*	A B B A	B A	
Two-Layer	*Convex Open Or Closed Regions*	A B B A	B A	
Three-Layer	Arbitrary (Complexity Limited by No. of Nodes)	A B B A	B A	

Thank you!

Any question?