Specialist Programme on Artificial Intelligence for IT & ITES Industry

Pattern Classification and Prediction (Cont.)

By *Dr Zhu Fangming*
fangming@nus.edu.sg

Singapore e-Government Leadership Centre
National University of Singapore

Inspire *Lead* *Transform*

Pattern Recognition Using Supervised Learning Techniques (II)

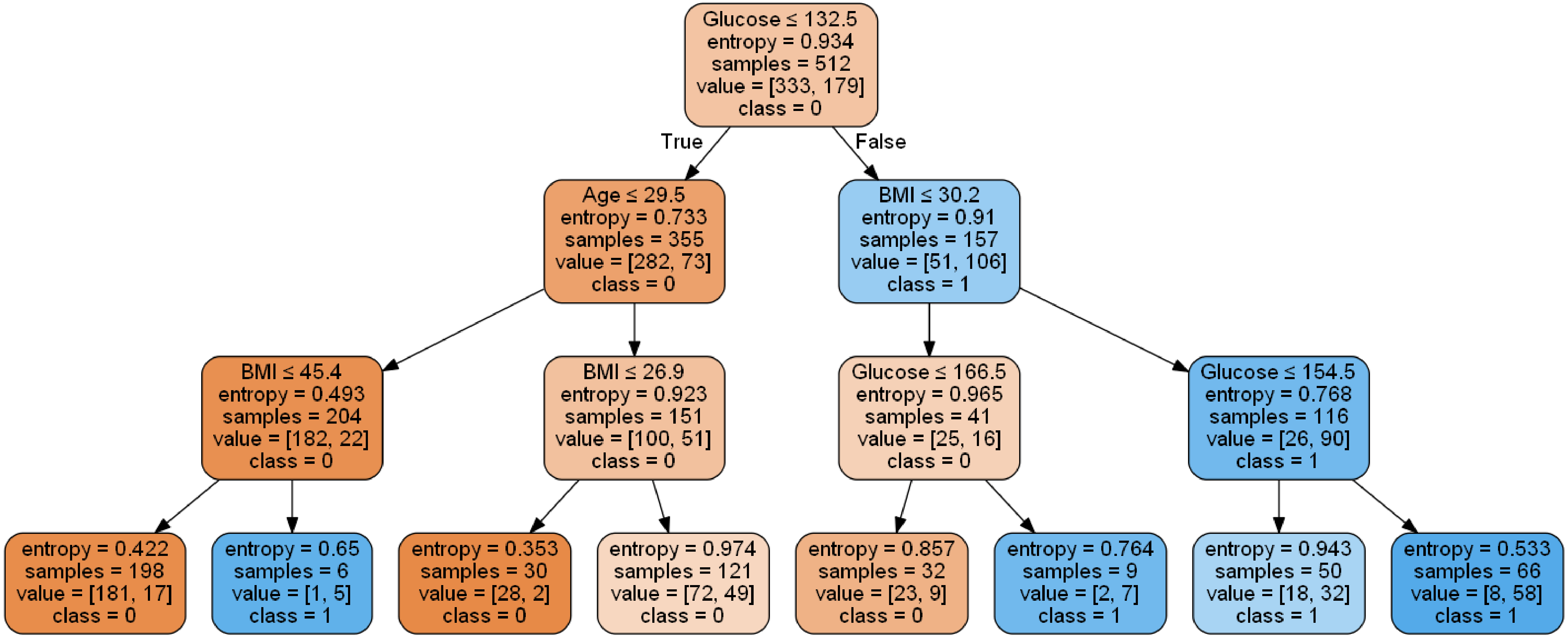
Supervised Learning Techniques (II)

- Decision Trees (DT)
- Neural Networks (NN)
- Support Vector Machines (SVM)

Decision Tree

- A decision tree is a flow-chart-like tree structure.
  - An internal node performs a test on an attribute
  - A branch represents a result of the test
  - A leaf node represents a class label
  - At each node, one feature is chosen to split training examples into distinct classes
  - A new sample is classified by following a matching path to a leaf node

Decision Tree

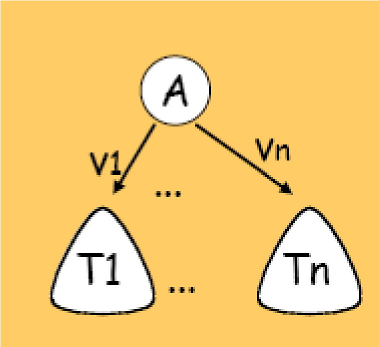


Applications of Decision Trees

- Customer Relationship Management
- Fraud Detection
- Churn Prediction
- Credit Risk Prediction
- Purchasing Behavior Prediction
- Fault Detection
- Sentiment Analysis
- Investment Solutions

Basic Algorithm: Quinlan's ID3/C4.5/C5.0

- create a root node for the tree
- if all examples from S belong to the same class Cj
- then label the root with Cj
- else
    - select the "most informative" attribute A with values v1, v2, , vn
    - divide the training set S into S1, …, Sn according to values v1,…,vn
    - recursively build subtrees T1,…,Tn for S1,…,Sn
    - generate decision tree T



Building Decision Tree

- Top-down tree construction
  - At start, all training data are at the root.
  - Partition the examples recursively by choosing one feature each time.
- At each node, available attributes are evaluated on the basis of separating the classes of the training examples. A goodness function is used for this purpose.
- Typical goodness measures:
  - Information gain (ID3/C4.5)
  - Information gain ratio (C4.5)
  - Gini index (CART)

Heuristic Search

- Search bias: Search the space of decision trees from simplest to increasingly complex (greedy search, no backtracking, prefer small trees)
- Search heuristics: At a node, select the attribute that is most useful for classifying examples, split the node accordingly

Stopping Criteria

- if all examples belong to same class C_j, label the leaf with C_j
- if all attributes were used, label the leaf with the most common value C_k of examples in the node
- **min_samples_split** - The minimum number of samples required to split an internal node.
- **min_samples_leaf** - The minimum number of samples required to be at a leaf node
- **max_depth -** The maximum depth of the tree.
- …

Overfitting and Tree Pruning

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Pre-pruning (forward pruning): stop growing the tree e.g.
    - When data split not statistically significant
    - Too few examples are in a split
  - Post-pruning: *Remove branches* from a "fully grown" tree — get a sequence of progressively pruned trees
    - Use a set of validation data to decide which is the "best pruned tree"

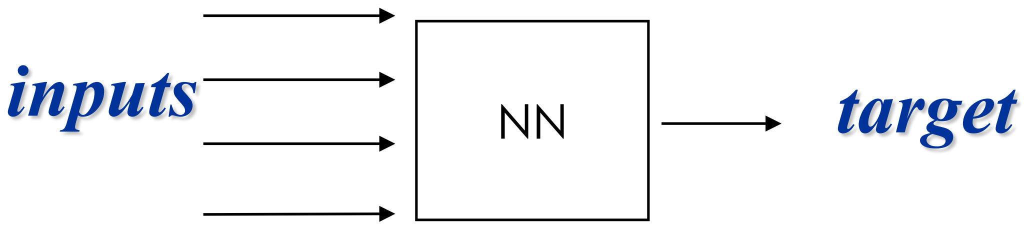
Pros and Cons of Decision Trees

- Pros:
  - simple to understand and interpret
  - little data preparation and little computation
  - indicates which attribute are most important for classification

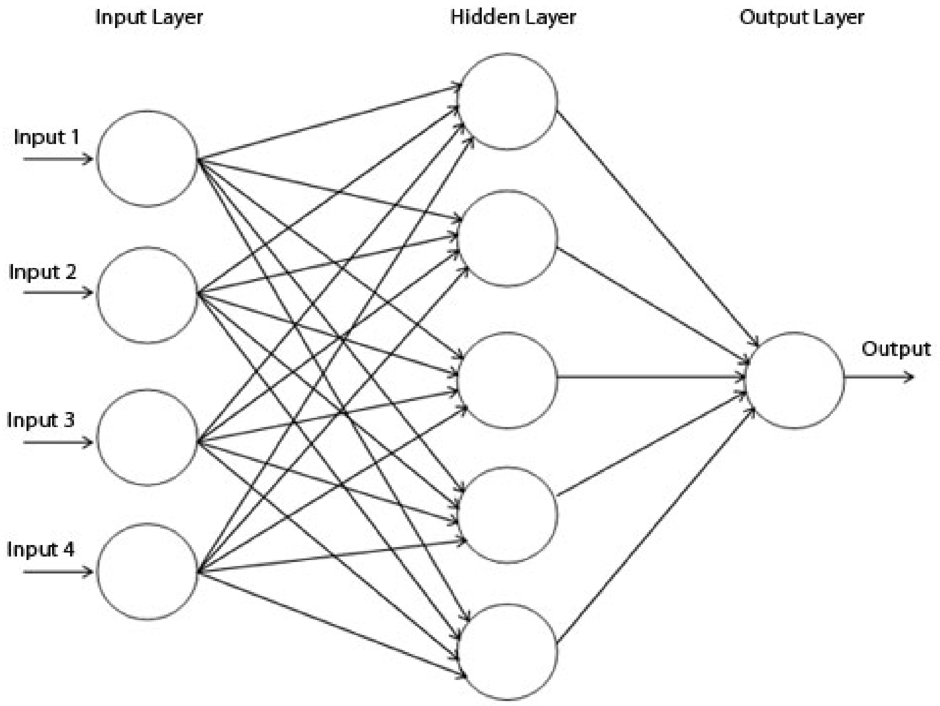
- Cons:
  - not guaranteed to produce an optimal decision tree
  - perform poorly with many classes and small data
  - over-complex trees do not generalise well from the training data (overfitting)

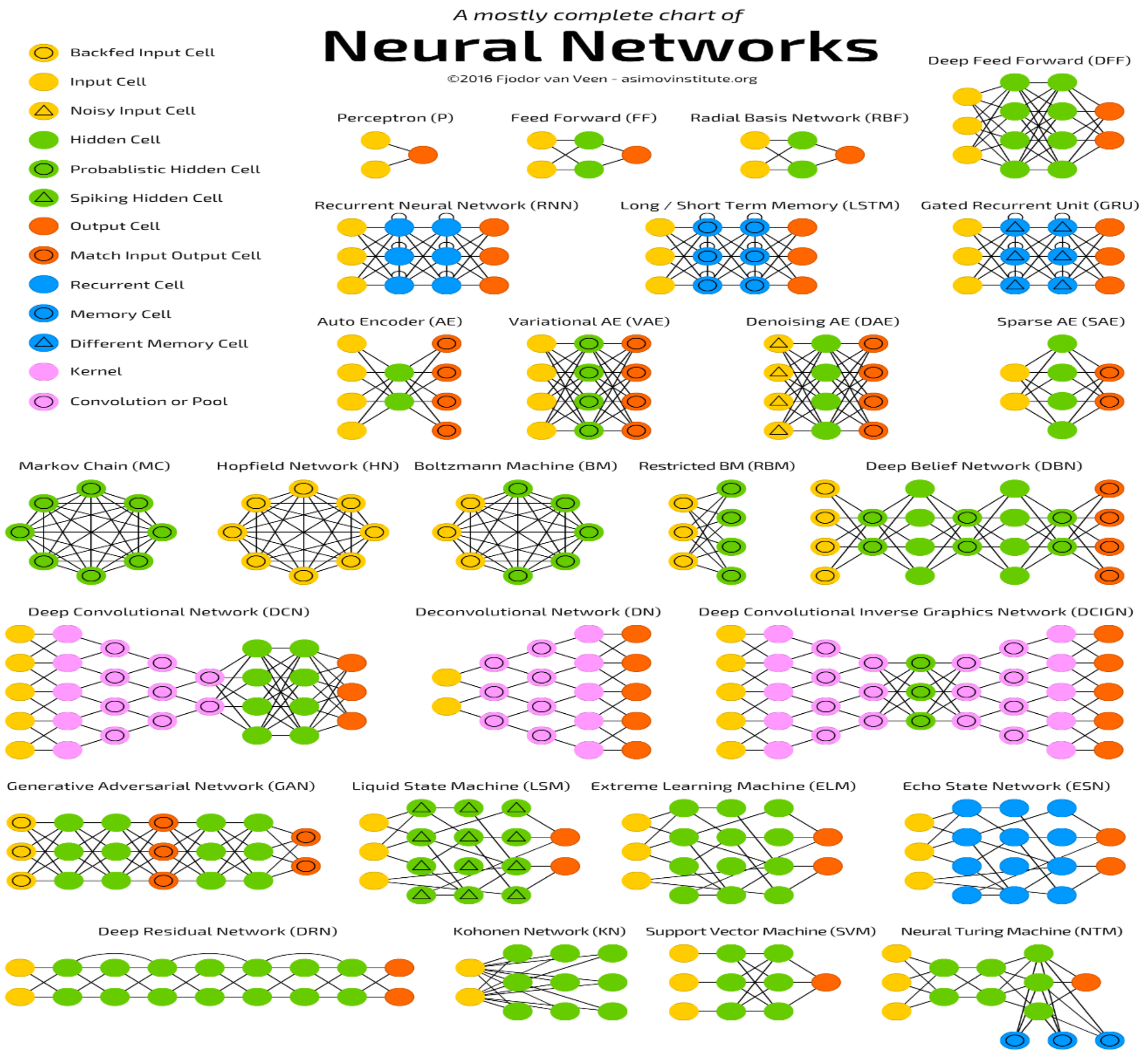
Neural Networks

- Neural Networks (NN) are biologically inspired and attempt to build computational models that operate like a human brain.
- These networks can “learn” from the data and recognize patterns.
- Make no assumptions about the data
- Can be very accurate
- Handle both numeric targets and categorical targets
- A black box….



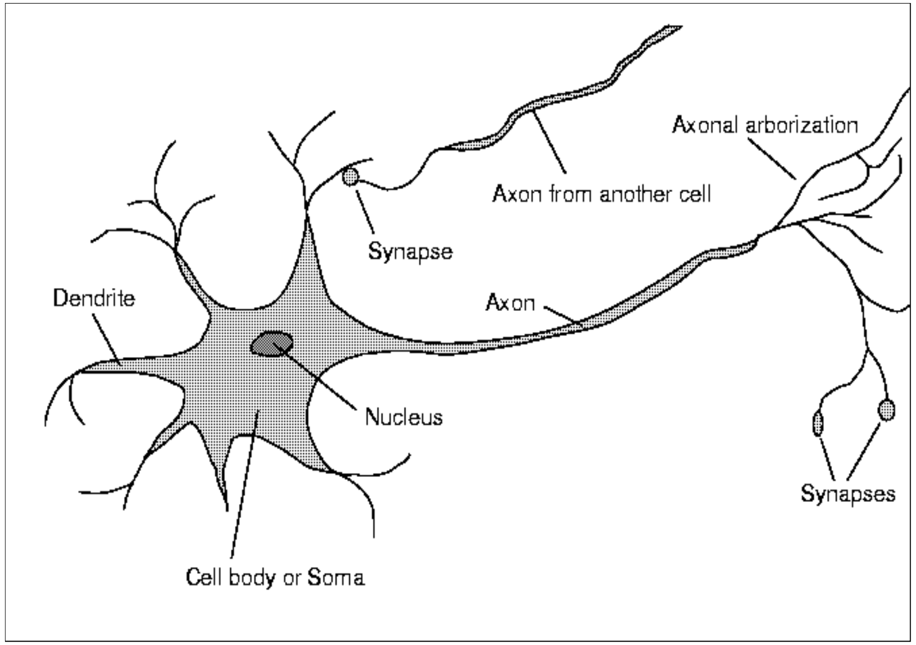
Neural Networks

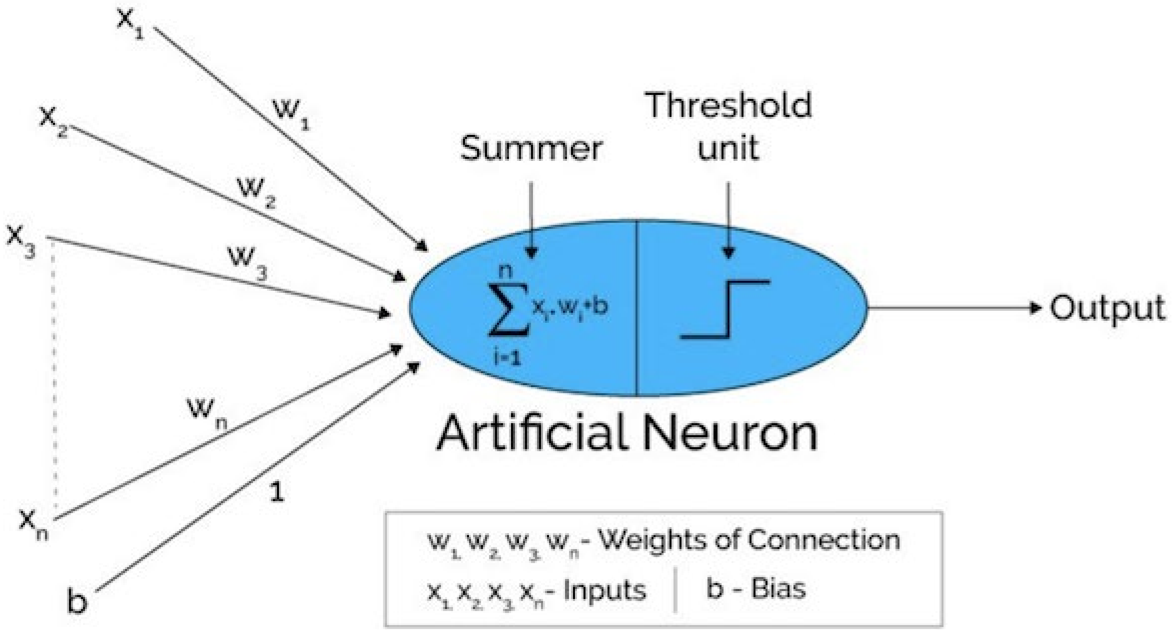




http://www.asimovinstitute.org/neural-network-zoo/

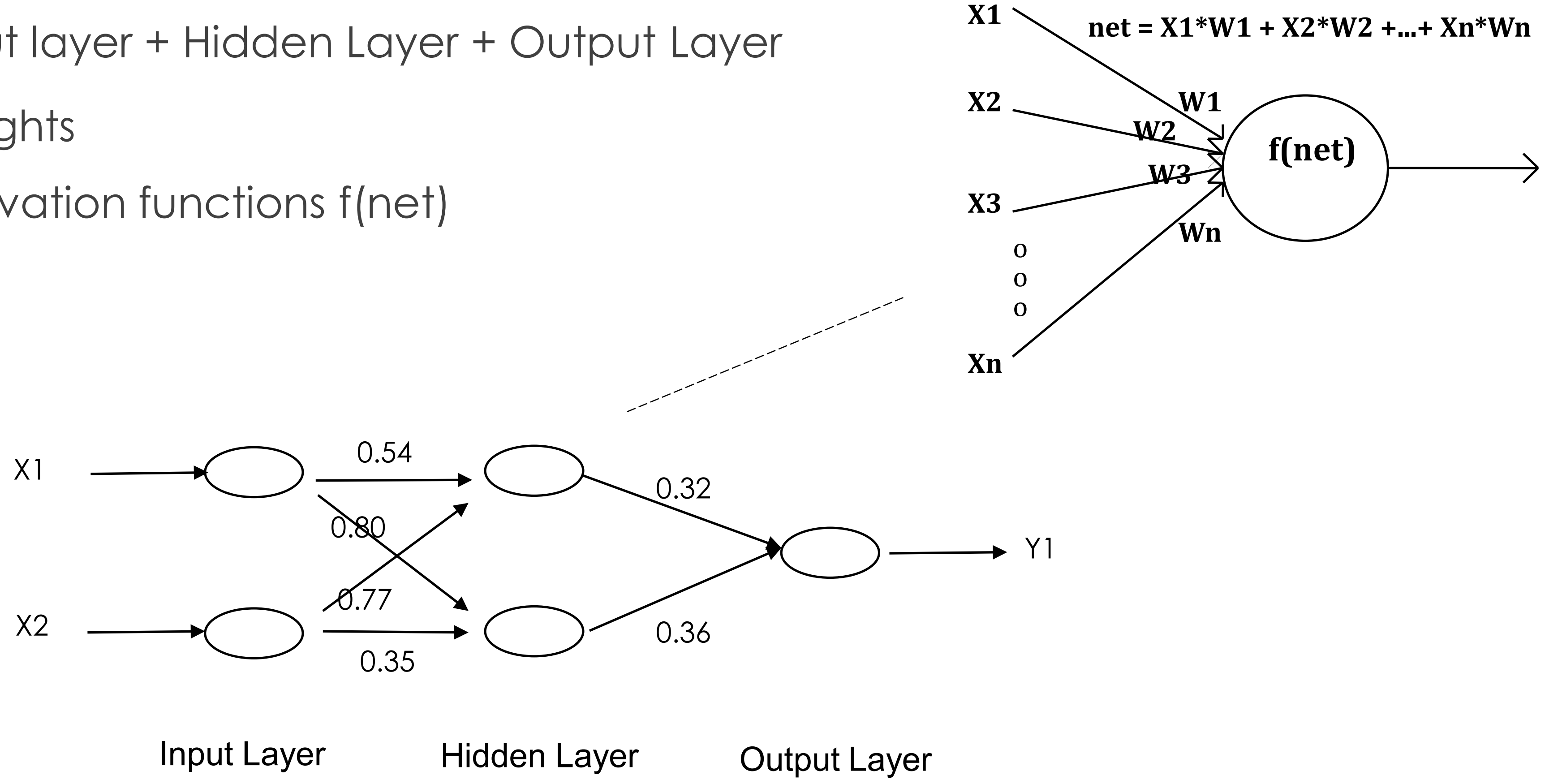
From Biological Neuron to Artificial Neuron





General Architecture of Neural Networks

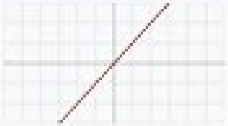
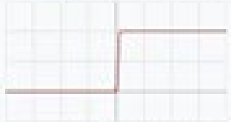
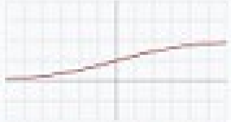
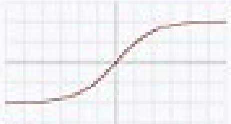
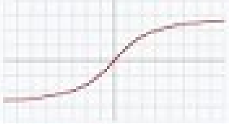
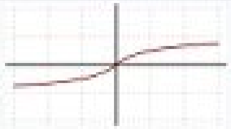
- Framework (in general, but not for all NNs)
  - Input layer + Hidden Layer + Output Layer
  - Weights
  - Activation functions f(net)



General Architecture of Neural Networks (cont.)

- Weights
  - Normally initial weights are randomised to small real numbers
- Learning rule
  - determine how to adapt connection weights in order to optimise the network performance $W_i(t+1)=W_i(t)+\Delta W_i(t)$
  - indicate how to calculate the weight adjustment during each training cycle
- Activation calculation & Weight adjustment
  - Compute the activation levels across the network
  - Weight adjustment based on the errors /distance

Activation functions

Name	Plot	Equation	Derivative (with respect to x)
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for} \quad x \neq 0 \\ ? & \text{for} \quad x = 0 \end{cases}$
Logistic (a.k.a. Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Softsign [7][8]		$f(x) = \frac{x}{1 + \lvert x \rvert}$	$f'(x) = \frac{1}{(1 + \lvert x \rvert)^2}$

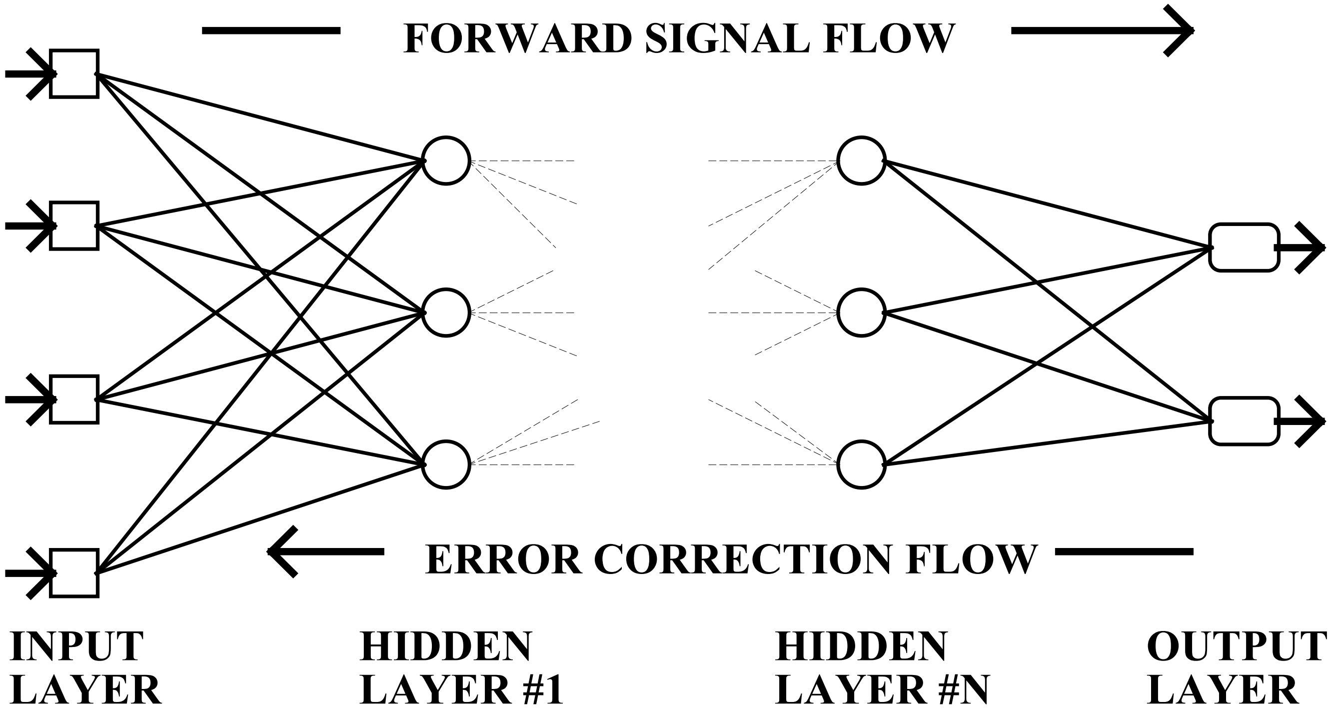
https://www.codeproject.com/Articles/1200392/Neural-Network

Training Neural Networks

- Require lots of training data
- Training can be slow!
- Limit training by
  - the time taken
  - number or training iterations
  - the accuracy

Multilayer Perceptron (MLP) with Backpropagation Learning

- Propagate signals forward and then errors backward
- Backpropogation (BP) ~ gradient descent learning
- Weights in hidden layers are adjusted to reduce aggregate errors in the output layer

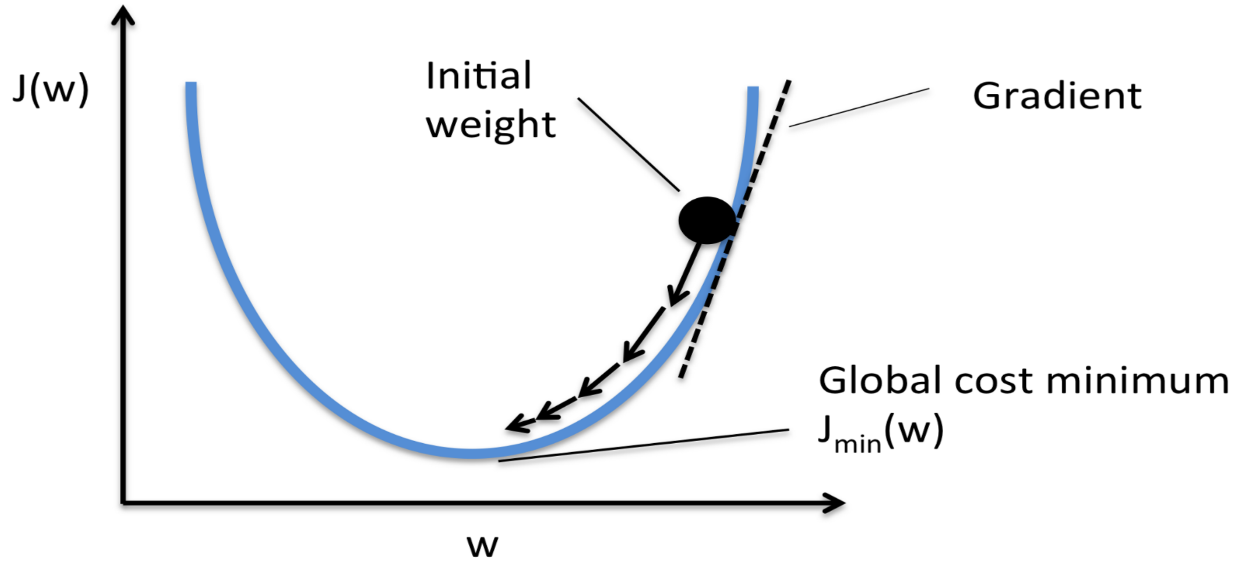


Gradient Descent Learning

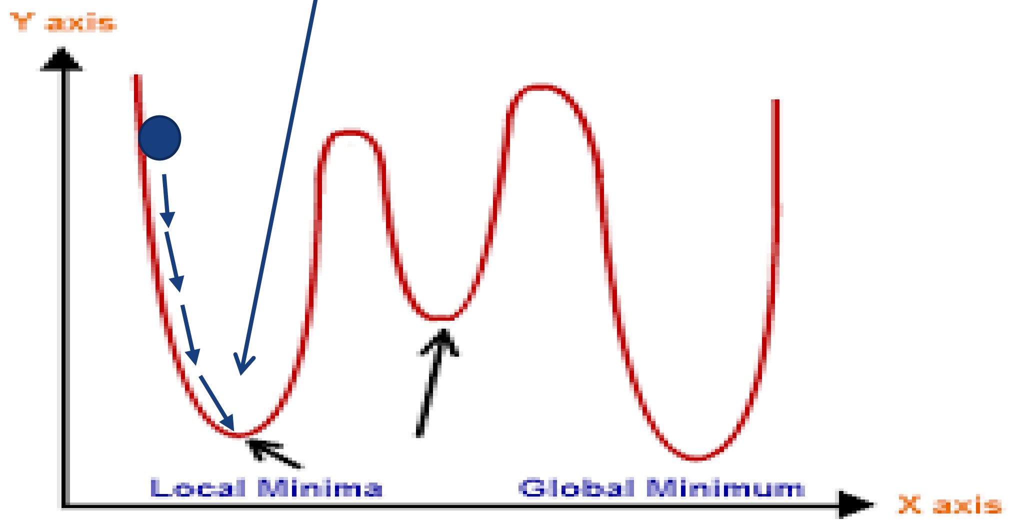
$$\Delta w_{ji}(t+1) = -\eta \frac{\partial E}{\partial w_{ji}(t)} + \alpha \Delta w_{ji}(t)$$

Leaning rate

Momentum rate

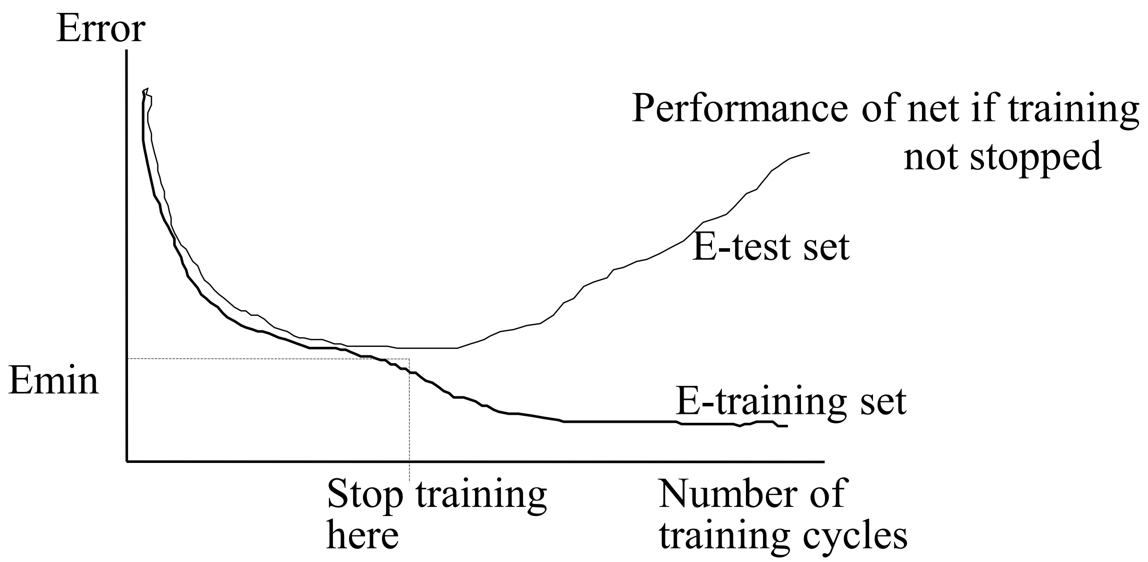


Get stuck into local minimum



Generalization & Overtraining /Overfitting

- *Generalization* is the ability of a network to correctly classify a pattern it has not seen (not been trained on). NNs generalize when they recognize patterns not previously trained on or when they predict new outcomes from past behaviors.
- Networks can be *overtrained*. It means that they memorize the training set and are unable to generalize well.

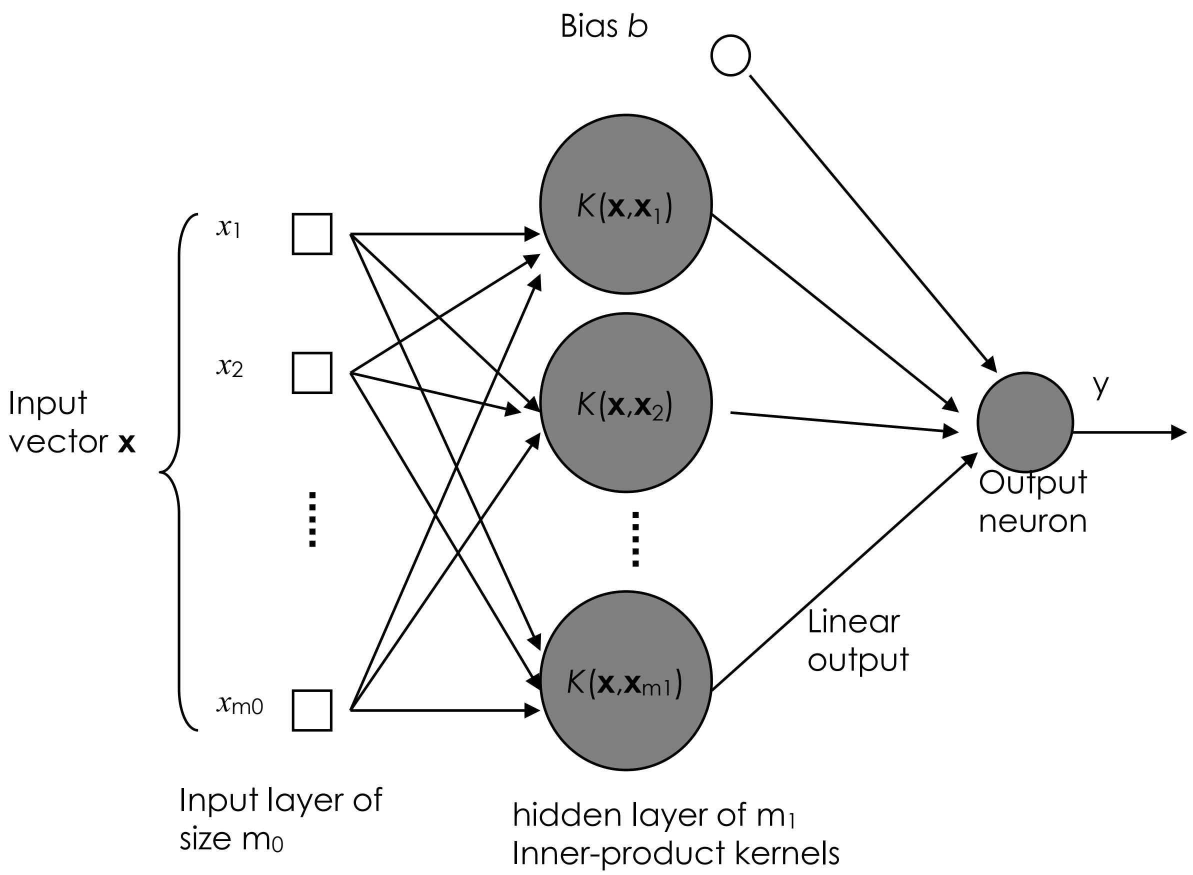


Applications of Neural Networks

- Image processing / Computer vision
- Natural language processing
- Data visualization
- Fault diagnosis
- Forecasting time series
- General mapping
- …

Support Vector Machines (SVM)

- Another category of feed forward networks [Vapnik, 1992, 1995, 1998]
- SVM can be used for pattern classification and non-linear regression – but uses statistical learning theory
- General architecture of a support vector machine
  - **Input layer**
  - **Hidden layer of Inner-product kernels (fully connected with the input layer)**
  - **Output neuron**



Support Vector Machines (SVM)

- For nonlinear problem, it uses a nonlinear mapping to transform the original training data into a higher dimension
- With the new dimension, it searches for the linear optimal separating hyperplane
- SVM finds this hyperplane using support vectors ("essential" training tuples) and margins (defined by the support vectors)
- Training can be slow but accuracy is high owing to their ability to model complex nonlinear decision boundaries (margin maximization)
- Applications:
    - handwritten digit recognition, object recognition, speaker identification, ...

SVM: Optimal Hyperplane & Support Vector

- Important concepts from the theoretical background
  - *Optimal hyperplane* for separable or non-separable patterns
  - *Support vector*
- A training pattern can be represented as a *vector* from the problem space
- Consider a group of training patterns
  - Training samples: $\{(\mathbf{x}_i, y_i)\}$ $i = 1, 2, \ldots, N$

    $\mathbf{x}_i$: the input pattern for the *i*-th example

    $y_i \in \{-1,1\}$): the corresponding desired output
  - The decision surface for the separation is a hyperplane

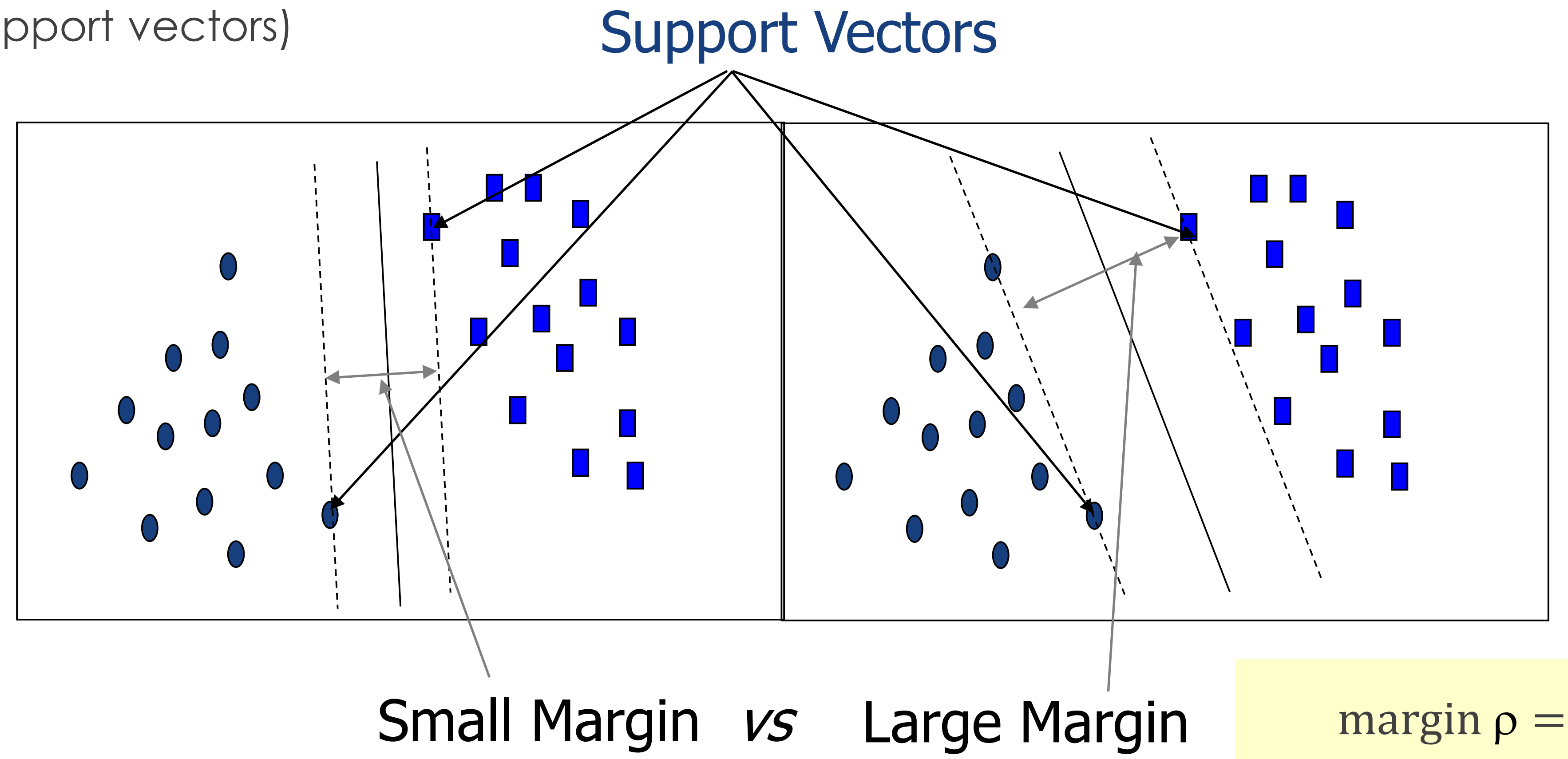
    $\mathbf{w}^T\mathbf{x} + b = 0$ (e.g. $w_1x_1 + w_2x_2 + \ldots + w_Nx_N + b = 0$)

    i.e. $\mathbf{w}^T\mathbf{x} + b \geq 0$ for $y_i = 1$

    $\mathbf{w}^T\mathbf{x} + b < 0$ for $y_i = -1$

SVM : Separation Margin & Support Vector

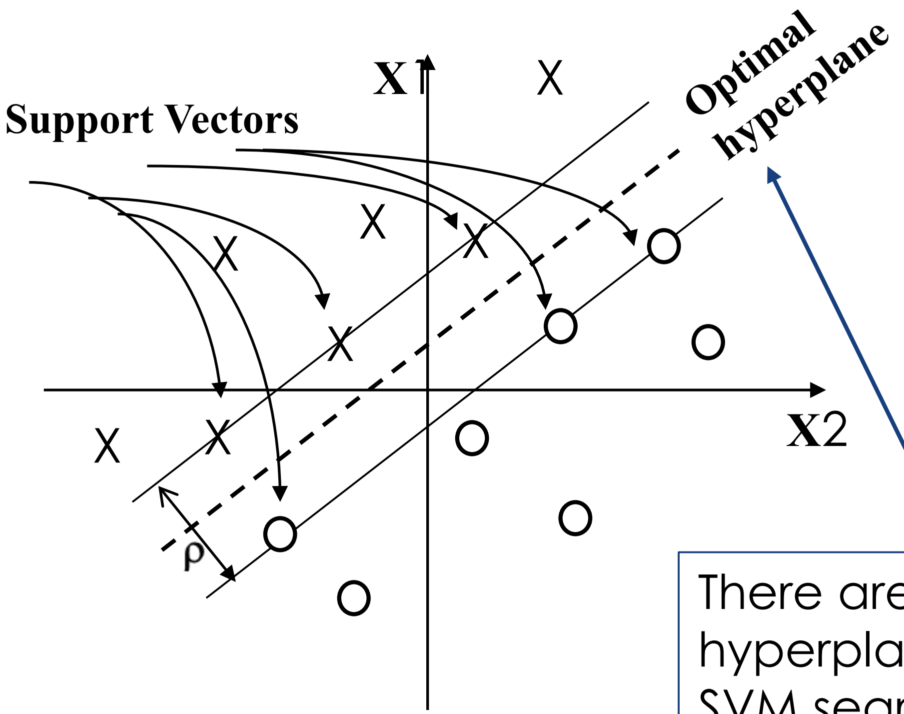
- *Margin of separation*
  - The separation between the decision surface hyperplane and the closest data points (support vectors)



$$\text{margin } \rho = \frac{2}{||\mathbf{w}||}$$

Hard Margin Linear SVM : Optimal Hyperplane & Support Vector

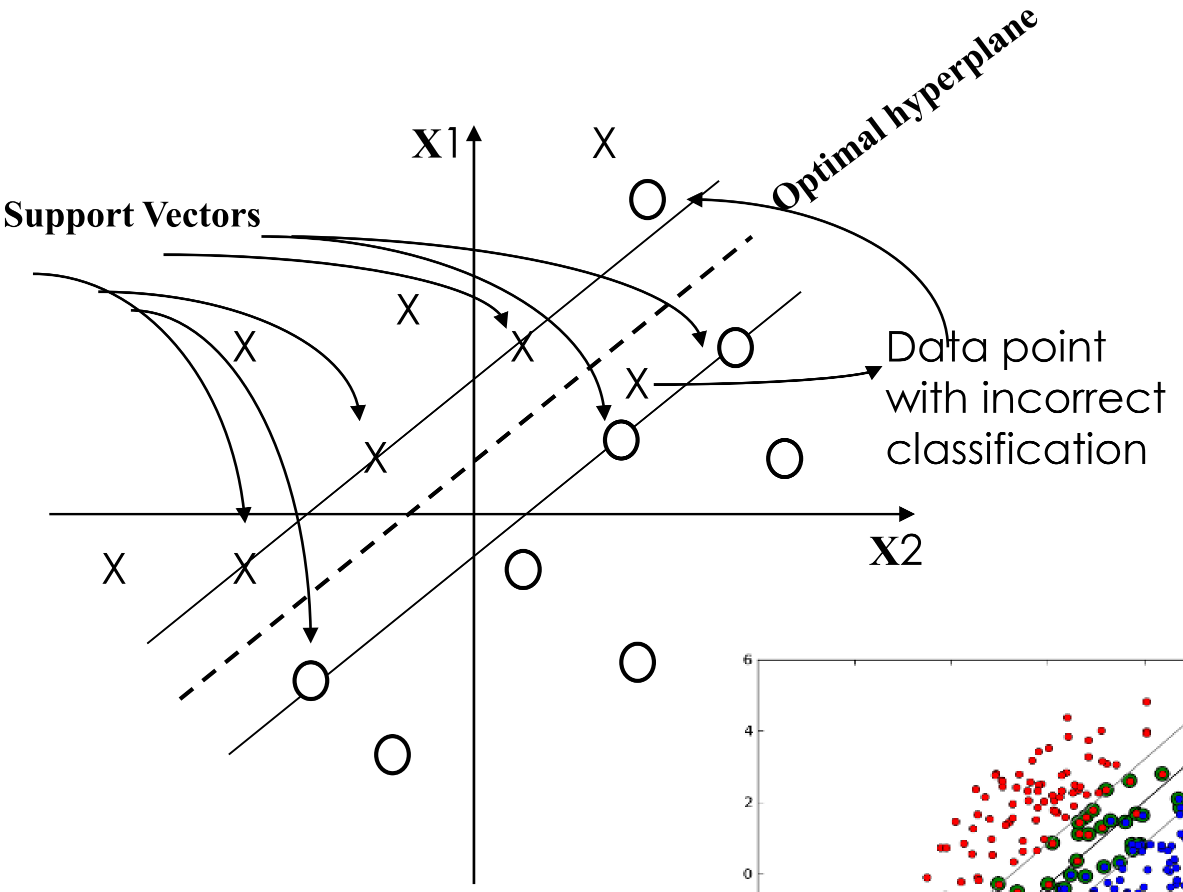
- The goal of a support vector machine for *linearly separable patterns* is to find the particular hyper-plane for which the margin of separation ρ is maximized.
- *Support vectors*: those data points that lie closest to the decision surface and are therefore the most difficult to classify

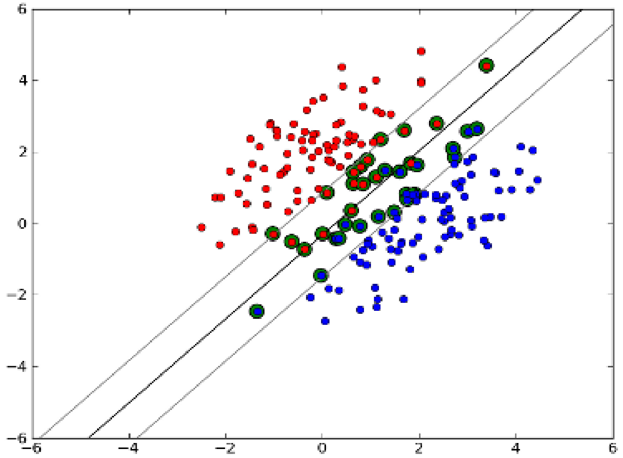


There are infinite hyperplanes, but SVM searches for the optimal hyperplane.

Soft Margin Linear SVM: Optimal Hyperplane & Support Vector

- Given a set of *not linearly separable* training patterns, it is not possible to construct a separating hyperplane without encountering classification error.
- The goal of a support vector machine for *not linearly separable patterns* is to find an optimal hyperplane that minimizes the misclassification error, averaged over the training set.





SVM: Soft margin solution

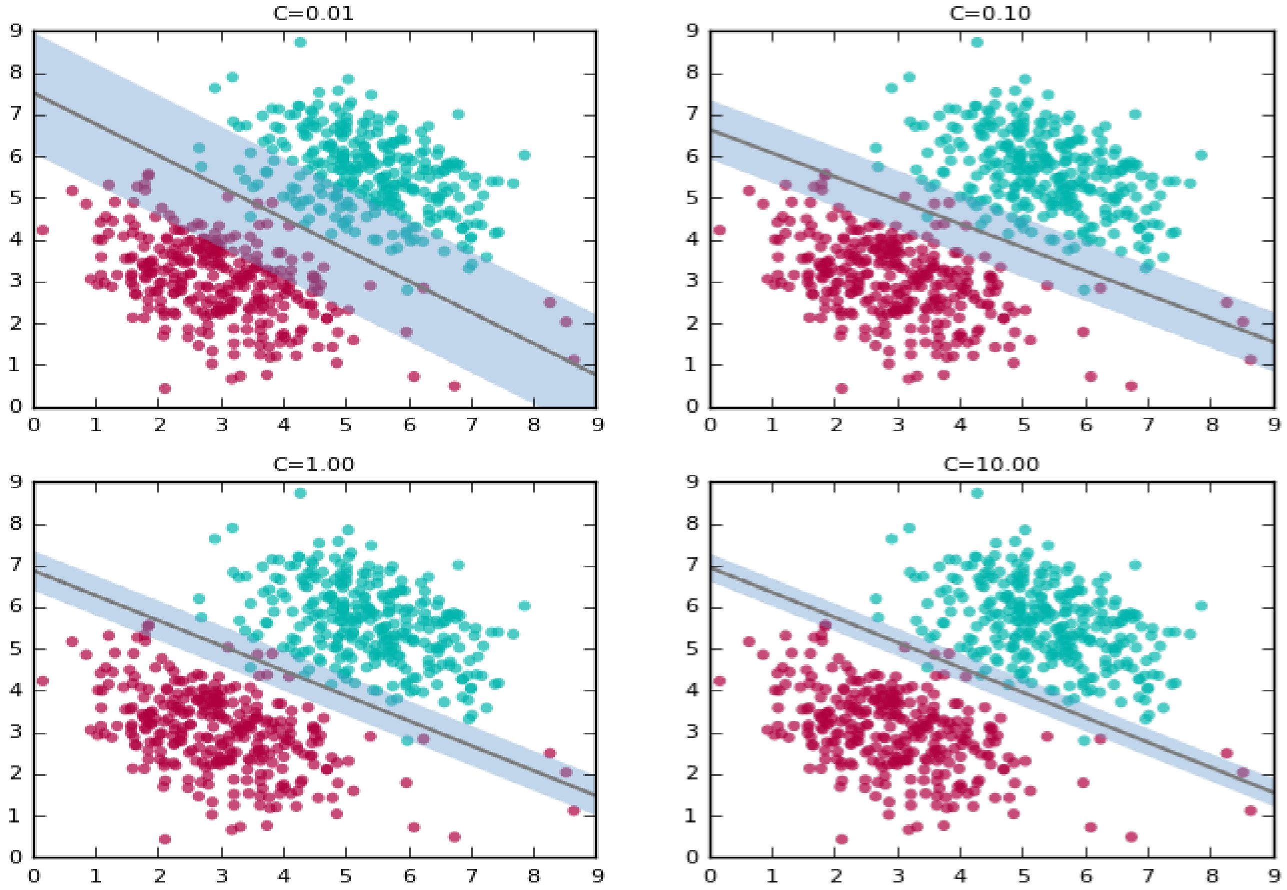
- There are optimization functions proposed for the case with soft margin, such as

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \xi_i \\ \text{subject to} & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \end{array}$$

- C is a penalty parameter
  - small $C \Rightarrow$ wide margin (more tolerance)
    - many support vectors will be on the margin
  - large $C \Rightarrow$ narrow margin
    - there will be few support vectors on the margin
  - $C \rightarrow \infty$ enforces all constraints $\Rightarrow$ hard margin

SVM: Soft margin solution - C value

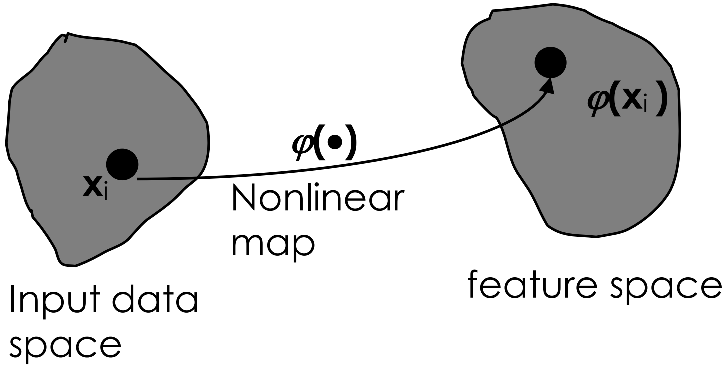
- A higher value of C implies you want lesser errors on the training data.

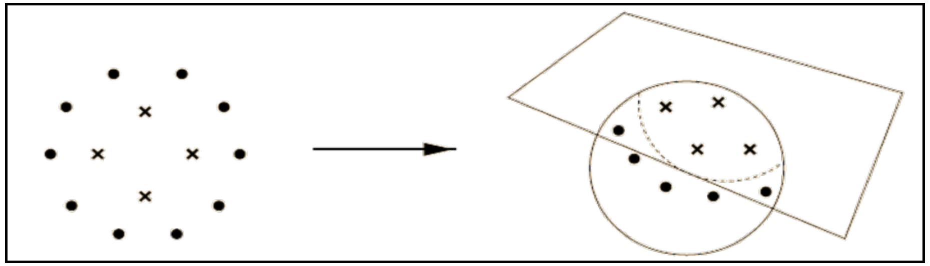


https://blog.statsbot.co/support-vector-machines-tutorial-c1618e635e93

SVM with Non–linear Kernels

- To construct a SVM for classification with an input space made up of non-linearly separable patterns
- Form Inner-product kernels
  - The multidimensional input space is transformed to a new feature space where the patterns are linearly separable with high probability, provided
    - **(a) The transformation is nonlinear**
    - **(b) The dimensionality of the feature is high enough**
  - A subset of training samples $\{x_1, x_2, \ldots x_{m1}\}$ will be used as support vectors
- Define the separating hyperplane as a linear function of vector drawn from the feature space rather than the original input space





SVM : Typical Kernel Functions for Nonlinear Classification

- Apply a kernel function $K(X_i, X_j)$ to the original data, i.e.

$$K(X_i, X_j) = \Phi(X_i)\ \Phi(X_j)$$

- Typical Kernel Functions

$$\text{Polynomial kernel of degree } h: \quad K(X_i, X_j) = (X_i \cdot X_j + 1)^h$$

$$\text{Gaussian radial basis function kernel}: \quad K(X_i, X_j) = e^{-\|X_i - X_j\|^2 / 2\sigma^2}$$

$$\text{Sigmoid kernel}: \quad K(X_i, X_j) = \tanh(\kappa X_i \cdot X_j - \delta)$$

Applications of SVM

- SVMs have been widely applied in
  - Bioinfomatics
  - Machine Vision
  - Text Categorization
  - Handwritten Character Recognition
  - ……

Workshop

Workshop

- Open the iPython notebook provided.
- You will build decision tree, neural network and SVM models in this workshop.
- As you go through the notebook, make sure you understand how each different model is built. (you can save notes as markdown in the notebook).
- Compare the performance of these models.
- Experiment with different parameter settings.
- You may try with your own datasets.

Contact eGL

Singapore e-Government Leadership Centre
National University of Singapore
29 Heng Mui Keng Terrace
Block D & E
Singapore 119620

Tel	:	**(65) 6516 1156**
Fax	:	**(65) 6778 2571**
URL	:	**www.egl.sg**
Email	:	**egl-enquiries@nus.edu.sg**

Inspire Lead Transform