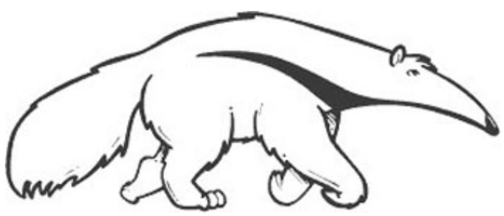
Machine Learning and Data Mining

Multi-layer Perceptrons & Neural Networks

Prof. Alexander Ihler

Fall 2012



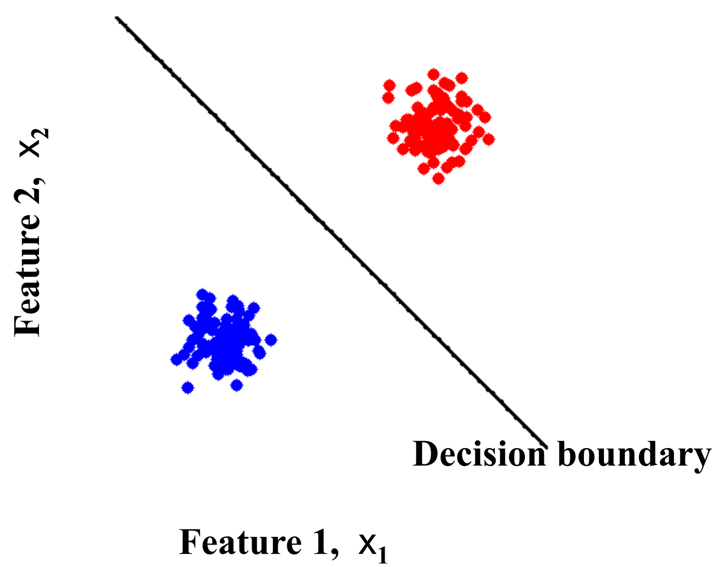
BREN:ICS
INFORMATION AND COMPUTER SCIENCES

UNIVERSITY of CALIFORNIA IRVINE

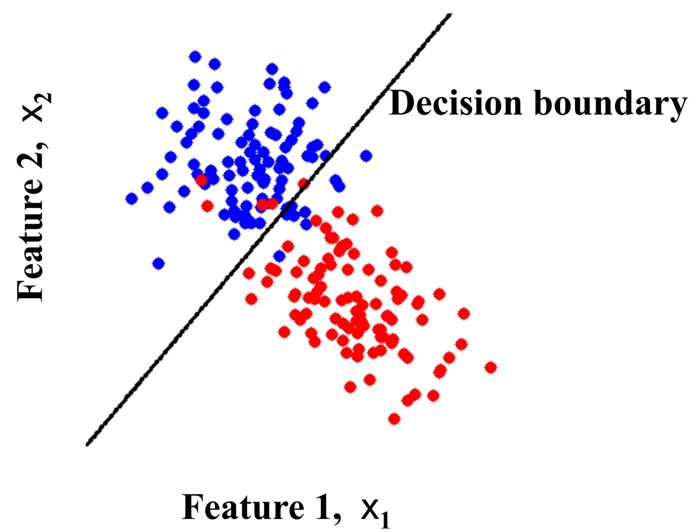
Linear Classifiers (Perceptrons)

- Linear Classifiers
  - a linear classifier is a mapping which partitions feature space using a linear function (a straight line, or a hyperplane)
  - separates the two classes using a straight line in feature space
  - in 2 dimensions the decision boundary is a straight line

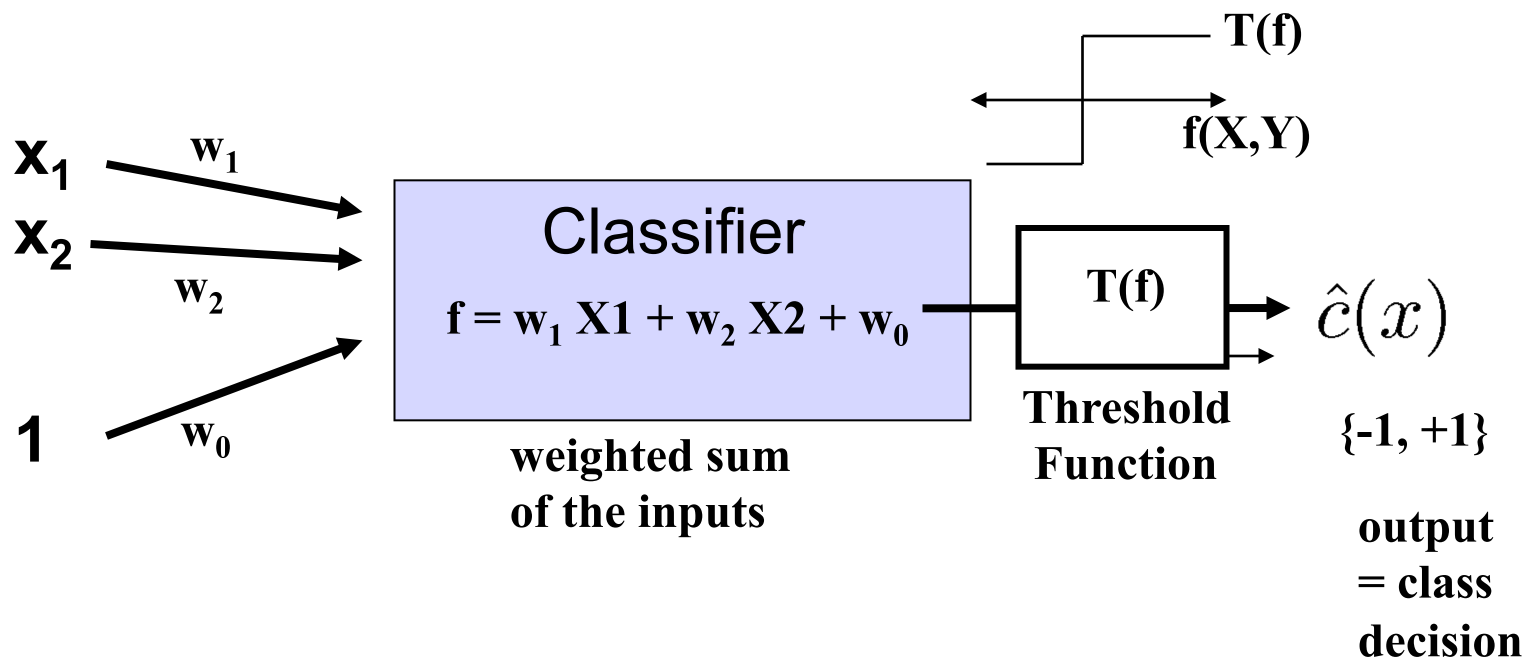
Linearly separable data



Linearly non-separable data



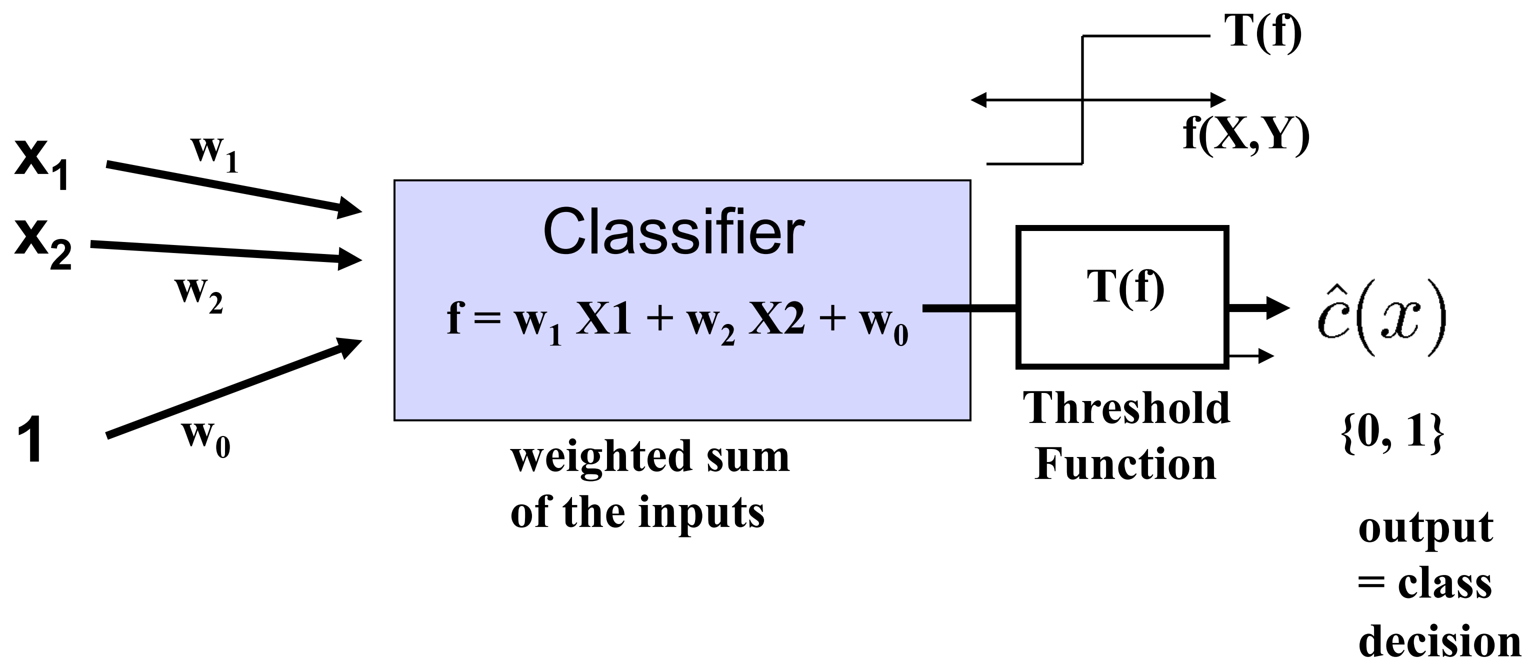
Perceptron Classifier (2 features)



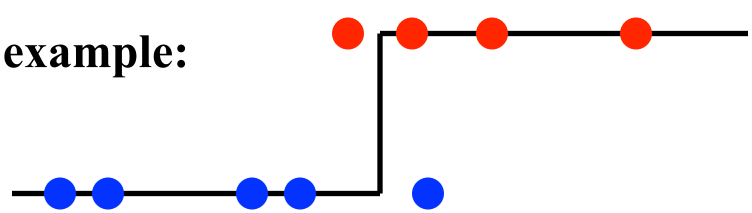
Decision Boundary at $f(x) = 0$

Solve: $X_2 = -w_1/w_2\, X_1 - w_0/w_2$ (Line)

Perceptron (Linear classifier)



1D example:



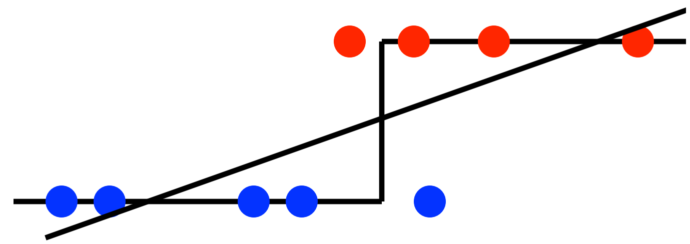
$T(f) = 0$ if $f < 0$

$T(f) = 1$ if $f > 0$

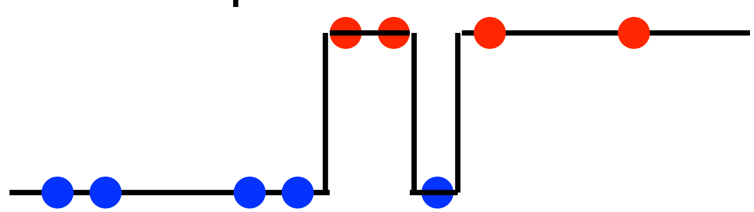
Decision boundary = "x such that $T(w_1 x + w_0)$ transitions"

Features and perceptrons

- Recall the role of features
  - We can create extra features that allow more complex decision boundaries
  - Linear classifiers
  - Features [1,x]
    - Decision rule: T(ax+b) = ax + b >/< 0
    - Boundary ax+b =0 => point
  - Features [1,x,x^2]
    - Decision rule T(ax^2+bx+c)
    - Boundary ax^2+bx+c = 0 = ?



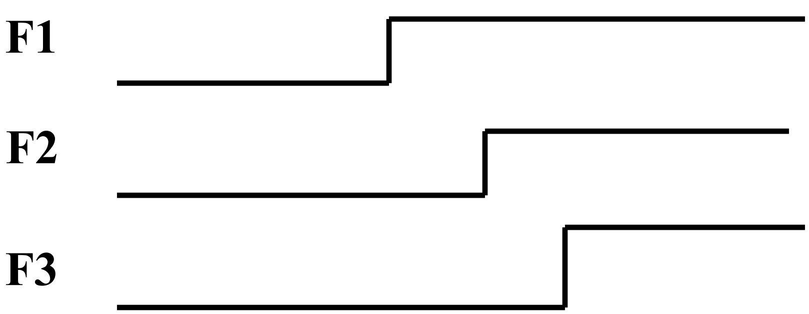
  - What features can produce this decision rule?



Features and perceptrons

- Recall the role of features
  - We can create extra features that allow more complex decision boundaries
  - For example, polynomial features

- What other kinds of features could we choose?
  - Step functions?

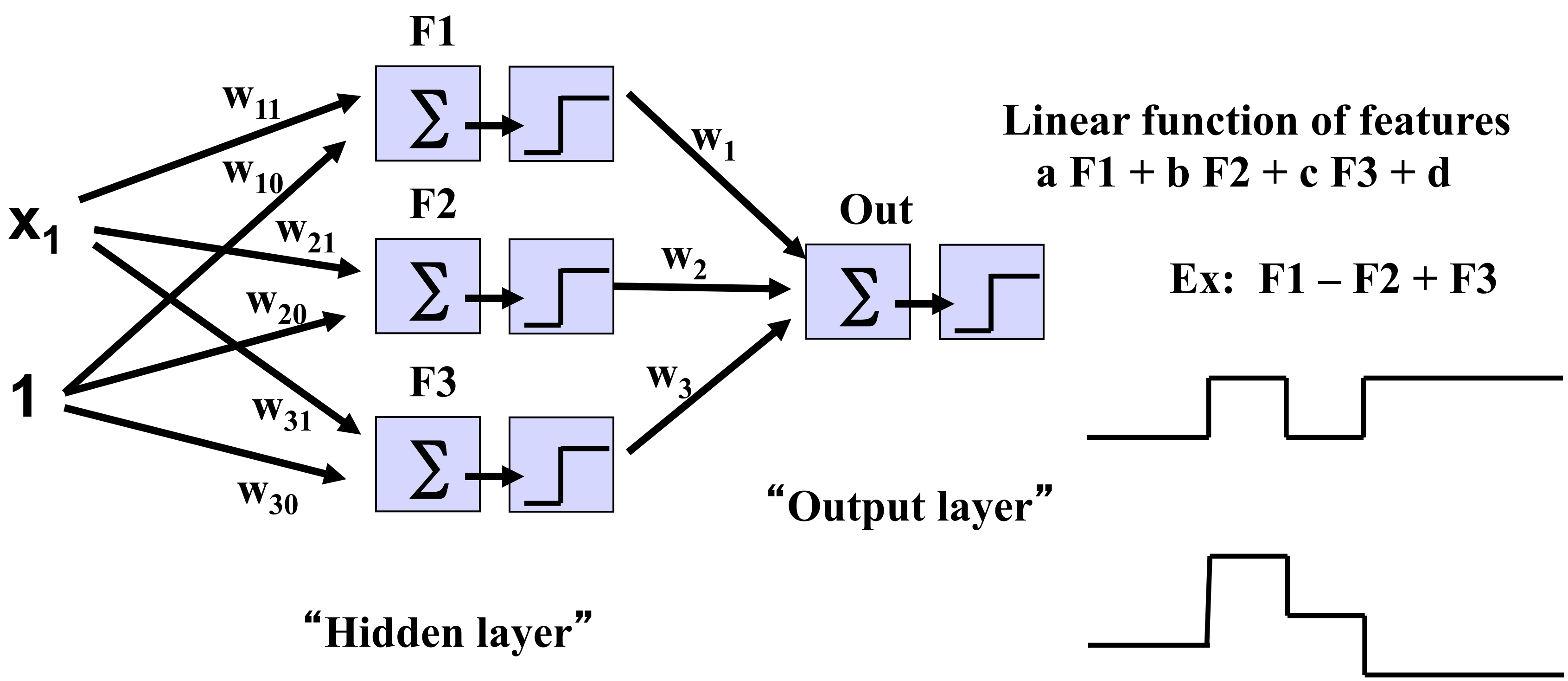


Linear function of features
a F1 + b F2 + c F3 + d

Ex: F1 – F2 + F3

Multi-layer perceptron model

- Step functions are just perceptrons!
  - "Features" are outputs of a perceptron
  - Combination of features output of another

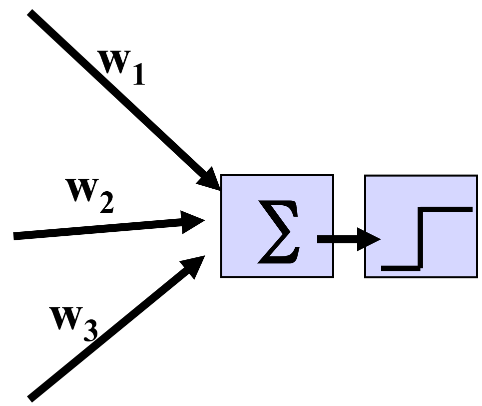


Features of MLPs

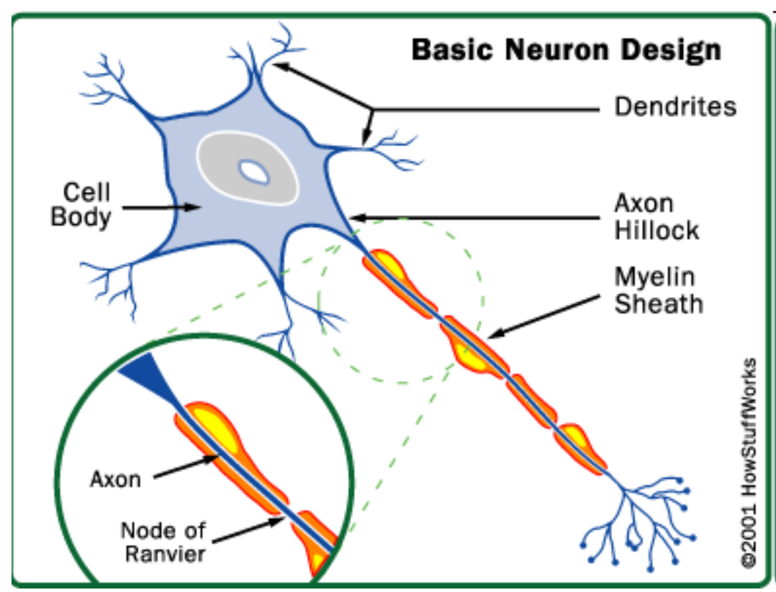
- Simple building blocks
  - Each element is just a perceptron f'n
- Can build upwards
  - 2 layer: simple features, complex output
  - 3 layer: complex features
  - 4 layer: even more so…
    - Current research: "deep" hierarchies
- Flexible function approximation
  - Can represent any function arbitrarily closely
  - Given enough hidden units
  - Even a 2-layer with enough hidden nodes

Neural networks

- Another term for MLPs
- Biological motivation



- Neurons
  - "Simple" cells
  - Dendrites sense charge
  - Cell weighs inputs
  - "Fires" axon



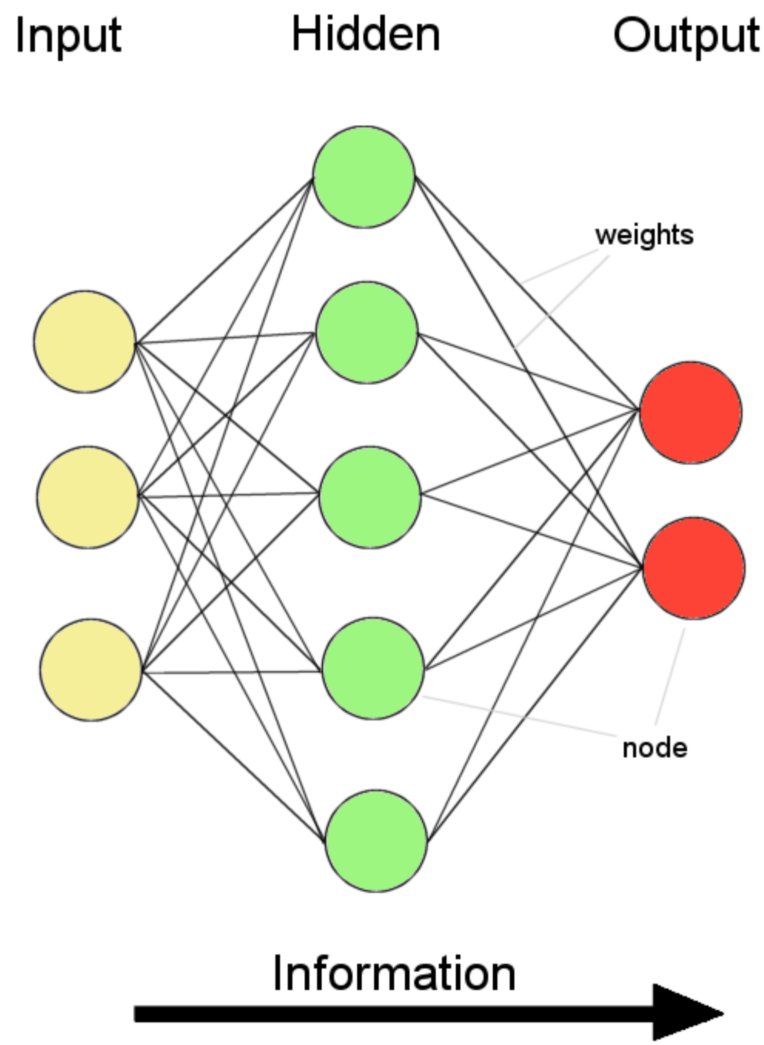
"How stuff works: the brain"

Feed-forward networks

- Information flows left – right
- Observed vars input
- Compute hidden nodes
- Compute next layer…

- Info distributed
- Parallel computation

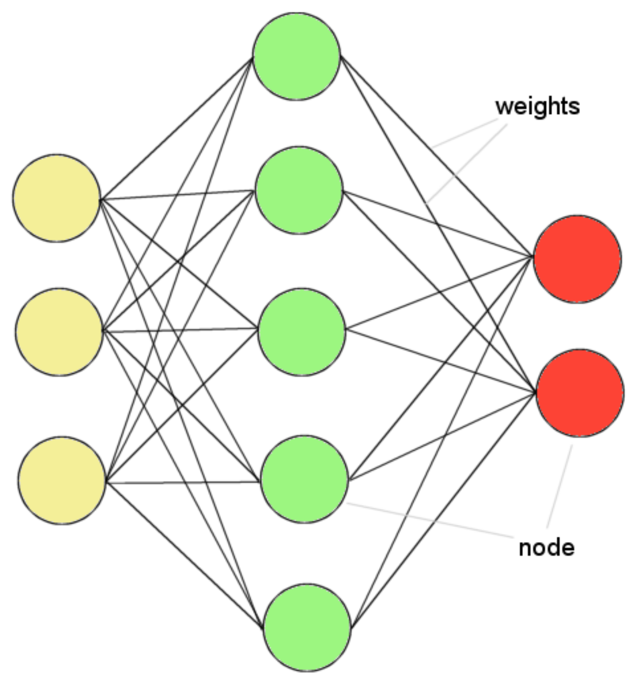
- Alternative: feedback
  - “Recurrent” neural nets
  - Cycles of dependence
  - More complex functions
  - Can have “memory”



Training MLPs

- Observe features "x" with target "y"
- Push "x" through NN = output is "ŷ"
- Error: $(y - \hat{y})^2$
- How should we update the weights to improve?

- Single layer
  - Logistic sigmoid function
  - Smooth, differentiable
- Optimize using:
  - Batch gradient descent
  - Online gradient descent



Backpropagation

- Just gradient descent…
- Apply the chain rule to the MLP

- Recall: logistic regression

$$\frac{\partial C}{\partial w_i} = -2(y - \hat{y}(w,x))\ \sigma'(w,x)\ x_i$$

Logistic regression

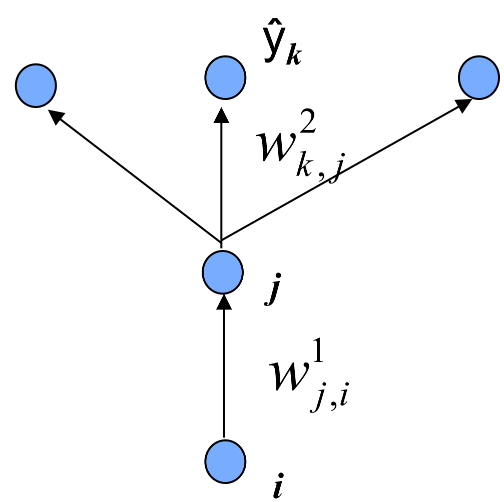
$$\hat{y}^{(i)} = \sigma(\sum_j w_j x_j^{(i)}) = (1 + \exp(-wx^{(i)}))^{-1}$$

$$\sigma'(z) = \sigma(z)(1 - \sigma(z))$$

- Multi-layer:
  - Output layer $\hat{y}_k = \sigma(s_k) = \sigma(w^2_{k1}\ h_1 + \ldots)$
  - Hidden layer $h_j = \sigma(t_j) = \sigma(w^1_{j1}\ x_1 + \ldots)$

$$\frac{\partial C}{\partial w^2_{kj}} = \boxed{-2(y_k - \hat{y}_k)\ \sigma'(s_k)\ h_j}$$

$$\frac{\partial C}{\partial w^1_{ji}} = -2\sum_k \boxed{(y_k - \hat{y}_k)\ \sigma'(s_k)}\ w_{kj}\ \sigma'(t_j)\ x_i$$



MLPs in practice

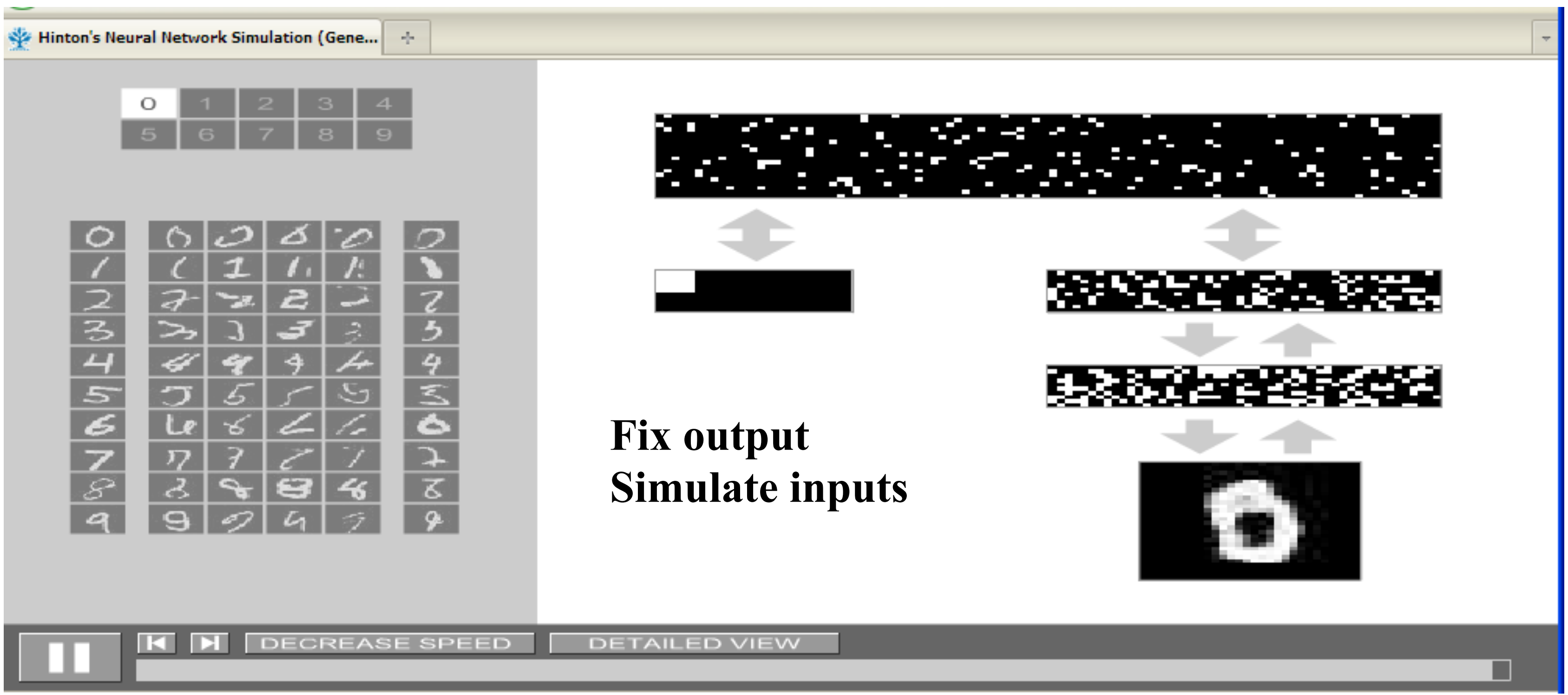
- Example: Deep belief nets (Hinton et al. 2007)
  - Handwriting recognition
  - Online demo
  - 10 label <=> 2000 top <=> 500 high <=> 500 mid <=> 784 pixels

MLPs in practice

- Example: Deep belief nets (Hinton et al. 2007)
  - Handwriting recognition
  - Online demo
  - 10 label <=> 2000 top <=> 500 high <=> 500 mid <=> 784 pixels

MLPs in practice

- Example: Deep belief nets (Hinton et al. 2007)
  - Handwriting recognition
  - Online demo
  - 10 label <=> 2000 top <=> 500 high <=> 500 mid <=> 784 pixels



Neural networks & DBNs

- Want to try them out?
- Matlab "Deep Learning Toolbox"

  https://github.com/rasmusbergpalm/DeepLearnToolbox

- Also
  - A built-in toolbox for Matlab
    - Have to have a license…
  - Netlab
    - Not updated in some time

Summary

- Neural networks, multi-layer perceptrons
- Cascade of simple perceptrons
  - Each just a linear classifier
  - Hidden units used to create new features
- Together, general function approximators
  - Enough hidden units (features) = any f'n
  - Can create nonlinear classifiers
  - Also used for function approximation, regression, …
- Training via backprop
  - Gradient descent; logistic; apply chain rule