Shallow Intro to Deep Learning'19

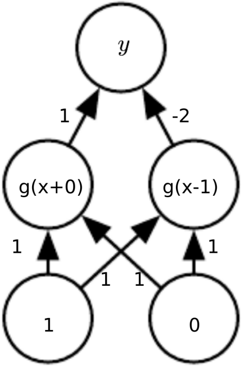
Feed Forward, Chapters 6, 6.1, 4.3, 5.9

Delft University of Technology

Lecturer: Jan van Gemert

What is the outcome of this Feed Forward net?

Book: chapter 6, 6.1

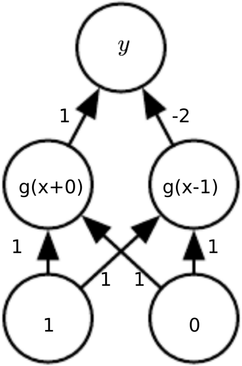


$$g(z) = \max\{0, z\}$$

What is the outcome of this Feed Forward net?

Book: chapter 6, 6.1

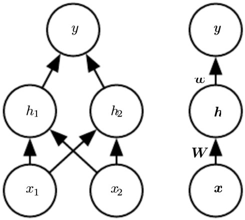
Outcome: 1



$$g(z) = \max\{0, z\}$$

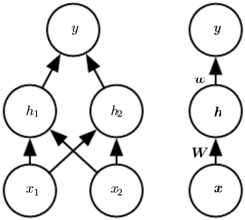
Deep Feed forward Networks (Multi-layered perceptron)

- Approximate some function f^*
- Eg: classifier $y = f^*(x)$
- Learn parameters θ for mapping $y = f(x; \theta)$



Deep Feed forward Networks (Multi-layered perceptron)

- Approximate some function f^*
- Eg: classifier $y = f^*(x)$
- Learn parameters θ for mapping $y = f(x; \theta)$

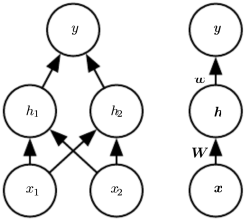


Q: How many parameters?

Deep Feed forward Networks (Multi-layered perceptron)

- Approximate some function f^*
- Eg: classifier $y = f^*(x)$
- Learn parameters θ for mapping $y = f(x; \theta)$

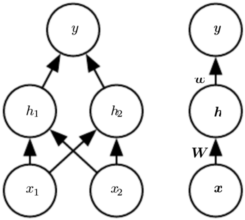
Q: How many parameters? A: 8



Deep Feed forward Networks (Multi-layered perceptron)

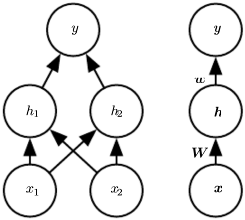
- Approximate some function f^*
- Eg: classifier $y = f^*(x)$
- Learn parameters θ for mapping $y = f(x; \theta)$

Q: How many parameters? A: 8
Q: Are the two graphs the same?



Deep Feed forward Networks (Multi-layered perceptron)

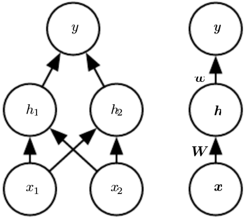
- Approximate some function f^*
- Eg: classifier $y = f^*(x)$
- Learn parameters θ for mapping $y = f(x; \theta)$



Q: How many parameters? A: 8
Q: Are the two graphs the same?
A: $g(x1w1 + x2w2 + c1)w5 + g(x1w3 + x2w4 + c2)w6 = g(x^\top W + c^\top)w$

Deep Feed forward Networks (Multi-layered perceptron)

- Approximate some function f^*
- Eg: classifier $y = f^*(x)$
- Learn parameters θ for mapping $y = f(x; \theta)$



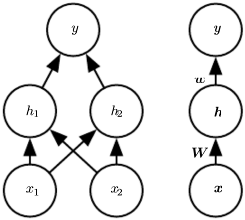
Q: How many parameters? A: 8
Q: Are the two graphs the same?
A: $g(x1w1 + x2w2 + c1)w5 + g(x1w3 + x2w4 + c2)w6 = g(x^\top W + c^\top)w$

Networked in a function chain $f(x) = f^{(3)}(f^{(2)}(f^{(1)}(x)))$

Q: For f, what is: first layer? Second? Output? Hidden? Depth?

Deep Feed forward Networks (Multi-layered perceptron)

- Approximate some function f^*
- Eg: classifier $y = f^*(x)$
- Learn parameters θ for mapping $y = f(x; \theta)$



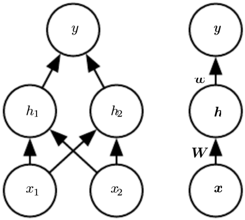
Q: How many parameters? A: 8
Q: Are the two graphs the same?
A: $g(x1w1 + x2w2 + c1)w5 + g(x1w3 + x2w4 + c2)w6 = g(x^\top W + c^\top)w$

Networked in a function chain $f(x) = f^{(3)}(f^{(2)}(f^{(1)}(x)))$

Q: For f, what is: first layer? Second? Output? Hidden? Depth?
A: first layer: $f^{(1)}$, Second: $f^{(2)}$, Output: $f^{(3)}$, Hidden: $f^{(2)}$, Depth: 3

Deep Feed forward Networks (Multi-layered perceptron)

- Approximate some function f^*
- Eg: classifier $y = f^*(x)$
- Learn parameters θ for mapping $y = f(x; \theta)$



Q: How many parameters? A: 8
Q: Are the two graphs the same?
A: $g(x1w1 + x2w2 + c1)w5 + g(x1w3 + x2w4 + c2)w6 = g(x^\top W + c^\top)w$

Networked in a function chain $f(x) = f^{(3)}(f^{(2)}(f^{(1)}(x)))$

Q: For f, what is: first layer? Second? Output? Hidden? Depth?
A: first layer: $f^{(1)}$, Second: $f^{(2)}$, Output: $f^{(3)}$, Hidden: $f^{(2)}$, Depth: 3

Goal: drive $f(x)$ to match $f^*(x)$

Training a network

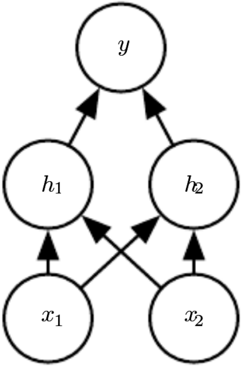
Chapter 4.3

Training set:

Data		Label
0	1	1
1	1	0
0	0	0
1	0	1

While not converged:

1. Present a training sample
2. Compare the results
3. Update the weights



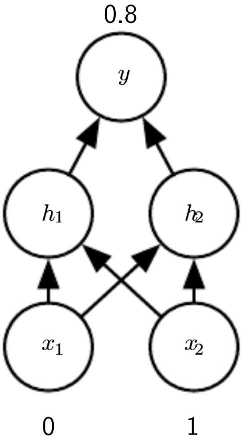
Training a network

Training set:

Data		Label
0	1	1
1	1	0
0	0	0
1	0	1

While not converged:

1. Present a training sample
2. Compare the results
3. Update the weights



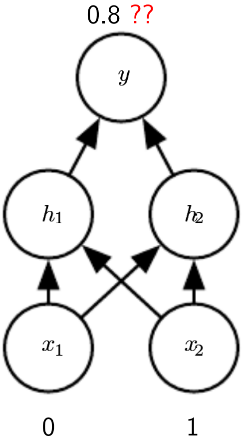
Training a network

Training set:

Data		Label
0	1	1
1	1	0
0	0	0
1	0	1

While not converged:

1. Present a training sample
2. Compare the results
3. Update the weights



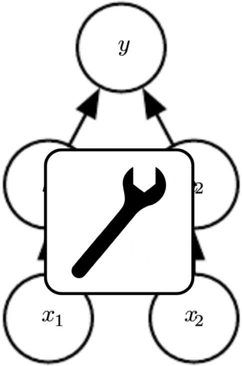
Training a network

Training set:

Data		Label
0	1	1
1	1	0
0	0	0
1	0	1

While not converged:

1. Present a training sample
2. Compare the results
3. Update the weights



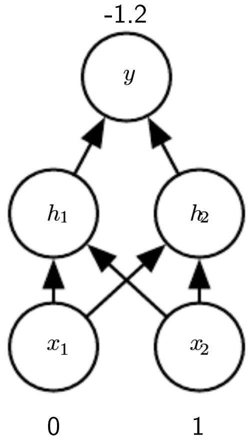
Training a network

Training set:

Data		Label
0	1	1
1	1	0
0	0	0
1	0	1

While not converged:

1. Present a training sample
2. Compare the results
3. Update the weights



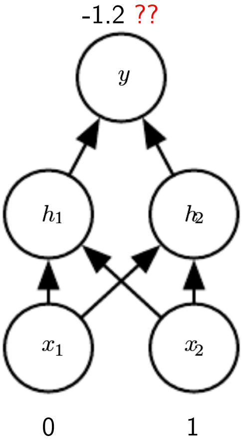
Training a network

Training set:

Data		Label
0	1	1
1	1	0
0	0	0
1	0	1

While not converged:

1. Present a training sample
2. Compare the results
3. Update the weights



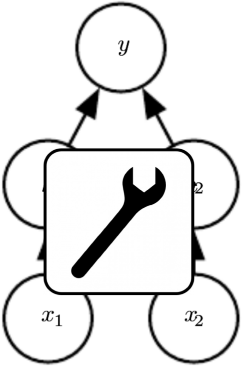
Training a network

Training set:

Data		Label
0	1	1
1	1	0
0	0	0
1	0	1

While not converged:

1. Present a training sample
2. Compare the results
3. Update the weights



Update weights

- Minimize some criterion, an objective/cost/loss/error-function
  For example, the mean squared error $f(x, y) = \frac{1}{n} \sum_{i=1}^{n} (h(x_i) - y_i)^2$

Update weights

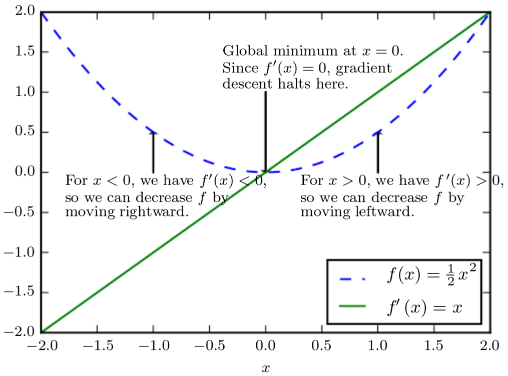
- Minimize some criterion, an objective/cost/loss/error-function
  For example, the mean squared error $f(x, y) = \frac{1}{n}\sum_{i=1}^{n}(h(x_i) - y_i)^2$
- Q: For $f(x)$, If I make a small change to x, how does it change $f(x)$?

Update weights

- Minimize some criterion, an objective/cost/loss/error-function
  For example, the mean squared error $f(x, y) = \frac{1}{n} \sum_{i=1}^{n} (h(x_i) - y_i)^2$
- Q: For $f(x)$, If I make a small change to x, how does it change $f(x)$?
- A: Derivative f' or $\frac{\partial f}{\partial x}$, tells you $f(x + \epsilon) \approx f(x) + \epsilon f'(x)$
- Q: How to reduce loss?

Update weights

- Minimize some criterion, an objective/cost/loss/error-function
  For example, the mean squared error $f(x,y) = \frac{1}{n}\sum_{i=1}^{n}(h(x_i) - y_i)^2$
- Q: For $f(x)$, If I make a small change to x, how does it change $f(x)$?
- A: Derivative f' or $\frac{\partial f}{\partial x}$, tells you $f(x+\epsilon) \approx f(x) + \epsilon f'(x)$
- Q: How to reduce loss? A: By moving in the opposite sign of the gradient



Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$?

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$? A: Single variable.

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$? A: Single variable.
- Q: What is the gradient $\nabla_\theta f(\mathbf{x}; \theta)$?

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$? A: Single variable.
- Q: What is the gradient $\nabla_\theta f(\mathbf{x}; \theta)$? A: Vector of all partial derivatives
- Q: What does this do: $\theta' = \theta - \epsilon \nabla_\theta f(\mathbf{x}; \theta)$?

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$? A: Single variable.
- Q: What is the gradient $\nabla_\theta f(\mathbf{x}; \theta)$? A: Vector of all partial derivatives
- Q: What does this do: $\theta' = \theta - \epsilon \nabla_\theta f(\mathbf{x}; \theta)$?
  A: Updates the parameters using gradient descent, where ϵ is the learning rate

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$? A: Single variable.
- Q: What is the gradient $\nabla_\theta f(\mathbf{x}; \theta)$? A: Vector of all partial derivatives
- Q: What does this do: $\theta' = \theta - \epsilon \nabla_\theta f(\mathbf{x}; \theta)$?
  A: Updates the parameters using gradient descent, where ϵ is the learning rate
- The loss on all samples: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} L(x^{(i)}, y^{(i)}, \theta)$

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$? A: Single variable.
- Q: What is the gradient $\nabla_\theta f(\mathbf{x}; \theta)$? A: Vector of all partial derivatives
- Q: What does this do: $\theta' = \theta - \epsilon \nabla_\theta f(\mathbf{x}; \theta)$?
  A: Updates the parameters using gradient descent, where ϵ is the learning rate
- The loss on all samples: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} L(x^{(i)}, y^{(i)}, \theta)$
- Gradient for all samples: $\nabla_\theta J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta L(x^{(i)}, y^{(i)}, \theta)$

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$? A: Single variable.
- Q: What is the gradient $\nabla_\theta f(\mathbf{x}; \theta)$? A: Vector of all partial derivatives
- Q: What does this do: $\theta' = \theta - \epsilon \nabla_\theta f(\mathbf{x}; \theta)$?
  A: Updates the parameters using gradient descent, where ϵ is the learning rate
- The loss on all samples: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} L(x^{(i)}, y^{(i)}, \theta)$
- Gradient for all samples: $\nabla_\theta J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta L(x^{(i)}, y^{(i)}, \theta)$
  Q: Why *not* use all samples?

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$? A: Single variable.
- Q: What is the gradient $\nabla_\theta f(\mathbf{x}; \theta)$? A: Vector of all partial derivatives
- Q: What does this do: $\theta' = \theta - \epsilon \nabla_\theta f(\mathbf{x}; \theta)$?
  A: Updates the parameters using gradient descent, where ϵ is the learning rate
- The loss on all samples: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} L(x^{(i)}, y^{(i)}, \theta)$
- Gradient for all samples: $\nabla_\theta J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta L(x^{(i)}, y^{(i)}, \theta)$
  Q: Why *not* use all samples? A: Huge datasets are impractical
- Stochastic Gradient Descent (SGD) is an approximation of the gradient from a small number of samples m'
- How does SGD update θ'?

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x};\theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial\theta_i} f(\mathbf{x};\theta)$? A: Single variable.
- Q: What is the gradient $\nabla_\theta f(\mathbf{x};\theta)$? A: Vector of all partial derivatives
- Q: What does this do: $\theta' = \theta - \epsilon\nabla_\theta f(\mathbf{x};\theta)$?
  A: Updates the parameters using gradient descent, where ϵ is the learning rate
- The loss on all samples: $J(\theta) = \frac{1}{m}\sum_{i=1}^{m} L(x^{(i)}, y^{(i)}, \theta)$
- Gradient for all samples: $\nabla_\theta J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \nabla_\theta L(x^{(i)}, y^{(i)}, \theta)$
  Q: Why *not* use all samples? A: Huge datasets are impractical
- Stochastic Gradient Descent (SGD) is an approximation of the gradient from a small number of samples m'
- How does SGD update θ'? $\theta' = \theta - \epsilon\frac{1}{m'}\sum_{i=1}^{m'} \nabla_\theta L(x^{(i)}, y^{(i)}, \theta)$

Stochastic Gradient Descent

Chapter 5.9

- If the loss/error/cost function is given by $y = f(\mathbf{x}; \theta)$
- Q: How many dimensions in partial derivative $\frac{\partial}{\partial \theta_i} f(\mathbf{x}; \theta)$? A: Single variable.
- Q: What is the gradient $\nabla_\theta f(\mathbf{x}; \theta)$? A: Vector of all partial derivatives
- Q: What does this do: $\theta' = \theta - \epsilon \nabla_\theta f(\mathbf{x}; \theta)$?
  A: Updates the parameters using gradient descent, where ϵ is the learning rate
- The loss on all samples: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} L(x^{(i)}, y^{(i)}, \theta)$
- Gradient for all samples: $\nabla_\theta J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta L(x^{(i)}, y^{(i)}, \theta)$
  Q: Why *not* use all samples? A: Huge datasets are impractical
- Stochastic Gradient Descent (SGD) is an approximation of the gradient from a small number of samples m'
- How does SGD update θ'? $\theta' = \theta - \epsilon \frac{1}{m'} \sum_{i=1}^{m'} \nabla_\theta L(x^{(i)}, y^{(i)}, \theta)$

This is the procedure we've already seen.

Training a network

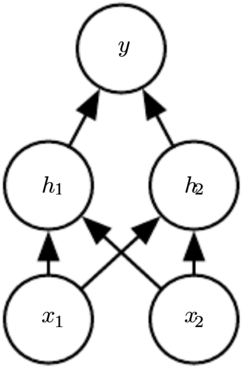
Chapter 4.3

Training set:

Data		Label
0	1	1
1	1	0
0	0	0
1	0	1

While not converged:

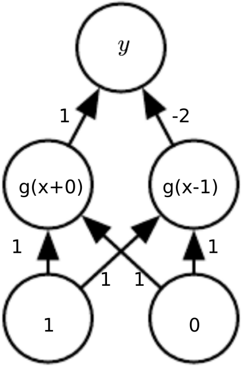
1. Present a training sample
2. Compare the results
3. Update the weights



Questions?

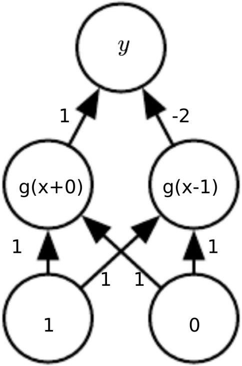
Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?



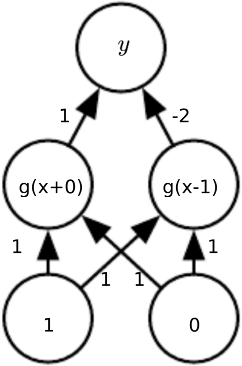
Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?
  A: A linear model, $f(x, w, b) = x^{\top} w + b$



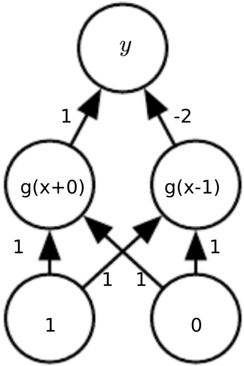
Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?
  A: A linear model, $f(x, w, b) = x^\top w + b$
- Linear is good, but possibly too rigid



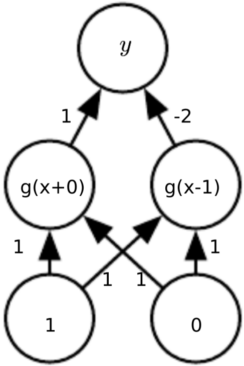
Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?
  A: A linear model, $f(x, w, b) = x^\top w + b$
- Linear is good, but possibly too rigid
- Extend to non-linear by applying a non-linear transformation $\phi(x)$ on the input



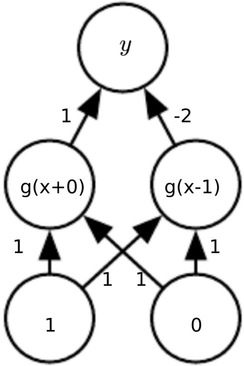
Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?
  A: A linear model, $f(x, w, b) = x^\top w + b$
- Linear is good, but possibly too rigid
- Extend to non-linear by applying a non-linear transformation $\phi(x)$ on the input
- The non-linear equation of f is



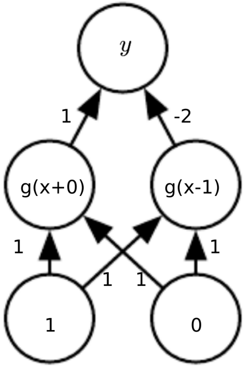
Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?
  A: A linear model, $f(x, w, b) = x^\top w + b$
- Linear is good, but possibly too rigid
- Extend to non-linear by applying a non-linear transformation $\phi(x)$ on the input
- The non-linear equation of f is
  $y = f(x, \theta, w, b) = \phi(x, \theta)^\top w + b$



Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?
  A: A linear model, $f(x, w, b) = x^\top w + b$
- Linear is good, but possibly too rigid
- Extend to non-linear by applying a non-linear transformation $\phi(x)$ on the input
- The non-linear equation of f is
  $y = f(x, \theta, w, b) = \phi(x, \theta)^\top w + b$

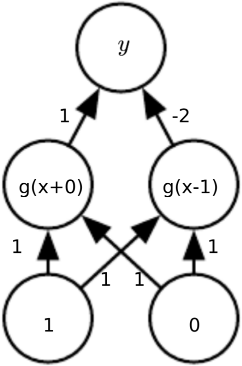


Three ways to make the input non-linear:

1. Generic kernel function
2. Designing feature extractors
3. Learn it..

Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?
  A: A linear model, $f(x, w, b) = x^\top w + b$
- Linear is good, but possibly too rigid
- Extend to non-linear by applying a non-linear transformation $\phi(x)$ on the input
- The non-linear equation of f is
  $y = f(x, \theta, w, b) = \phi(x, \theta)^\top w + b$

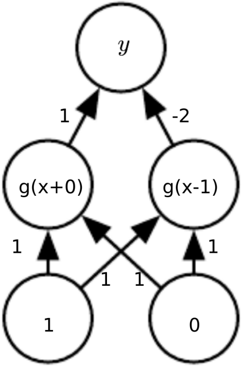


Three ways to make the input non-linear:

1. Generic kernel function
2. Designing feature extractors
3. Learn it.. $\quad y = f(x, \theta, w, b) = \phi(x, \theta)^\top w + b$

Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?
  A: A linear model, $f(x, w, b) = x^\top w + b$
- Linear is good, but possibly too rigid
- Extend to non-linear by applying a non-linear transformation $\phi(x)$ on the input
- The non-linear equation of f is
  $y = f(x, \theta, w, b) = \phi(x, \theta)^\top w + b$



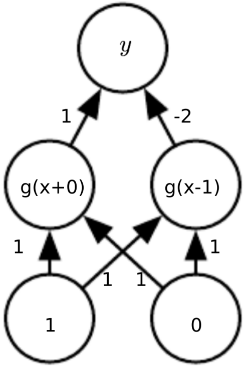
Three ways to make the input non-linear:

1. Generic kernel function
2. Designing feature extractors
3. Learn it.. $\quad y = f(x, \theta, w, b) = \phi(x, \theta)^\top w + b$

Step 3 allows to add knowledge to restrict 1 but more flexible than 2.

Deep Feed forward Networks (Multi-layered perceptron)

- Q: What is the equation of the output unit?
  A: A linear model, $f(x, w, b) = x^\top w + b$
- Linear is good, but possibly too rigid
- Extend to non-linear by applying a non-linear transformation $\phi(x)$ on the input
- The non-linear equation of f is
  $y = f(x, \theta, w, b) = \phi(x, \theta)^\top w + b$



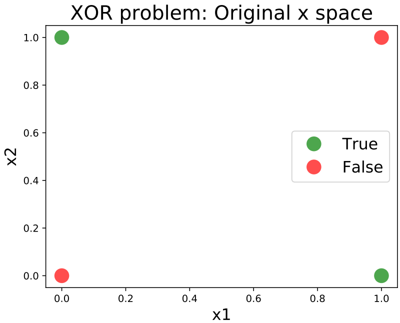
Three ways to make the input non-linear:

1. Generic kernel function
2. Designing feature extractors
3. Learn it.. $y = f(x, \theta, w, b) = \phi(x, \theta)^\top w + b$

Step 3 allows to add knowledge to restrict 1 but more flexible than 2.
It learns the representation (features)

Example: XOR

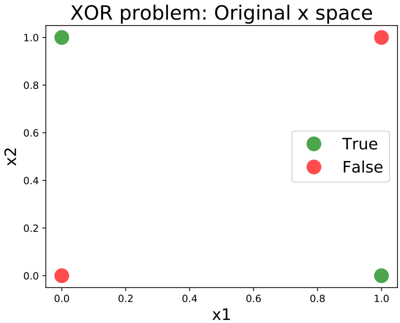
- Input: $\mathbb{X} = \{[0,0]^\top, [0,1]^\top, [1,0]^\top, [1,1]^\top\}$
- Labels: $\mathbb{Y} = \{0,1,1,0\}$
- Model: $f(x,w,b) = x^\top w + b$



Can it learn this?

Example: XOR

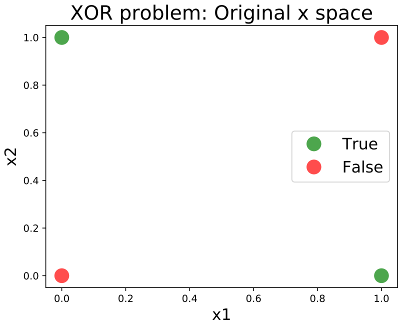
- Input: $\mathbb{X} = \{[0,0]^\top, [0,1]^\top, [1,0]^\top, [1,1]^\top\}$
- Labels: $\mathbb{Y} = \{0, 1, 1, 0\}$
- Model: $f(x, w, b) = x^\top w + b$



Can it learn this? Why not?
Result: $b = \frac{1}{2}, w = 0$; what does that do?

Example: XOR

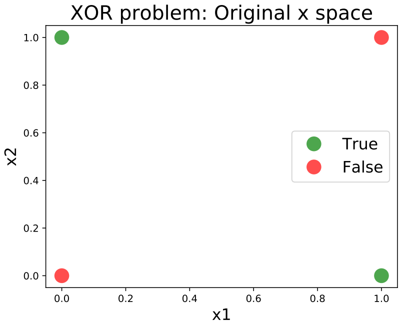
- Input: $\mathbb{X} = \{[0,0]^\top, [0,1]^\top, [1,0]^\top, [1,1]^\top\}$
- Labels: $\mathbb{Y} = \{0, 1, 1, 0\}$
- Model: $f(x, w, b) = x^\top w + b$



Can it learn this? Why not?
Result: $b = \frac{1}{2}, w = 0$; what does that do? everything $\frac{1}{2}$
Solution?

Example: XOR

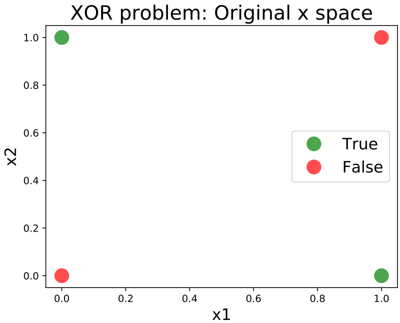
- Input: $\mathbb{X} = \{[0,0]^\top, [0,1]^\top, [1,0]^\top, [1,1]^\top\}$
- Labels: $\mathbb{Y} = \{0,1,1,0\}$
- Model: $f(x,w,b) = x^\top w + b$



Can it learn this? Why not?
Result: $b = \frac{1}{2}, w = 0$; what does that do? everything $\frac{1}{2}$
Solution? Add a hidden layer $h = f^1(x, W, c)$ and $y = f^2(h, w, b)$

Example: XOR

- Input: $\mathbb{X} = \{[0,0]^\top, [0,1]^\top, [1,0]^\top, [1,1]^\top\}$
- Labels: $\mathbb{Y} = \{0, 1, 1, 0\}$
- Model: $f(x, w, b) = x^\top w + b$



Can it learn this? Why not?
Result: $b = \frac{1}{2}, w = 0$; what does that do? everything $\frac{1}{2}$
Solution? Add a hidden layer $h = f^1(x, W, c)$ and $y = f^2(h, w, b)$
Complete model: $f(x, W, c, w, b) = f^2(f^1(x))$

Activation function

Complete model: $f(x, W, c, w, b) = f^2(f^1(x))$

Activation function

Complete model: $f(x, W, c, w, b) = f^2(f^1(x))$

- What function should f^1 compute? Linear?

Activation function

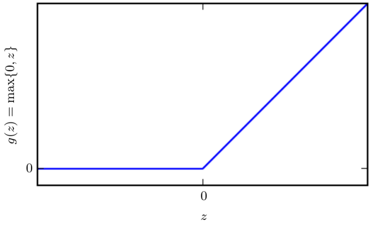
Complete model: $f(x, W, c, w, b) = f^2(f^1(x))$

- What function should f^1 compute? Linear? no.
  $f^1(x) = W^\top x$ and $f^2(h) = w^\top h$ then $f(x) = w^\top W^\top x$,
- Make non-linear by 'activation' function. $h = g(W^\top x + c)$
- Activation function applied **element-wise** $h_i = g(x^\top W_{:,i} + c_i)$
- Rectified linear unit $g(z) = \max\{0, z\}$ (graph?)

Activation function

Complete model: $f(x, W, c, w, b) = f^2(f^1(x))$

- What function should f^1 compute? Linear? no. $f^1(x) = W^\top x$ and $f^2(h) = w^\top h$ then $f(x) = w^\top W^\top x$,
- Make non-linear by 'activation' function. $h = g(W^\top x + c)$
- Activation function applied **element-wise** $h_i = g(x^\top W_{:,i} + c_i)$
- Rectified linear unit $g(z) = \max\{0, z\}$ (graph?)



Solution to XOR

Full model $f(x, W, c, w, b) = w^\top \max\{0, W^\top x + c\} + b$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = 0$$

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Q: can you validate this is a solution?

Solution to XOR

Full model $f(x, W, c, w, b) = w^\top \max\{0, W^\top x + c\} + b$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = 0$$

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Q: can you validate this is a solution?

$$W^\top X =$$

Solution to XOR

Full model $f(x, W, c, w, b) = w^\top \max\{0, W^\top x + c\} + b$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = 0$$

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Q: can you validate this is a solution?

$$W^\top X = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

Solution to XOR

Full model $f(x, W, c, w, b) = w^\top \max\{0, W^\top x + c\} + b$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = 0$$

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Q: can you validate this is a solution?

$$W^\top X = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

$$\max\{0, W^\top X + c\} =$$

Solution to XOR

Full model $f(x, W, c, w, b) = w^\top \max\{0, W^\top x + c\} + b$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = 0$$

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Q: can you validate this is a solution?

$$W^\top X = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

$$\max\{0, W^\top X + c\} = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Solution to XOR

Full model $f(x, W, c, w, b) = w^\top \max\{0, W^\top x + c\} + b$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = 0$$

Q: plot $\max\{0, W^\top X + c\}$ in 2d?

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Q: can you validate this is a solution?

$$W^\top X = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

$$\max\{0, W^\top X + c\} = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Solution to XOR

Full model $f(x, W, c, w, b) = w^\top \max\{0, W^\top x + c\} + b$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = 0$$

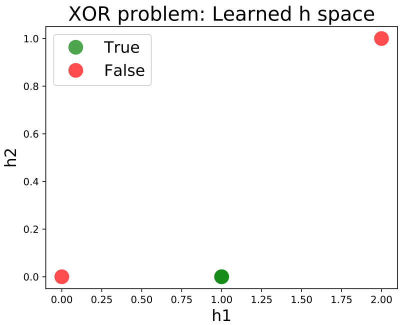
$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Q: can you validate this is a solution?

$$W^\top X = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

$$\max\{0, W^\top X + c\} = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Q: plot $\max\{0, W^\top X + c\}$ in 2d?



Solution to XOR

Full model $f(x, W, c, w, b) = w^\top \max\{0, W^\top x + c\} + b$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = 0$$

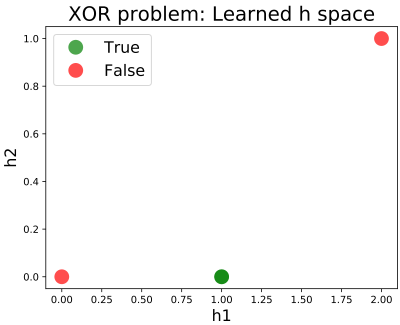
$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Q: can you validate this is a solution?

$$W^\top X = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

$$\max\{0, W^\top X + c\} = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Q: plot $\max\{0, W^\top X + c\}$ in 2d?



Solution:

$$w^\top \max\{0, W^\top X + c\} + b =$$

Solution to XOR

Full model $f(x, W, c, w, b) = w^\top \max\{0, W^\top x + c\} + b$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = 0$$

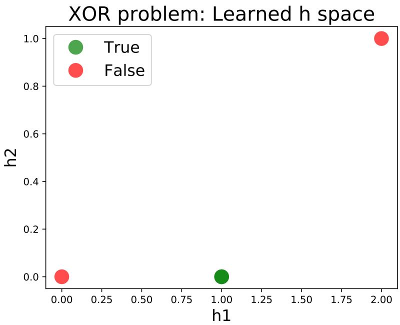
$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Q: can you validate this is a solution?

$$W^\top X = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

$$\max\{0, W^\top X + c\} = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Q: plot $\max\{0, W^\top X + c\}$ in 2d?



Solution:

$$w^\top \max\{0, W^\top X + c\} + b = \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix}$$

Questions?

Summary

- Feed forward networks
- How to train them using SGD
- How representations are learned (XOR example)