CSC321 Lecture 3: Linear Classifiers
– or –
What good is a single neuron?

Roger Grosse

Overview

- Classification: predicting a discrete-valued target
- In this lecture, we focus on binary classification: predicting a binary-valued target
- Examples
  - predict whether a patient has a disease, given the presence or absence of various symptoms
  - classify e-mails as spam or non-spam
  - predict whether a financial transaction is fraudulent

Overview

Design choices so far

- **Task:** regression, classification
- **Model/Architecture:** linear
- **Loss function:** squared error
- **Optimization algorithm:** direct solution, gradient descent, perceptron

Overview

Binary linear classification

- **classification:** predict a discrete-valued target
- **binary:** predict a binary target $t \in \{0, 1\}$
  - Training examples with $t = 1$ are called positive examples, and training examples with $t = 0$ are called negative examples. Sorry.
- **linear:** model is a linear function of $\mathbf{x}$, followed by a threshold:

$$z = \mathbf{w}^T \mathbf{x} + b$$

$$y = \begin{cases} 1 & \text{if } z \geq r \\ 0 & \text{if } z < r \end{cases}$$

Some simplifications

Eliminating the threshold

- We can assume WLOG that the threshold $r = 0$:

$$\mathbf{w}^T\mathbf{x} + b \geq r \quad \Longleftrightarrow \quad \mathbf{w}^T\mathbf{x} + \underbrace{b - r}_{\triangleq b'} \geq 0.$$

Some simplifications

Eliminating the threshold

- We can assume WLOG that the threshold $r = 0$:

$$\mathbf{w}^T\mathbf{x} + b \geq r \quad \Longleftrightarrow \quad \mathbf{w}^T\mathbf{x} + \underbrace{b - r}_{\triangleq b'} \geq 0.$$

Eliminating the bias

- Add a dummy feature x_0 which always takes the value 1. The weight w_0 is equivalent to a bias.

Some simplifications

Eliminating the threshold

- We can assume WLOG that the threshold $r = 0$:

$$\mathbf{w}^T\mathbf{x} + b \geq r \quad \Longleftrightarrow \quad \mathbf{w}^T\mathbf{x} + \underbrace{b - r}_{\triangleq b'} \geq 0.$$

Eliminating the bias

- Add a dummy feature x_0 which always takes the value 1. The weight w_0 is equivalent to a bias.

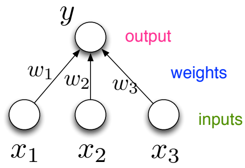
Simplified model

$$z = \mathbf{w}^T\mathbf{x}$$

$$y = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

As a neuron

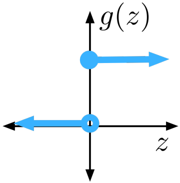
- This is basically a special case of the neuron-like processing unit from Lecture 1.



output bias i'th weight

$$y = g\left(b + \sum_i x_i w_i\right)$$

nonlinearity i'th input



- Today's question: what can we do with a single unit?

Examples

NOT

x_0	x_1	t
1	0	1
1	1	0

Examples

NOT

x_0	x_1	t
1	0	1
1	1	0

$$b > 0$$
$$b + w < 0$$

$$b = 1,\ w = -2$$

Examples

NOT

x_0	x_1	t
1	0	1
1	1	0

$$
\begin{aligned}
b &> 0 \\
b + w &< 0
\end{aligned}
$$

Examples

NOT

x_0	x_1	t
1	0	1
1	1	0

$$b > 0$$
$$b + w < 0$$

$$b = 1,\ w = -2$$

Examples

AND

x_0	x_1	x_2	t
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Examples

AND

x_0	x_1	x_2	t
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$b < 0$

Examples

AND

x_0	x_1	x_2	t
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$$b < 0$$

$$b + w_2 < 0$$

Examples

AND

x_0	x_1	x_2	t
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$$b < 0$$
$$b + w_2 < 0$$
$$b + w_1 < 0$$

Examples

AND

x_0	x_1	x_2	t
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$$b < 0$$
$$b + w_2 < 0$$
$$b + w_1 < 0$$
$$b + w_1 + w_2 > 0$$

Examples

AND

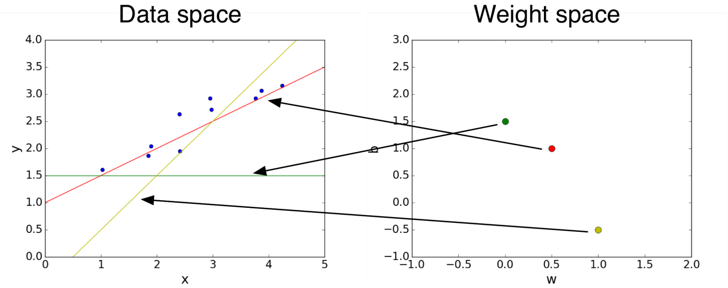
x_0	x_1	x_2	t
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$$\begin{aligned} b &< 0 \\ b + w_2 &< 0 \\ b + w_1 &< 0 \\ b + w_1 + w_2 &> 0 \end{aligned}$$

$$b = -1.5, \; w_1 = 1, \; w_2 = 1$$

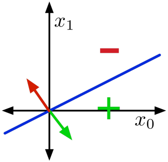
The Geometric Picture

Recall from linear regression:



The Geometric Picture

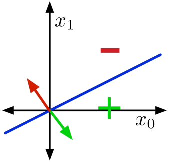
Input Space, or **Data Space**

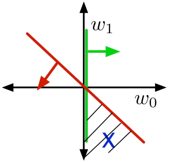


- Here we're visualizing the **NOT** example
- Training examples are points
- Hypotheses are half-spaces whose boundaries pass through the origin
- The boundary is the decision boundary
  - In 2-D, it's a line, but think of it as a hyperplane
- If the training examples can be separated by a linear decision rule, they are linearly separable.

The Geometric Picture

Weight Space



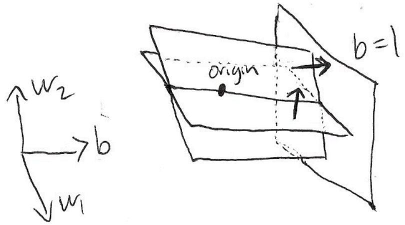


$$w_0 > 0$$
$$w_0 + w_1 < 0$$

- Hypotheses are points
- Training examples are half-spaces whose boundaries pass through the origin
- The region satisfying all the constraints is the feasible region; if this region is nonempty, the problem is feasible

The Geometric Picture

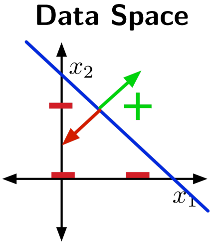
- The **AND** example requires three dimensions, including the dummy one.
- To visualize data space and weight space for a 3-D example, we can look at a 2-D slice:



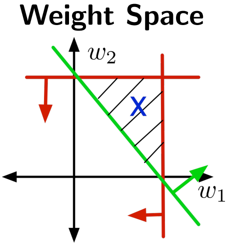
- The visualizations are similar, except that the decision boundaries and the constraints need not pass through the origin.

The Geometric Picture

Visualizations of the **AND** example



Slice for $x_0 = 1$

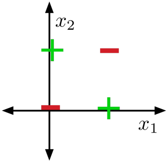


Slice for $w_0 = -1$

What happened to the fourth constraint?

The Geometric Picture

Some datasets are not linearly separable, e.g. **XOR**



The Perceptron Learning Rule

- Let's mention a classic classification algorithm from the 1950s: the perceptron

- Frank Rosenblatt, with the image sensor (left) of the Mark I Perceptron40

The Perceptron Learning Rule

The idea:

- If $t = 1$ and $z = \mathbf{w}^\top \mathbf{x} > 0$
  - then $y = 1$, so no need to change anything.

The Perceptron Learning Rule

The idea:

- If $t = 1$ and $z = \mathbf{w}^\top \mathbf{x} > 0$
  - then $y = 1$, so no need to change anything.
- If $t = 1$ and $z < 0$
  - then $y = 0$, so we want to make z larger.

The Perceptron Learning Rule

The idea:

- If $t = 1$ and $z = \mathbf{w}^\top \mathbf{x} > 0$
  - then $y = 1$, so no need to change anything.
- If $t = 1$ and $z < 0$
  - then $y = 0$, so we want to make z larger.
  - Update:

$$\mathbf{w}' \leftarrow \mathbf{w} + \mathbf{x}$$

The Perceptron Learning Rule

The idea:

- If $t = 1$ and $z = \mathbf{w}^\top \mathbf{x} > 0$
  - then $y = 1$, so no need to change anything.
- If $t = 1$ and $z < 0$
  - then $y = 0$, so we want to make z larger.
  - Update:

$$\mathbf{w}' \leftarrow \mathbf{w} + \mathbf{x}$$

  - Justification:

$$\begin{aligned} \mathbf{w}'^T \mathbf{x} &= (\mathbf{w} + \mathbf{x})^T \mathbf{x} \\ &= \mathbf{w}^T \mathbf{x} + \mathbf{x}^T \mathbf{x} \\ &= \mathbf{w}^T \mathbf{x} + \|\mathbf{x}\|^2. \end{aligned}$$

The Perceptron Learning Rule

For convenience, let targets be $\{-1, 1\}$ instead of our usual $\{0, 1\}$.

Perceptron Learning Rule:

Repeat:

For each training case $(\mathbf{x}^{(i)}, t^{(i)})$,

$$z^{(i)} \leftarrow \mathbf{w}^T \mathbf{x}^{(i)}$$

If $z^{(i)} t^{(i)} \leq 0$,

$$\mathbf{w} \leftarrow \mathbf{w} + t^{(i)} \mathbf{x}^{(i)}$$

Stop if the weights were not updated in this epoch.

The Perceptron Learning Rule

Compare:

- SGD for linear regression

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha(y - t)\,\mathbf{x}$$

- perceptron

$$z \leftarrow \mathbf{w}^T\mathbf{x}$$

$$\text{If } zt \leq 0,$$

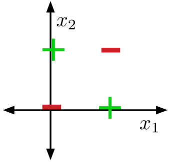
$$\mathbf{w} \leftarrow \mathbf{w} + t\mathbf{x}$$

The Perceptron Learning Rule

- Under certain conditions, if the problem is feasible, the perceptron rule is guaranteed to find a feasible solution after a finite number of steps.
- If the problem is infeasible, all bets are off.
  - Stay tuned...
- The perceptron algorithm caused lots of hype in the 1950s, then people got disillusioned and gave up on neural nets.
- People were discouraged about fundamental limitations of linear classifiers.

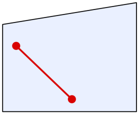
Limits of Linear Classification

- Visually, it's obvious that **XOR** is not linearly separable. But how to show this?



Limits of Linear Classification

Convex Sets



- A set $\mathcal{S}$ is convex if any line segment connecting points in $\mathcal{S}$ lies entirely within $\mathcal{S}$. Mathematically,

$$\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{S} \implies \lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2 \in \mathcal{S} \quad \text{for } 0 \leq \lambda \leq 1.$$

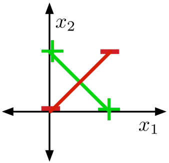
- A simple inductive argument shows that for $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathcal{S}$, weighted averages, or convex combinations, lie within the set:

$$\lambda_1 \mathbf{x}_1 + \cdots + \lambda_N \mathbf{x}_N \in \mathcal{S} \quad \text{for } \lambda_i > 0,\ \lambda_1 + \cdots \lambda_N = 1.$$

Limits of Linear Classification

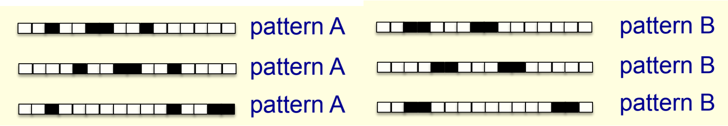
Showing that XOR is not linearly separable

- Half-spaces are obviously convex.
- Suppose there were some feasible hypothesis. If the positive examples are in the positive half-space, then the green line segment must be as well.
- Similarly, the red line segment must line within the negative half-space.



- But the intersection can't lie in both half-spaces. Contradiction!

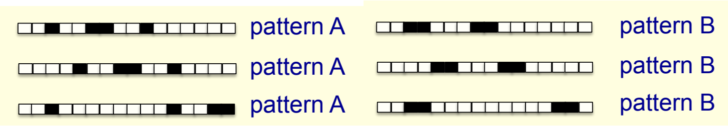
A more troubling example



- These images represent 16-dimensional vectors. White $= 0$, black $= 1$.
- Want to distinguish patterns A and B in all possible translations (with wrap-around)
- Translation invariance is commonly desired in vision!

Limits of Linear Classification

A more troubling example



- These images represent 16-dimensional vectors. White = 0, black = 1.
- Want to distinguish patterns A and B in all possible translations (with wrap-around)
- Translation invariance is commonly desired in vision!
- Suppose there's a feasible solution. The average of all translations of A is the vector $(0.25, 0.25, \ldots, 0.25)$. Therefore, this point must be classified as A.
- Similarly, the average of all translations of B is also $(0.25, 0.25, \ldots, 0.25)$. Therefore, it must be classified as B. Contradiction!

Credit: Geoffrey Hinton

Limits of Linear Classification

- Sometimes we can overcome this limitation using feature maps, just like for linear regression. E.g., for **XOR**:

$$\phi(\mathbf{x}) = \begin{pmatrix} x_1 \\ x_2 \\ x_1 x_2 \end{pmatrix}$$

x_1	x_2	$\phi_1(\mathbf{x})$	$\phi_2(\mathbf{x})$	$\phi_3(\mathbf{x})$	t
0	0	0	0	0	0
0	1	0	1	0	1
1	0	1	0	0	1
1	1	1	1	1	0

- This is linearly separable. (Try it!)
- Not a general solution: it can be hard to pick good basis functions. Instead, we'll use neural nets to learn nonlinear hypotheses directly.