Overview of Machine Learning

in particular, Supervised Learning

Chris Cornwell

Mar 13, 2025

Outline

Machine Learning

Supervised learning

Perceptron algorithm

Outline

Machine Learning

Supervised learning

Perceptron algorithm

What is Machine Learning?

Definition by Tom Mitchell:

> *A "computer program" is said to* ***learn*** *from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

- The definition is intentionally general. Often, could think of E as "training" (updates to how program runs), based on observed data.
- "computer program" (for us) means a procedure or function, implemented on a computer, that produces output from given input. The output is how the program is supposed to achieve the task T.
- The procedures discussed in class – linear regression and the Perceptron algorithm for half-space model – fit into this paradigm...*kind of*.

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to **learn** from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

1. Linear regression.
   - “program”: the process taking input (x, potentially multiple variables), “predicting” a label $\hat{y}$. (with $\hat{y} = \hat{m}x + \hat{b}$.)
   - T: fit observed points $(x_1, y_1), \ldots, (x_n, y_n)$ well with predictions $(x_1, \hat{y}_1), \ldots, (x_n, \hat{y}_n)$, with expectation of good fit on *unobserved* data.
   - E: ??
     The data are used to get $\hat{m}$ and $\hat{b}$, but you don’t really “improve” with repeated use of data.
     A closed form for best choice of $\hat{m}, \hat{b}$: compute $(A^T A)^{-1} A^T \mathbf{y}$.
   - P: Mean squared error.

   One should not expect nice closed form in general.

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to **learn** from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

2. The Perceptron algorithm.
   - "program": the process taking input ($\mathbf{x} \in \mathbb{R}^d$, or something *turned into* $\mathbf{x} \in \mathbb{R}^d$), "predicting" a label $+1$ or -1. (using $W = (\mathbf{w}, b) \in \mathbb{R}^{d+1}$ to decide label.)
   - T: predicting labels correctly…including on *unobserved* data.
   - E: looking through observed data $X_i = (\mathbf{x}_i, 1)$, label y_i, and updating $W^{(t+1)} = W^{(t)} + y_i X_i$ when i found with $W^{(t)} \cdot (y_i X_i) \leq 0$.
   - P: ??

     Whether its labels on all observed data are correct. But, only two results: *True* or *False*.

     If data is linearly separable, enough of experience E improves this measure (changing to *True*). Only happens if linearly separable.

What types of <u>tasks</u> and <u>algorithms</u> in machine learning?

Outline

Machine Learning

Supervised learning

Perceptron algorithm

The goal of supervised learning

How to achieve the goal

Linearly separable

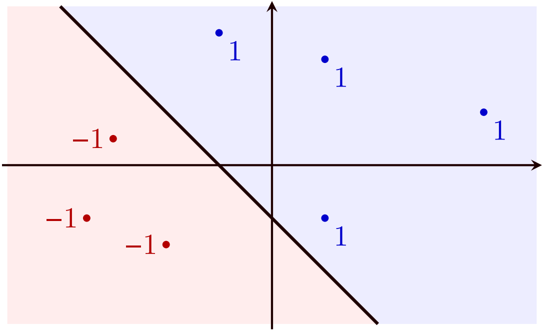

Figure: The hyperplane $H = \{(x_1, x_2) \in \mathbb{R}^2 : x_1 + x_2 + 1 = 0\}$, corresponding positive and negative regions, $\mathbf{w} = (1, 1)$, $b = 1$

Not linearly separable

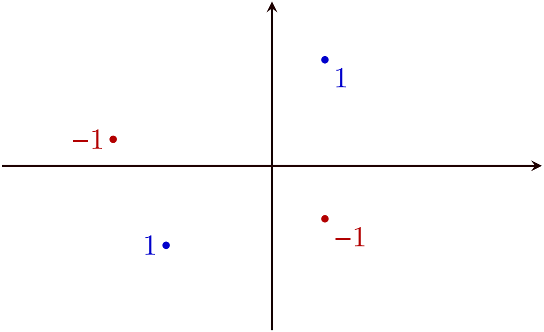

Figure: A data set in $\mathbb{R}^2$ that is not linearly separable.

Not linearly separable

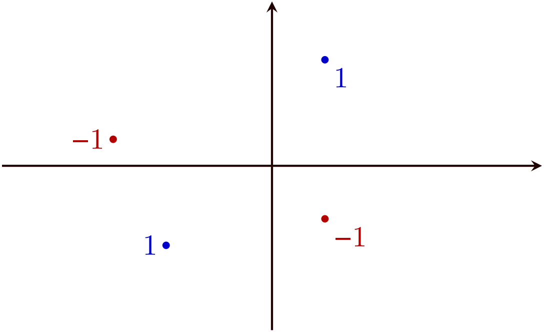

Figure: A data set in $\mathbb{R}^2$ that is not linearly separable.

- A criterion (checkable, in theory) that is equivalent to "not linearly separable"?

Outline

Machine Learning

Supervised learning

Perceptron algorithm

Setup for Perceptron algorithm

Labeled data: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{\pm 1\}$ for all i.

Assuming labeled data is linearly separable, the Perceptron algorithm is a procedure that is guaranteed to find a hyperplane that separates the data.[1]

---

[1] Introduced in *The perceptron: A probabilistic model for information storage and organization in the brain*, F. Rosenblatt, Psychological Review **65** (1958), 386–407.

Setup for Perceptron algorithm

Labeled data: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{\pm 1\}$ for all i.
Assuming labeled data is linearly separable, the Perceptron algorithm is a procedure that is guaranteed to find a hyperplane that separates the data.[1]
To describe it: for each $\mathbf{x}_i$, use X_i to denote the $(d+1)$-vector consisting of $\mathbf{x}_i$ with 1 appended at the end;
Additionally, use W to denote the vector $\mathbf{w}$ with b appended at the end.

---

[1] Introduced in *The perceptron: A probabilistic model for information storage and organization in the brain*, F. Rosenblatt, Psychological Review **65** (1958), 386–407.

Setup for Perceptron algorithm

Labeled data: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{\pm 1\}$ for all i.
Assuming labeled data is linearly separable, the Perceptron algorithm is a procedure that is guaranteed to find a hyperplane that separates the data.[1]
To describe it: for each $\mathbf{x}_i$, use X_i to denote the $(d+1)$-vector consisting of $\mathbf{x}_i$ with 1 appended at the end;
Additionally, use W to denote the vector $\mathbf{w}$ with b appended at the end.
Note that $W \cdot X_i = \mathbf{w} \cdot \mathbf{x}_i + b$.
For linearly separable data, our goal is to find $W \in \mathbb{R}^{d+1}$ so that $W \cdot X_i$ and y_i have the same sign (both positive or both negative), for all $1 \leq i \leq n$.

- Equivalently, we need $y_i W \cdot X_i > 0$ for all $1 \leq i \leq n$.

---

[1] Introduced in *The perceptron: A probabilistic model for information storage and organization in the brain*, F. Rosenblatt, Psychological Review **65** (1958), 386–407.

Perceptron algorithm

Suppose the data is linearly separable. Also, `x` is an $n \times d$ array of points, with ith row equal to $\mathtt{x}_i$, and `y` is array of the labels. The Perceptron algorithm finds `W` iteratively as follows.[2]

[2] Recall, in pseudo-code block, left-facing arrow means *assign* to variable on left.

Perceptron algorithm

Suppose the data is linearly separable. Also, `x` is an $n \times d$ array of points, with ith row equal to $\mathbf{x}_i$, and `y` is array of the labels. The Perceptron algorithm finds `W` iteratively as follows.[2]

```
input: x, y  ## x is n by d, y is 1d array
X← append 1 to each row of x
W← (0,0,...,0)  ## Initial W
while (exists i with y[i]*dot(W, X[i]) ≤ 0){
    W←W + y[i]*X[i]
}
return W
```

[2]Recall, in pseudo-code block, left-facing arrow means *assign* to variable on left.

Perceptron algorithm, stopping time

Under our assumptions for Perceptron algorithm, a guarantee on eventually stopping.

Theorem

Define $R = \max_i |X_i|$ *and* $B = \min_i\{|V| : \forall i, y_i V \cdot X_i \geq 1\}$*. Then, the Perceptron algorithm stops after at most* $(RB)^2$ *iterations and, when it stops with output* W*, then* $y_i W \cdot X_i > 0$ *for all* $1 \leq i \leq n$*.*

Perceptron algorithm, stopping time

Under our assumptions for Perceptron algorithm, a guarantee on eventually stopping.

Theorem

Define $R = \max_i |X_i|$ *and* $B = \min_i\{|V| : \forall i, y_i V \cdot X_i \geq 1\}$. *Then, the Perceptron algorithm stops after at most* $(RB)^2$ *iterations and, when it stops with output* W*, then* $y_i W \cdot X_i > 0$ *for all* $1 \leq i \leq n$.

Idea of proof: Write W^* for vector that realizes the minimum B. Also, write $W^{(t)}$ for the vector W on the t^{th} step, with $W^{(1)} = (0, 0, \ldots, 0)$.

Perceptron algorithm, stopping time

Under our assumptions for Perceptron algorithm, a guarantee on eventually stopping.

Theorem

Define $R = \max_i |X_i|$ and $B = \min_i\{|V| : \forall i, y_i V \cdot X_i \geq 1\}$. Then, the Perceptron algorithm stops after at most $(RB)^2$ iterations and, when it stops with output W, then $y_i W \cdot X_i > 0$ for all $1 \leq i \leq n$.

Idea of proof: Write W^* for vector that realizes the minimum B. Also, write $W^{(t)}$ for the vector W on the t^{th} step, with $W^{(1)} = (0, 0, \ldots, 0)$. Using how $W^{(t+1)}$ is obtained from $W^{(t)}$, can show that $W^* \cdot W^{(T+1)} \geq T$ after $T + 1$ iterations. Also, using the condition on $W^{(T)}$ that necessitates an update, can show that $|W^{(T+1)}| \leq \sqrt{T}R$. (Both statements, use induction.)

Perceptron algorithm, stopping time

Under our assumptions for Perceptron algorithm, a guarantee on eventually stopping.

Theorem

Define $R = \max_i |X_i|$ and $B = \min_i\{|V| : \forall i, y_i V \cdot X_i \geq 1\}$. Then, the Perceptron algorithm stops after at most $(RB)^2$ iterations and, when it stops with output W, then $y_i W \cdot X_i > 0$ for all $1 \leq i \leq n$.

Idea of proof: Write W^* for vector that realizes the minimum B. Also, write $W^{(t)}$ for the vector W on the t^{th} step, with $W^{(1)} = (0, 0, \ldots, 0)$. Using how $W^{(t+1)}$ is obtained from $W^{(t)}$, can show that $W^* \cdot W^{(T+1)} \geq T$ after $T + 1$ iterations. Also, using the condition on $W^{(T)}$ that necessitates an update, can show that $|W^{(T+1)}| \leq \sqrt{T}R$. (Both statements, use induction.)

Now, by Cauchy-Schwarz inequality, $T \leq BR\sqrt{T}$, which we can rearrange to $T \leq (BR)^2$.