Overview of Machine Learning
in particular, Supervised Learning

Chris Cornwell

Mar 13, 2025

Outline

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 function implemented on a computer that produces output from given input. The output is how the program achieves 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.
   - Output of $\hat{y}$ on input x (potentially multiple variables).
   - T: fit observed points $\{(x_i, y_i)\}_{i=1}^n$ well with predictions $\{(x_i, \hat{y}_i)\}$, where $\hat{y}_i = mx_i + b$ for some m, b (an expectation of $(x, \hat{y})$ being good fit on *unobserved* data.
   - E: ??

     The data are used to get m and b, but you don't really "improve" with repeated use of data.

     Closed form for best choice of m, b, computing $(A^T A)^{-1} A^T \mathbf{y}$.
   - P: Mean squared error.

   Having closed form, result of simplicity of the form of $\hat{y}_i$.

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.
   - Output of label ± 1 on input $\mathbf{x} \in \mathbb{R}^d$ (or something *turned into* $\mathbf{x} \in \mathbb{R}^d$).
   - *T*: predict labels correctly, using $W = (\mathbf{w}, b) \in \mathbb{R}^{d+1}$ to decide label, $y = \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} + b)$
     ...hopefull works 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 are the general types of tasks in machine learning?

Supervised learning: the program learns from sample data that has labels. Goal: determine underlying function from sample data.

- Housing price prediction
- Whether emails are phishing or not phishing.
- Determine if a satellite image of ocean has floating trash.
- Try to auto-complete a sentence being typed.

Unsupervised learning: there is sample data, but the data does not have any labels. Goal: discover something (a pattern, grouping, or some insight) about the data.

- Business application: Market segmentation.
- News feed (grouping similar news articles).
- Separate audio sources in a mixed signal.
- Organize computing clusters.

Reinforcement learning.

Outline

The goal of supervised learning

Have an "input space" $\mathbb{R}^d$ (could be more general space) and output space, or label space, Y.

From a sample $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in Y$, drawn from an (unknown) probability distribution $P_{X,Y} : \mathbb{R}^d \times Y \rightarrow [0, \infty)$.

Goal is to learn, from $\mathcal{S}$, a function $f : \mathbb{R}^d \rightarrow Y$ that "fits" (*approximates*) well the distribution $P_{X,Y}$. You might not be able to have the graph of f be such that it is typically very "close" to samples from $P_{X,Y}$. However, ideally, for an $\mathbf{x} \in \mathbb{R}^d$ the point on the graph is approximately the expected value, given $\mathbf{x}$.

How to achieve the goal

Most often, you choose a *parameterized class* of functions; i.e., there is a parameter space Ω, and an $\omega \in \Omega$ determines a function $f_\omega : \mathbb{R}^d \longrightarrow Y$.
To learn a function that fits well, you find a set of parameters.
The performance measure: **(empirical) loss function** $\mathcal{L}_\mathcal{S} : \Omega \to \mathbb{R}$. (The definition of the empirical loss function uses $\mathcal{S}$ in its definition.)

For linear regression

Have sample data $\mathcal{S}$, with data points x_i in $\mathbb{R}$ (so, $d = 1$). The parameter space $\Omega = \mathbb{R}^2 = \{(m, b) \mid m \in \mathbb{R}, b \in \mathbb{R}\}$, and if $\omega_0 = (m_0, b_0)$ then

$$f_{\omega_0}(x) = m_0 x + b_0.$$

Loss function, use the MSE. That is, set

$$\mathcal{L}_{\mathcal{S}}(m, b) = \frac{1}{n}\sum_{i=1}^{n}(mx_i + b - y_i)^2.$$

Outline

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.[1]

[1] 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 $\mathtt{x}_i$, and y is array of the labels. The Perceptron algorithm finds W iteratively as follows.[1]

```
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
```

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