image from: alphr.com

MACHINE LEARNING & DATA MINING

A question from IQ test

Following the pattern shown in the number sequence below, what is the missing number?

1, 8, 27, ?, 125, 216

Possible answers:

- 36
- 45
- 46
- 64
- 99

Defined as an ML task

X_{train}	y_{train}
1	1
2	8
3	27
5	125
6	216

$$X_{\text{test}} = (4,)$$

My solution

$$\hat{y} = \frac{1}{12}\left(35x^5 - 595x^4 + 3757x^3 - 10745x^2 + 13860x - 6300\right)$$

Fits perfectly! My answer:

- 99

Which solution is better?

Is this solution:

$$\hat{y} = x^3$$

better than this one:

$$\hat{y} = \frac{1}{12}\left(35x^5 - 595x^4 + 3757x^3 - 10745x^2 + 13860x - 6300\right)?$$

Which solution is better?

Is this solution:

$$\hat{y} = x^3$$

better than this one:

$$\hat{y} = \frac{1}{12}\left(35x^5 - 595x^4 + 3757x^3 - 10745x^2 + 13860x - 6300\right)?$$

One can argue that:

- First one is more suitable **in the context of IQ test**

Which solution is better?

Is this solution:

$$\hat{y} = x^3$$

better than this one:

$$\hat{y} = \frac{1}{12}\left(35x^5 - 595x^4 + 3757x^3 - 10745x^2 + 13860x - 6300\right)?$$

One can argue that:

- First one is more suitable **in the context of IQ test**

A priori knowledge about the target

No Free Lunch theorems

(informally stated)

If we don’t have any prior knowledge about the data:

- All algorithms perform the same on average (over all possible targets)

Simple example

Simple example

- Discrete finite feature space: $x \in \{1, 2, 3, 4, 5\}$

Simple example

- Discrete finite feature space: $x \in \{1, 2, 3, 4, 5\}$
- Binary classification task: $y \in \{0, 1\}$

Simple example

- Discrete finite feature space: $x \in \{1, 2, 3, 4, 5\}$
- Binary classification task: $y \in \{0, 1\}$
- '0-1' loss: $L(y_1, y_2) = 1 - \delta[y_1, y_2]$

Simple example

- Discrete finite feature space: $x \in \{1, 2, 3, 4, 5\}$
- Binary classification task: $y \in \{0, 1\}$
- '0-1' loss: $L(y_1, y_2) = 1 - \delta[y_1, y_2]$
- There are $2^5 = 32$ possible $X \to Y$ target mappings

Simple example

- Discrete finite feature space: $x \in \{1, 2, 3, 4, 5\}$
- Binary classification task: $y \in \{0, 1\}$
- '0-1' loss: $L(y_1, y_2) = 1 - \delta[y_1, y_2]$
- There are $2^5 = 32$ possible $X \to Y$ target mappings
- Split data to train and test parts of sizes 4:1

Simple example

- Discrete finite feature space: $x \in \{1, 2, 3, 4, 5\}$
- Binary classification task: $y \in \{0, 1\}$
- '0-1' loss: $L(y_1, y_2) = 1 - \delta[y_1, y_2]$
- There are $2^5 = 32$ possible $X \rightarrow Y$ target mappings
- Split data to train and test parts of sizes 4:1
- Consider two algorithms:

Simple example

- Discrete finite feature space: $x \in \{1, 2, 3, 4, 5\}$
- Binary classification task: $y \in \{0, 1\}$
- '0-1' loss: $L(y_1, y_2) = 1 - \delta[y_1, y_2]$
- There are $2^5 = 32$ possible $X \to Y$ target mappings
- Split data to train and test parts of sizes 4:1
- Consider two algorithms:
    A. Predict most popular label of the training set (random guess if draw)

Simple example

- Discrete finite feature space: $x \in \{1, 2, 3, 4, 5\}$
- Binary classification task: $y \in \{0, 1\}$
- '0-1' loss: $L(y_1, y_2) = 1 - \delta[y_1, y_2]$
- There are $2^5 = 32$ possible $X \to Y$ target mappings
- Split data to train and test parts of sizes 4:1
- Consider two algorithms:
  A. Predict most popular label of the training set (random guess if draw)
  B. Predict least popular label of the training set (random guess if draw)

Simple example

- Discrete finite feature space: $x \in \{1, 2, 3, 4, 5\}$
- Binary classification task: $y \in \{0, 1\}$
- '0-1' loss: $L(y_1, y_2) = 1 - \delta[y_1, y_2]$
- There are $2^5 = 32$ possible $X \to Y$ target mappings
- Split data to train and test parts of sizes 4:1
- Consider two algorithms:
  A. Predict most popular label of the training set (random guess if draw)
  B. Predict least popular label of the training set (random guess if draw)
- Evaluate expected loss (over all possible target mappings and train-test splits)

Definitions

Definitions

- Finite input set X, $|X| = n$

Definitions

- Finite input set X, $|X| = n$
- Finite output set Y, $|Y| = r$

Definitions

- Finite input set X, $|X| = n$
- Finite output set Y, $|Y| = r$
- A loss function $L(y_1, y_2) \in \mathbb{R}$

Definitions

- Finite input set X, $|X| = n$
- Finite output set Y, $|Y| = r$
- A loss function $L(y_1, y_2) \in \mathbb{R}$
- Target relationship f. The probability of x to have label y is $f(x, y)$

Definitions

- Finite input set X, $|X| = n$
- Finite output set Y, $|Y| = r$
- A loss function $L(y_1, y_2) \in \mathbb{R}$
- Target relationship f. The probability of x to have label y is $f(x, y)$
- A training set d of (x, y) pairs $(x \in d_X \subset X, y \in d_Y \subset Y)$, sampled from some $P(d \mid f)$, $|d| = m$

Definitions

- Finite input set X, $|X| = n$
- Finite output set Y, $|Y| = r$
- A loss function $L(y_1, y_2) \in \mathbb{R}$
- Target relationship f. The probability of x to have label y is $f(x, y)$
- A training set d of (x, y) pairs $(x \in d_X \subset X, y \in d_Y \subset Y)$, sampled from some $P(d \mid f)$, $|d| = m$
- A test point $q \in X$, $q \notin d_X$

Definitions

- Finite input set X, $|X| = n$
- Finite output set Y, $|Y| = r$
- A loss function $L(y_1, y_2) \in \mathbb{R}$
- Target relationship f. The probability of x to have label y is $f(x, y)$
- A training set d of (x, y) pairs $(x \in d_X \subset X, y \in d_Y \subset Y)$, sampled from some $P(d \mid f)$, $|d| = m$
- A test point $q \in X$, $q \notin d_X$
- A true label for the test point $y_F \in Y$, obtained by sampling Y according to $f(q, \cdot)$

Definitions

- Finite input set X, $|X| = n$
- Finite output set Y, $|Y| = r$
- A loss function $L(y_1, y_2) \in \mathbb{R}$
- Target relationship f. The probability of x to have label y is $f(x, y)$
- A training set d of (x, y) pairs $(x \in d_X \subset X, y \in d_Y \subset Y)$, sampled from some $P(d \mid f)$, $|d| = m$
- A test point $q \in X$, $q \notin d_X$
- A true label for the test point $y_F \in Y$, obtained by sampling Y according to $f(q, \cdot\,)$
- A classifier $P(y_H \mid q, d)$ and a prediction $y_H \in Y$

Definitions

- Finite input set X, $|X| = n$
- Finite output set Y, $|Y| = r$
- A loss function $L(y_1, y_2) \in \mathbb{R}$
- Target relationship f. The probability of x to have label y is $f(x, y)$
- A training set d of (x, y) pairs $(x \in d_X \subset X, y \in d_Y \subset Y)$, sampled from some $P(d \mid f)$, $|d| = m$
- A test point $q \in X$, $q \notin d_X$
- A true label for the test point $y_F \in Y$, obtained by sampling Y according to $f(q, \cdot\,)$
- A classifier $P(y_H \mid q, d)$ and a prediction $y_H \in Y$
- Cost $c = L(y_H, y_F)$

Definitions (2)

'Homogeneous' loss:

- Such loss function that:

$$\forall c \in \mathbb{R}, y_H \in Y :$$

$$\sum_{y_F} \delta\left(c, L(y_H, y_F)\right) = \Lambda(c)$$

  i.e. the value of the sum does not depend on y_H

- This means such loss function has no a priori preference for one value of Y over another
- $L(y_1, y_2) = 1 - \delta[y_1, y_2]$ — is homogeneous
- $L(y_1, y_2) = (y_1 - y_2)^2$ — is not

Definitions (3)

'Vertical' $P(d\,|\,f)$:

- Likelihood $P(d\,|\,f)$ determines how d was generated from f
- It is called 'vertical' if it doesn't depend on $f(x \notin d_X, y)$
- This property prevents data leakage
- Honest sampling of x from some $\pi(x)$ and then choosing the associated y by sampling from $f(x, \cdot\,)$ would result in a vertical likelihood:

$$P(d\,|\,f) = \prod_{i=1}^{m} \pi(d_X(i))\, f(d_X(i), d_Y(i))$$

No Free Lunch theorems

(formally stated)

Theorem 1 For homogeneous loss L, the uniform average over all f of $P(c \,|\, d, f)$ equals $\Lambda(c)/r$.

Theorem 2 For off training set error, a vertical $P(d \,|\, f)$, and a homogeneous loss L, the uniform average over all targets f of $P(c \,|\, f, m)$ equals $\Lambda(c)/r$.

Discussion

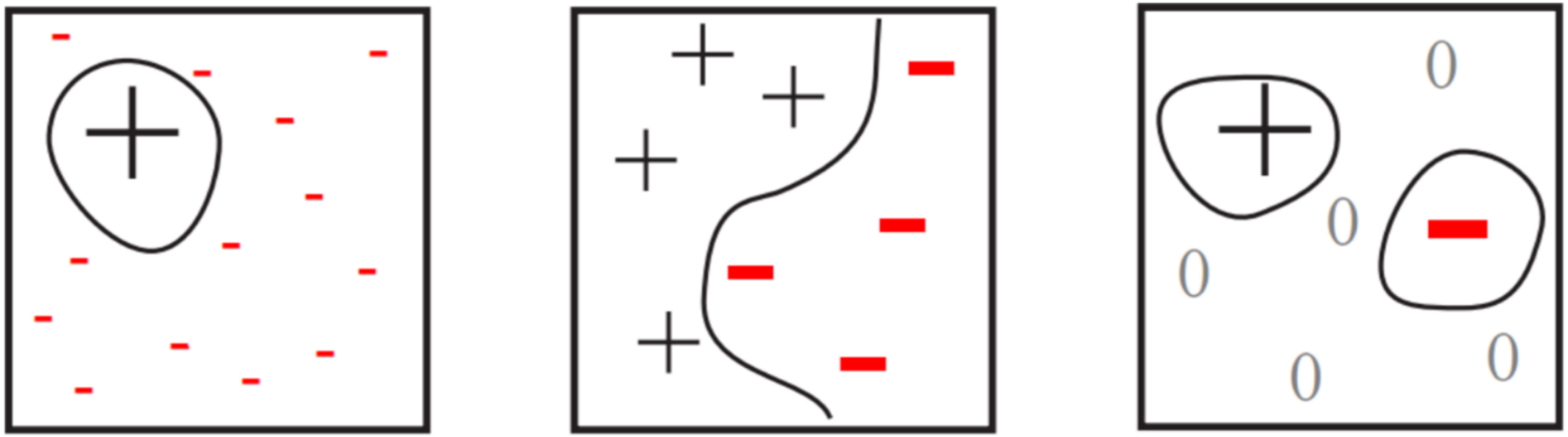
- No free lunch theorem states that **on average over all datasets** all learning algorithms are equally bad at learning. E.g.:
  - some crazy algorithm:

$$\hat{y} = \frac{\left\lfloor \sum_i (x_i)^{1+\sqrt{i+\pi}} \right\rfloor \mod 17}{\left\lceil \sum_i (x_i)^{-1+\sqrt{i+\pi}} \right\rceil \mod 13}$$

  - any configuration of SVM with cross-validation

  perform equally **on average**.

Discussion



Possible learning algorithm behaviors in **problem space**

- + better than the average
- – worse than the average

Is machine learning useless?

Is machine learning useless?

No

Why not?

- No free lunch theorem tests an algorithm against **all possible problems**

Why not?

- No free lunch theorem tests an algorithm against **all possible problems**

The space of all possible problems is HUGE!

Why not?

- No free lunch theorem tests an algorithm against **all possible problems**

The space of all possible problems is HUGE!

- E.g. for $|X| = 270$ and binary classification there are 2^{270} problems (around the number of atoms in the Universe)

Why not?

- No free lunch theorem tests an algorithm against **all possible problems**

The space of all possible problems is HUGE!

- E.g. for $|X| = 270$ and binary classification there are 2^{270} problems (around the number of atoms in the Universe)
- In real world we have:

Why not?

- No free lunch theorem tests an algorithm against **all possible problems**

The space of all possible problems is HUGE!

- E.g. for $|X| = 270$ and binary classification there are 2^{270} problems (around the number of atoms in the Universe)
- In real world we have:
  - **data scientist** with prior knowledge of the world

Why not?

- No free lunch theorem tests an algorithm against **all possible problems**

The space of all possible problems is HUGE!

- E.g. for $|X| = 270$ and binary classification there are 2^{270} problems (around the number of atoms in the Universe)
- In real world we have:
  - **data scientist** with prior knowledge of the world
  - problem description (recall the IQ test example)

Why not?

- No free lunch theorem tests an algorithm against **all possible problems**

The space of all possible problems is HUGE!

- E.g. for $|X| = 270$ and binary classification there are 2^{270} problems (around the number of atoms in the Universe)
- In real world we have:
  - **data scientist** with prior knowledge of the world
  - problem description (recall the IQ test example)
  - data description

Literature

Wolpert, David (1996), "The Lack of A Priori Distinctions between Learning Algorithms", Neural Computation, pp. 1341-1390

Wolpert, David H. ”The supervised learning no-free-lunch theorems.” Soft computing and industry. Springer London, 2002. 25-42

The example is borrowed from: https://github.com/kazeevn/no_free_lunch (there are also nicely presented proofs there).